UNIVERSITAT DE BARCELONA

Adapting by copying.
Towards a sustainable machine learning

Irene Unceta

UNIVERSITAT DE BARCELONA

Adapting by copying
Towards a sustainable machine learning

Irene Unceta

Departament de Matemàtiques i Informàtica
Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

A dissertation submitted in partial fulfillment of
the requirements for the Degree of Doctor in Philosophy

Supervisors Oriol Pujol & Jordi Nin

November 2020

Abstract

Despite the rapid growth of machine learning in the past decades, deploying automated decision making systems in practice remains a challenge for most companies. On an average day, data scientists face substantial barriers to serving models into production. Production environments are complex ecosystems, still largely based on *on-premise* technology, where modifications are timely and costly. Given the rapid pace with which the machine learning environment changes these days, companies struggle to stay up-to-date with the latest software releases, the changes in regulation and the newest market trends. As a result, machine learning often fails to deliver according to expectations. And more worryingly, this can result in unwanted risks for users, for the company itself and even for the society as a whole, insofar the negative impact of these risks is perpetuated in time. In this context, adaptation is an instrument that is both necessary and crucial for ensuring a sustainable deployment of industrial machine learning.

This dissertation is devoted to developing theoretical and practical tools to enable adaptation of machine learning models in company production environments. More precisely, we focus on devising mechanisms to exploit the knowledge acquired by models to train future generations that are better fit to meet the stringent demands of a changing ecosystem. We introduce copying as a mechanism to replicate the decision behaviour of a model using another that presents differential characteristics, in cases where access to both the models and their training data are restricted. We discuss the theoretical implications of this methodology and show how it can be performed and evaluated in practice. Under the conceptual framework of actionable accountability we also explore how copying can be used to ensure risk mitigation in circumstances where deployment of a machine learning solution results in a negative impact to individuals or organizations.

Resum

Malgrat el ràpid creixement que ha experimentat l'ús de l'aprenentatge automàtic, la seva implementació continua sent un repte per a moltes empreses. Els científics de dades s'enfronten diàriament a nombroses barreres a l'hora de desplegar els models en els entorns productius. Aquests entorns són complexos, majoritàriament basats en tecnologies *on-premise*, on els canvis són costosos. És per això que les empreses tenen dificultats per a mantenir-se al dia amb les versions de programari, els canvis en la regulació o les tendències del mercat. Com a conseqüència, el rendiment de l'aprenentatge automàtic està sovint per sota de les expectatives. I cosa que és més preocupant, això pot derivar en riscos per als usuaris, per a les pròpies empreses i per a la societat en el seu conjunt, quan l'impacte negatiu d'aquests riscos es mantingui en el temps. En aquest context, l'adaptació es un element necessari i imprescindible per a assegurar la sostenibilitat.

Aquest treball està dedicat a desenvolupar les eines teòriques i pràctiques necessàries per a possibilitar l'adaptació dels models d'aprenentatge automàtic en entorns de producció. En concret, ens centrem en concebre mecanismes per reutilitzar el coneixement adquirit pels models d'aprenentatge automàtic per a entrenar futures generacions que satisfaguin les demandes d'un entorn altament canviant. Introduïm la idea de copiar, com un mecanisme que permet replicar el comportament decisori d'un model incorporant característiques diferencials, en escenaris on l'accés tant a les dades com al propi model està restringit. Discutim les implicacions teòriques d'aquesta metodologia i demostrem com les còpies poden ser entrenades i avaluades a la pràctica. Sota el marc de la responsabilitat accionable, explorem com les còpies poden explotar-se per a la mitigació de riscos quan el desplegament d'una solució basada en l'aprenentatge automàtic pugui tenir un impacte negatiu sobre les persones o les organitzacions.

Laburpena

Azken urteetan ikaskuntza automatikoa azkar hazi bada ere, erabakiak hartzeko sistema automatizatuak ezartzea erronka handia da enpresa askorentzat. Datu-zientzialariek egunero oztopo ugari izaten dituzte modeloak produkzioan zabaltzeko orduan. Produkzio-inguruneak ekosistema konplexuak dira, batez ere *on-premise* teknologietan oinarritzen direnak, non aldaketak garestiak diren. Horregatik, enpresek zailtasun handiak dituzte softwarearen azken bertsioekin, araudiaren aldaketekin edota merkatuaren joera berriekin eguneratuta egoteko. Ondorioz, askotan ikaskuntza automatikoaren errendimendua itxaropenen oso azpitik dago. Eta are kezkagarriagoa dena, arriskuak ekar ditzake erabiltzaileentzat, enpresentzat eta gizarte osoarentzat, arrisku horien eragin negatiboa denboran zehar iraunarazten den neurrian. Testuinguru honetan, ikaskuntza automatikoaren garapen industrialaren jasangarritasuna bermatzeko beharrezko eta ezinbesteko elementua da egokitzapena.

Lan honen xedea produkzio-inguruneetan ikaskuntza automatikoa egokitzeko beharrezkoak diren tresna teoriko eta praktikoak garatzea da. Zehazki, eskaera aldakorrak asetzeko hobeto prestatuta dauden belaunaldi berriak entrenatu ahal izateko modeloek bereganatutako ezagutza berrerabiltzeko mekanismoak garatzean zentratzen gara. Kopiak aurkezten ditugu modelo baten jokabide erabakitzailea erreplikatzeko aukera ematen duten mekanismo gisa, datuak eta modeloa bera eskuratzeko aukera mugatuta dagoen agertokietan. Metodologia honen inplikazio teorikoak eztabaidatzen ditugu eta kopiak praktikan nola entrenatu eta ebaluatu daitezkeen erakusten dugu. Halaber, erantzukizun eragingarriaren esparruan, ikaskuntza automatikoan oinarritutako irtenbide bat hedatzeak pertsonengan edo erakundeetan eragin negatiboa izan dezakeen kasuetan, kopiak arriskuak arintzeko tresna gisa nola ustiatu daitezkeen aztertzen dugu.

Resumen

A pesar del rápido crecimiento del aprendizaje automático en últimas décadas, su implementación sigue siendo un reto para muchas empresas. Los científicos de datos se enfrentan a diario a numerosas barreras a la hora de desplegar los modelos en producción. Los entornos de producción son ecosistemas complejos, a menudo basados en tecnologías *on-premise*, donde los cambios son costosos. Es por eso que las empresas tienen serias dificultades para mantenerse al día con las últimas versiones de software, los cambios en la regulación o las tendencias del mercado. Como consecuencia, el rendimiento del aprendizaje automático está muy por debajo de las expectativas. Y lo que es más preocupante, esto puede derivar en riesgos para los usuarios, para las propias empresas e incluso para la sociedad en su conjunto, mientras el impacto negativo de dichos riesgos se perpetúe en el tiempo. En este contexto, la adaptación se revela como un elemento necesario e imprescindible para asegurar la sostenibilidad.

Este trabajo está dedicado a desarrollar herramientas que posibiliten la adaptación de los modelos de aprendizaje automático en entornos de producción. Nos centramos en concebir mecanismos para reutilizar el conocimiento adquirido por los modelos para entrenar futuras generaciones mejor preparadas para satisfacer las demandas de un entorno cambiante. Introducimos la copia como un mecanismo para replicar el comportamiento decisorio de un modelo dotándolo de características diferenciales, en escenarios con información restringida. Discutimos las implicaciones teóricas de esta metodología y demostramos como las copias pueden ser entrenadas y evaluadas en la práctica. Bajo el marco de la responsabilidad accionable, exploramos cómo las copias pueden explotarse para la mitigación de riesgos cuando despliegue de una solución basada en el aprendizaje automático deriva en un impacto negativo sobre las personas o las organizaciones.

Acknowledgments

Sentada en la butaca bajo el porche del jardín, con la pose erguida, el traje gris, las uñas de los pies pintadas de rosa claro y el libro en el regazo, Amama, a sus 90 años y pico, contó el otro día que la vida se le ha pasado en un soplo. A mi también estos tres años.

My first and deepest words of gratitude go to my supervisors Jordi Nin and Oriol Pujol without whom this work would not have been possible. I thank them for their direction and their patience along these years. And above all, for their kindness and friendship. This hasn't always been an easy path, but has been a path that I haven't walked alone.

A special thanks to the whole team at BBVA Data & Analytics. To Elena Alfaro for believing in this initiative and supporting it along the way. To Roberto and Javi for their warm welcome. To Juan Murillo for his gentle words of advice and his willingness to contribute. To the whole team in Madrid, to whom I have always looked up to, even from far far away. And of course, to the people who used to work at the office in Barcelona and to those who still do. My immense gratitude to all of them. To Alberto Rubio and Jordi Aranda, to whom I have often turned for guidance and who have always answered my call. To Axel, with whom I've shared part of this journey. And specially, to Jose, who accepted the challenge of embarking on this project in the middle of the journey and who has been relentless in his effort to anchor it in reality. I thank him for his wise counsel.

Last but not least, I would like to mention several people that made this work possible with their support and love. My gratitude goes to all of them. To Gonzalo, Iñaki and Pedro, for giving me the opportunity to observe. To the people in ESADE, for giving me the impulse to finish and to start again. To Iñaki, for this learning journey together. To Marc Cuxart, Berta, Genís and Sandra, whose life stories are also mine. Their struggles, their wins and their losses too. To Marc Mela, who I never meet enough. To Arnau and Pere, who sang and played for me. To Núria and Bernat, who came back. To Carla, who never leaves for real. And to Marc Lemus, who has always been at the other side of the line. And above all, I would to thank my parents and brother, to whom I can always go back to.

Mi viaje a la ciencia de datos empezó durante el verano de 2003, cuando me matriculé a la Udako

Gandias Unibertsitatea, también conocida como la UGU; universidad por la cuál tengo el honor de haber sido la primera y única egresada. Mis primeros pasos en Python los dí entonces. Siempre de la mano del profesor Zubillaga, que se esmeró por enseñarme el orden y el buen hacer programador. Años después estas enseñanzas se extendieron también a la física, disciplina por la que opté como opción universitaria y de la que me he ido inexorablemente alejando con el tiempo. Además de a muchos otros ámbitos, en los que he volcado tanto mis alegrias como muchas de mis frustraciones. Por todo ello, no quiero dejar de agradecerle a Zubi el haberme propuesto un camino que he disfrutado recorriendo y, en ocasiones, también dejando atrás. Tampoco quiero dejar de esperar que se quede un rato más y ver a dónde nos lleva todo esto.

Contents

List of Tables

List of Figures

Introduction

Machine learning is rapidly infiltrating critical areas of society that have a substantial impact on people's lives. The impulse of preliminary experiences during the past decades is now flourishing and has led to a growing global market for research and application. From financial and insurance markets [132][221] to autonomous car driving [46][165], the criminal justice system [9] or clinical decision support [92][148], the tendency has prevailed in recent years to devolve decision making to machine learning models. As these different areas of application expand, so does our understanding of the challenges we face when exploiting this technology in practice.

Today, the deployment of machine learning in the industry is far from being sustainable. Company production environments are highly demanding. In delivering machine learning solutions into production, data scientists need to account for all the different elements that interact with a model throughout its lifespan. These include the data and its sources, the choice of software, the technological infrastructure for production, the existing regulatory framework or the different stakeholders and business areas. Constraints imposed on one or several of these elements largely condition how machine learning models are deployed. As a result, off-the-shelf machine learning techniques often yield sub-optimal results or can only be exploited during a limited period of time. A situation which requires new solutions. On top of that, a growing trend in the machine learning community is claiming that improving predictive power ought not to be the sole purpose of the researchers and organizations developing and deploying these models [30][135][205]. Indeed, there exist legitimate concerns about the potential negative impact of reliance upon this technology [5][34][189][160][212]. In recent years, commercial machine learning models have been shown to reproduce discriminatory practices [17][23][31][105][129] against disadvantaged groups, for reasons of gender [33][42], ethnicity [9][41][128][191] or sexual orientation [100]. Concerns have also been raised regarding the lack of safety [29], interpretability[51][95][206][213] or privacy [84][215][218] of this technology; a condition which could entail pernicious consequences.

These findings highlight the need to deepen our understanding of machine learning to move towards a scenario where this technology is not only profitable for companies, but also sustainable and safe for the

society as a whole. A need, in turn, that poses new questions which require new answers. Notably, *Which are the constraints that prevent a sustainable deployment of machine learning? How can we adapt trained machine learning solutions to changes in their environment? How is this problem formalized and which tools do we have at our disposal to solve it? How can we modify models which display potentially dangerous shortcomings have but which have already been served into production? Which control mechanisms can be enforced to prevent undesired negative impacts of machine learning?* This thesis aims to answer some of these questions.

Motivation and objectives

The importance of developing a sustainable framework for machine learning has long been recognized by both the industry and the scientific community. However, most market applications still struggle with overcoming deployment issues in the long run. Existing models are often not capable of adapting to their new environment and are therefore rendered obsolete or substituted by newer ones. This incurs in large economical costs for companies, who need to invest numerous resources on re-building their prediction pipelines, often from scratch.

This thesis addresses the issue of how machine learning models can learn to adapt to their environment by reusing the knowledge acquired from generation to generation. In particular, it studies how this adaptation can be performed in scenarios where there is little to no access to the models or to the data they were trained with. Of particular interest to us are the practical applications of this approach, which are discussed in different levels of detail through both general examples and specific business use cases.

The long-term goal of this research is to ensure a sustainable use of machine learning in company production environments. More specifically, this study seeks to fulfill the following objectives:

- Understand the mechanisms that enable environmental adaptation of machine learning models by building differential replicas of existing classifiers that display a similar behavior, yet are better fit to survive in the considered environment.

- Review current and past practices to transfer knowledge from one form of representation to another and, in doing so, identify those situations where the existing approaches fail to provide a satisfactory answer.

- Develop the theory behind inheritance by copying in order to understand the practical and theoretical consequences of replicating the decision behavior of a model using another that presents additional features and characteristics in scenarios with limited knowledge.

- Evaluate the feasibility of this technique in practice to ensure actionable accountability of machine learning against rapidly changing conditions.

The result of this study is valuable to industry practitioners as well as the scientific community in general in developing more sustainable practices that ensure a successful deployment of machine learning in highly demanding, industrial environments.

Contributions and thesis outline

The general outline for this dissertation is shown in Fig. 1. This thesis is presented as a story in three acts. The first act, *Concept*, lays the basis for our work and describes the conceptual proposal on which the remaining ideas are build. The second act, *Theory*, presents the theoretical and practical background behind our proposed methodological approach. The present document contains the results of an Industrial PhD thesis. As such, it is devoted to developing new knowledge that can be transferred to the appropriate stakeholders to solve pressing real-life problems for specific industrial applications. Hence, the third act, *Practice*, discusses practical applications of this approach for the industry. In the following we summarize the main contributions of each act, or part, separately:

CONCEPT The first part of this thesis discusses the need to provide effective mechanisms to ensure *environmental adaptation* of machine learning models in time. *Chapter 1* introduces the notion of *differential replication* as a technique to reuse the knowledge acquired by a model to train future generations. This discussion has led to the following scientific publication:

- Unceta, I., Nin, J., and Pujol, O. Environmental adaptation and differential replication in machine learning. *Entropy 22*, 10 (2020). doi = 10.3390/e22101122

THEORY The second part of this thesis discusses differential replication in scenarios where access to either the original model or its training data or both are limited. *Chapter 2* provides an overview of related work. *Chapter 3* introduces the notion of *inheritance by copying* and discusses its mathematical foundations. Finally, *Chapter 4* proposes the *empirical fidelity error* as a reliable performance metric and provides empirical proof of the feasibility of copying in practice through different experiments. The content of these chapters has been published in the form of the following contributions:

- Unceta, I., Nin, J., and Pujol, O. Copying machine learning classifiers. *IEEE Access 8*, 11 (2020), 160268–160284. doi = 10.1109/ACCESS.2020.3020638

- Unceta, I., Palacios, D., Nin, J., and Pujol, O. Sampling unknown decision functions to build classifier copies. In *Proceedings of 17th International Conference on Modeling Decisions for Artificial Intelligence* (Sant Cugat, Spain, 2020), pp. 192–204. doi: 10.1007/978-3-030-57524-3_16

PRACTICE The third part of this thesis proposes a practical framework to exploit differential replication through copying as a tool for ensuring that machine learning systems can be deployed safely and

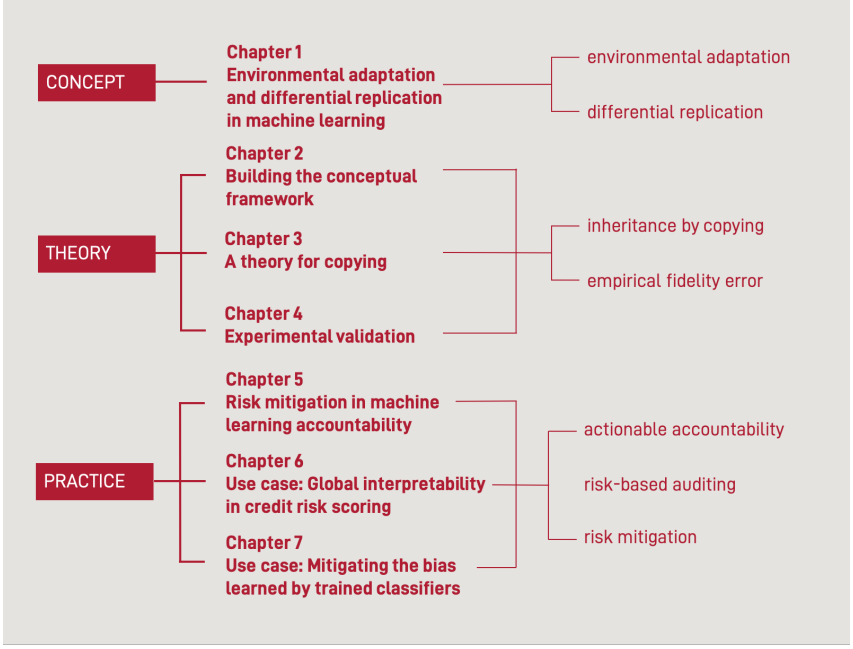


Fig. 1 Diagram of the thesis outline, showing the three parts and the chapters they each include, together with the main concepts discussed.

sustainably in company production environments. *Chapter 5* introduces the notion of *actionable accountability* as a two-staged process which includes both *risk-based auditing* and *risk mitigation*. *Chapter 6* and *Chapter 7* present two real use cases where this tool is applied in practice. Altogether, these ideas have been published under the following titles:

- Unceta, I., Nin, J., and Pujol, O. Risk mitigation in algorithmic accountability: The role of machine learning copies. *PlosONE* (2020). doi = 10.1371/journal.pone.0241286

- Unceta, I., Nin, J., and Pujol, O. Towards global explanations for credit risk scoring. In *Workshop on Challenges and Opportunities for AI in Financial Services: the Impact of Fairness, Explainability, Accuracy, and Privacy (FEAP-AI4Fin)* (Montreal, Canada, 2018)

- Unceta, I., Nin, J., and Pujol, O. Using copies to remove sensitive data: A case study on fair superhero alignment prediction. In *Pattern Recognition and Image Analysis. IbPRIA 2019. Lecture Notes in Computer Science* (Madrid, Spain, 2019), vol. 11867, Springer, Berlin, Heidelberg, pp. 182–193. doi: 10.1007/978-3-030-31332-6_16

- Unceta, I., Nin, J., and Pujol, O. From batch to online learning using copies. In *Artificial Intelligence Research and Development. Frontiers in Artificial Intelligence and Applications* (2019), vol. 319, IOS press, Amsterdam, The Netherlands, pp. 125–134. doi: 10.3233/FAIA190115 [Online] Available from: http://ebooks.iospress.nl/volumearticle/52828

Other contributions that are out of the main scope of this thesis are listed below:

- Unceta, I., Nin, J., and Pujol, O. Transactional compatible representations for high value client identification: A financial case study. In *Complex Networks XI: Proceedings of the 11th Conference on Complex Networks CompleNet* (Exeter, UK, 2020), Springer, Berlin, Heidelberg, pp. 334–345

The following pages present a journey from the general to the particular. Throughout the document the reader is presented with different ideas developed at different levels of detail. Each part is designed to be self-contained. All parts begin with a short introduction and are then divided into the different chapters. Depending on his or her background, the reader may choose to go through the introductory sections at the beginning of each chapter or move forward directly to the content. The key findings are summarized at the end of each chapter. A reader interested in the practical applications of inheritance by copying to ensure machine learning accountability may profit from this summary to bypass Parts 1 and 2 and focus on Part 3 instead. Alternatively, someone more interested in the mathematics behind copying should more carefully study Part 2; specially, *Chapter 3* and *Chapter 4*. Finally, a reader who wishes to obtain a high-level understanding of this thesis, may choose to concentrate efforts on Part 1, where the notion of differential replication is first introduced. This document ends with a summary of our conclusions and an outline of future research.

Part I

Concept

"If during the long course of ages and under varying conditions of life, organic beings vary at all in the several parts of their organization, [...] I think it would be a most extraordinary fact if no variation ever had occurred useful to each being's own welfare, in the same way as so many variations have occurred useful to man. But if variations useful to any organic being do occur, assuredly individuals thus characterized will have the best chance of being preserved in the struggle for life; and from the strong principle of inheritance they will tend to produce offspring similarly characterized. This principle of preservation, I have called, for the sake of brevity, Natural Selection."

– Charles Darwin, *Origin of the Species*

The first part of this thesis presents a general framework for machine learning adaptation in rapidly changing environments. We introduce the idea of differential replication, which refers to the possibility of reusing the knowledge acquired by a machine learning model to train subsequent generations of models that replicate its decision behavior under new environmental constraints. This process of adaptation ensures model preservation, even in face of hard conditions, and allows for a more sustainable and efficient deployment of machine learning. In what follows, we formalize the problem of environmental adaptation and introduce differential replication as a possible solution. We describe some of its practical applications and provide an overview of the mechanisms available to build differential replicas of machine learning models. As we will later see, it is one of these mechanisms, copying, that we are most interested in.

Chapter 1

Differential replication in machine learning

1.1 Survival of the fittest

The instinct for self-preservation is a powerful primal force that governs the life of all living creatures. Natural Selection explores how organisms adapt to a changing environment in their struggle for survival [61]. In this context, conditions for survival are intrinsically defined by a complex, generally unknown fitness function. The closer organisms move towards the optimal value of this function, the better fit they are to face the hard conditions imposed by their environment and, hence, the better chance they have at survival. The level of adaptation to the environment therefore plays a key in ensuring preservation.

This predominant role of the environment is not unique to living organisms. It is also present in aspects of human society, from business to culture, including everything from economic changes, adjustment of moral and ethical concerns, regulatory revisions or the reframing of societal rules that results from unexpected global crises or natural catastrophes. In a smaller scale, it also affects machine learning model deployment. Indeed, the success or failure of a predictive model is largely influenced by its immediate surroundings. Not in vain did the Gartner Data Science Team Survey [118] find that over 60% of machine learning models designed and trained in companies during 2018 are never actually served into production, due mostly to a failure to meet the constraints imposed by their immediate environment. Hence, it seems reasonable to assume that understanding this environment is a necessary first step when devising any industrial machine learning solution.

A machine learning model's *environment* comprises all the elements that interact with the model

throughout its lifespan, including the data and their different sources, the deployment infrastructure, the governance protocol, or the regulatory framework. These elements may be both internal and external to a company. Internal elements refer to those, such as the feature engineering process or the deployment infrastructure, that are controlled by the data scientists and which are related, to a certain extent, to their strategic decisions. External elements, on the other hand, come from outside the company itself and are therefore generally out of its control. They refer, for example, to the trends of the market, the behavior of consumers, the relationship with third parties or any other aspect that may affect a machine learning based product or service. Both internal and external components impose requirements on how models are designed, trained and served into production. Moreover, these requirements are prone to change in time A machine learning model's environment can therefore be understood as a dynamic set of constraints that evolve throughout a model's lifespan. To survive in such an environment and ensure a sustained delivery over time, machine learning models need to adapt to new conditions.

This idea of adaptation has been present in the literature since the early times of machine learning, as practitioners have had to devise ways in which to adapt theoretical proposals to their everyday-life scenarios [14][248][134]. As the discipline has evolved, so have the techniques available to this end, giving rise to new areas of research. Consider, for example, situations where the underlying data distribution changes resulting in a concept drift. Traditional batch learners are incapable of adapting to such drifts. Instead, online learning algorithms were devised to iteratively update their knowledge according to such changes in the data distributions [35]. Another example is that of transfer learning. Studies on this field focus on cases where learning a given task can be improved through the transfer of knowledge acquired when learning a related task [182][231][255]. In addition, in cases where the change of task is accompanied by a change in domain, domain adaptation and domain generalization study how data labelled in a single [56] or multiple [142] source domains can be leveraged to learn a classifier on unseen data in another domain. In all these cases, a given machine learning solution needs to be adapted to a new domain or task. Yet, this adaptation doe snot require the definition of a new model hypothesis space. There are situations, however, where it is not the data distributions or the problem domain that change, but the environmental constraints; and, as a consequence, the feasible solution space. This is an altogether different problem that deserves further attention. Say that one of the original input attributes is no longer available, that a deployed black-box solution is required to be interpretable or that updated software licenses require moving our current machine learning system to a new production environment. These changes generally require the definition of a new model in a different hypothesis space. Say that a company wants to focus on a new client portfolio. This may require evolving from a binary classification setting to a multi-class configuration [79]. Another example is that where there is a change in the business needs. Commercial machine learning applications are designed to answer very specific business objectives that may evolve in time. Take, for example, fraud detection algorithms [103], which need to be regularly updated to incorporate new types of fraud that may not be feasible in the original scenario. In all these cases, structural changes to a model's environment introduce new operational constraints that cannot be met by the existing solution or a modified version. Instead, it might be necessary to move to a new

hypothesis space.

Here, we are concerned with such situations where a drastic change in the demands of a machine learning environment requires some form of adaptation. In what follows, we study and formalize this problem of environmental adaptation and discuss possible solutions. A straightforward approach is to discard the existing model and re-train another in a new space. A main drawback of this approach, however, is that in discarding the existing solution altogether, we also discard all the knowledge it acquired. We are therefore left to rebuild and validate the full machine learning stack from scratch. A process that is usually tiresome as well as costly. Hence, the re-training approach may not always be the most efficient nor the most effective way for tackling this challenge. Here, we discuss alternative solutions that imitate the way in which biological systems adapt to changes. In particular, we stress the importance of reusing the knowledge acquired by the already existing models in order to train a second generation that can better adapt to the new conditions. We review different strategies to this end and categorize them under the umbrella of differential replication. Finally, we present examples of real situations where the differential replication can be used to solve the problem of environmental adaptation in practice. We begin by providing an overview of how the problem of adaptation has been treated in the machine learning literature as of late.

1.2 Modelling adaptation to new environments

The most well known research branch for model adaptation is *transfer learning*. Transfer learning refers to scenarios where the knowledge acquired when solving one task is recycled to solve a different, yet related task [182]. In general, the problem of transfer learning can be mathematically framed as follows. Given source and target domains $\mathcal{D}_s$ and $\mathcal{D}_t$ and their corresponding tasks $\mathcal{T}_s$ and $\mathcal{T}_t$, such that $\mathcal{D}_s \neq \mathcal{D}_t$, the goal of transfer learning is to build a target conditional distribution $\mathsf{P}(y_t|x_t)$ in $\mathcal{D}_t$ for task $\mathcal{T}_t$ from the information obtained when learning $\mathcal{T}_s$ in $\mathcal{D}_s$. In general, the difference between $\mathcal{D}_s$ and $\mathcal{D}_t$ is given by a change in the data distribution, either in the marginal distribution of x and y or in the joint distribution of both. Observe that the change in any of those distributions directly affects the objective function of the optimization problem. This results in a change in the optimization landscape for the target problem. A graphical illustration of this problem is shown in Fig. 1.1(a), where the gray and red lines correspond to the source and target optimization objective level sets, respectively; and the shaded red area encloses the set of possible solutions for the defined hypothesis space. Transferring the knowledge from source to target requires moving from the original optimum in the source domain to a new optimum in the target domain. This process is done by exploiting the knowledge of the original solution, *i.e.* by transferring the knowledge between both domains. Advantages of this kind of learning when compared with the traditional scheme are that learning is performed much faster, requiring less data, and even achieving better accuracy results. Examples of methods addressing these issues are pre-training methods [78, 111] and warm-start conditioning methods[121, 127].

In transfer learning, it usually holds that $\mathcal{T}_s \neq \mathcal{T}_t$. There are cases, however, where the task remains the same for both the source and the target domains. This is the case of *domain adaptation* [56, 142]. Domain adaptation is a sub-field of transfer learning that studies cases where there is a change in the data distribution from the source to the target domain. In particular, it deals with learning knowledge representations for one domain such that they can be transferred to another related target domain. This changes of domain can be found, for example, in many of the scenarios resulting from the COVID-19 pandemic. In order to minimize interactions with the points of sale, several countries have decided to extend the limit of transactions where card payments are accepted without requiring cardholders to introduce their pin-code from 20 to 50 euros. Domain adaptation can be of use here to adapt card fraud detection algorithms to the new scenario.

Finally, another related branch that deals with model adaptation is that of *concept drift* [161]. In concept drift, it is the statistical properties of the target variable that change over time. In general, this happens in the presence of data streams [152]. Under these circumstances adaptive techniques are usually used to detect the drift and adjust the model to the new incoming data.

Here, we focus in an altogether different adaptation problem. In our described scenario, the task remains the same, $\mathcal{T}_s = \mathcal{T}_t$, but changes in the environmental conditions render the existing solution non-apt for the considered task. The new environmental conditions can be formally defined as a set of constraints, $\mathcal{C}$, that are added to the problem. As a result of these constraints the solution obtained for the source scenario can lay outside the new feasible set. The adaptation problem consists of finding a new compatible solution that lies within this set.

We can frame this problem using the former notation as follows. Given a domain $\mathcal{D}$, its corresponding task $\mathcal{T}$, and the set of original environmental constraints $\mathcal{C}_s$ that make the solution of this problem feasible, we assume a scenario where a hypothesis space $\mathcal{H}_s$ has already been defined. In this context, we want to learn a new solution for the same task and domain, but for a new target scenario defined by a new set of feasibility constraints $\mathcal{C}_t$, where $\mathcal{C}_t \neq \mathcal{C}_s$. In the most general case, solving this problem requires the definition of a new hypothesis space $\mathcal{H}_t$. In a concise form and considering an optimization framework this can be expressed as below, where *Scenario I* and *Scenario II* refer to source and target respectively.

$$
\begin{array}{lll}
\textit{Scenario I} & & \textit{Scenario II} \\
\text{for } \mathcal{T} \text{ in } \mathcal{D} & & \text{for } \mathcal{T} \text{ in } \mathcal{D} \\
 & \rightarrow & \\
\underset{\text{for } h \in \mathcal{H}_s}{\text{maximize}} \quad \mathsf{P}(y|x;h) & & \underset{for\ h \in \mathcal{H}_t}{\text{maximize}} \quad \mathsf{P}(y|x;h) \\
\text{subject to} \quad \mathcal{C}_s & & \text{subject to} \quad \mathcal{C}_t
\end{array}
$$

We refer to this problem as *environmental adaptation*. Under the above notation, the initial solution, the existing optimum, corresponds to a model h_s that belongs to the hypothesis space $\mathcal{H}_s$ defined for the first scenario. This is a model that fulfills the constraints $\mathcal{C}_s$ and maximizes $\mathsf{P}(y|x;h)$ for a training dataset $\mathscr{S} = \{(x, y)\}$, defined by task $\mathcal{T}$ on the domain $\mathcal{D}$. Adaptation involves transitioning from this scenario to *Scenario II*, a process which may be straightforward, although this is not always the case.

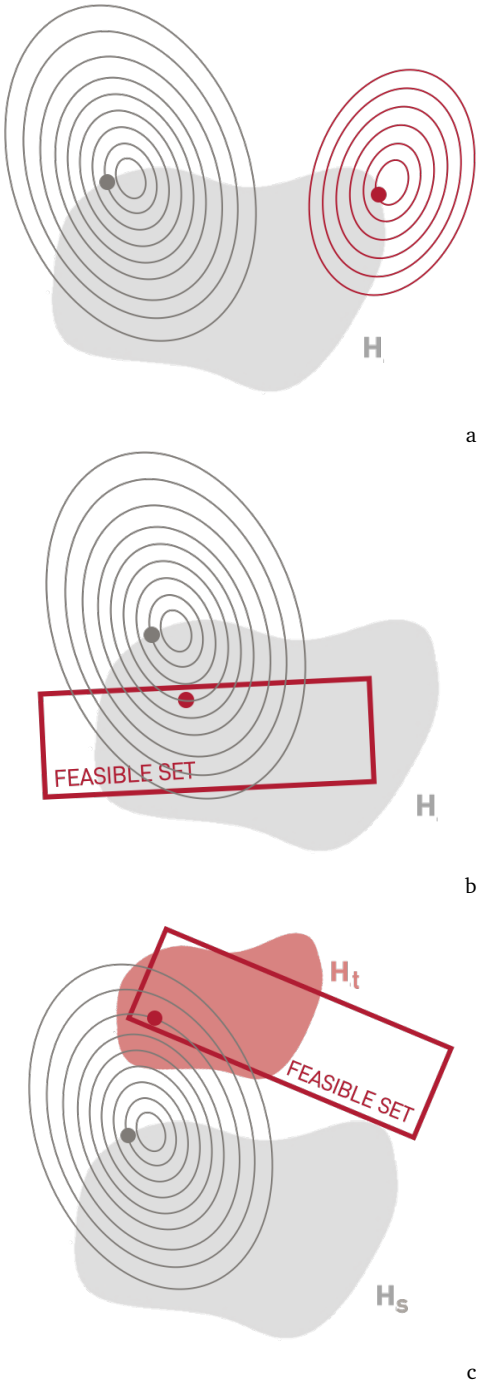


Fig. 1.1 The problems of (a) transfer learning and environmental adaptation for a case (b) where the new new feasible set overlaps with the existing hypothesis space and (c) where there is no such overlap. The gray and red lines and dots correspond to the set of possible solutions and the obtained optimum for the source and target domains, respectively. The shaded areas show the defined hypothesis spaces.

Take, for example, the two cases displayed in Fig. 1.1(b) and Fig. 1.1(c). In this figure, the optimization objective level sets defined by the domain and the task are displayed in gray, while the shaded area corresponds to the defined hypothesis space $\mathcal{H}_s$. The rectangles shown in red refer to the feasible set defined by new environmental constraints, $\mathcal{C}_t$. Observe that in both figures, the source solution (in gray) is not feasible for the target scenario. In Fig. 1.1(b) the new feasible set is compatible with the existing hypothesis space. Hence, environmental adaptation in this case may simply imply finding a new optimum in this space that complies with $\mathcal{C}_t$. In other cases the whole set of solutions defined by the source hypothesis space is unfeasible in the target scenario. This happens when there is no overlap between the feasible set defined by the target constraints and the set of models defined by $\mathcal{H}_s$. An example of this is shown in Fig. 1.1(c), where the constraints exclude the models in $\mathcal{H}_s$ from the set of possible solutions. In such cases, adaptation requires that we define an altogether new hypothesis space $\mathcal{H}_t$ that is compatible with the new environment and where we can find an optimal solution for the given domain and task.

Once again, note that this problem is different to that of transfer learning and domain adaptation. For both these settings, the solution in the source domain, while sub-optimal, is generally still feasible in the target domain. In environmental adaptation, however, the solution in the source scenario is often unfeasible in the target scenario. For illustration purposes consider the case of a multivariate Gaussian kernel support vector machine. Assume that due to changes in the existing regulation, this model is required to be fully interpretable in the considered application. The new set of constraints is not compatible with the source scenario and hence we would require a complete change of substrate, *i.e.* a new hypothesis space.

In what follows, we introduce the notion of *differential replication* of machine learning models as an efficient approach to ensuring environmental adaptation. Differential replication enables model survival in highly demanding environments, by building on the knowledge acquired by previously trained models in future generations. This effectively involves solving the optimization problem for *Scenario II* considering the solution obtained for *Scenario I*.

1.3 Differential replication

Under the theory of Natural Selection, environmental adaptation relies on changes in the phenotype of a species over several generations to guarantee its survival in time. This is sometimes referred to as *differential reproduction*. In the same lines, we define *differential replication* of a machine learning model as a cloning process in which traits are inherited from generation to generation of models, while at the same time adding variations that make descendants more suitable for the new environment. More formally, differential replication refers to the process of finding a solution h_t that fulfills the constraints $\mathcal{C}_t$, *i.e.* it is a feasible solution, while preserving/inheriting features from h_s. In general, $\mathsf{P}(y|x;h_t) \sim \mathsf{P}(y|x;h_s)$, so that in the best case scenario, we would like to preserve or improve the performance of the source solution

h_s, here referred to as the parent. However, this is a requirement that may not always be achieved. In a biological simile, requiring a guepard to be able to fly may imply loosing its ability to run fast.

Sometimes, differential replication can straightforwardly be applied by discarding the existing model and re-training a new one. However, it is worth considering the costs of this approach. In general, rebuilding a model from scratch (i) implies obtaining the clearance from the legal, business, ethical, and engineering departments, (ii) does not guarantee that a good or better solution of the objective function will be achieved[1], (iii) requires a whole new iteration of the machine learning pipeline, which is costly and time-consuming, (iv) assumes full access to the training dataset, which may no longer be available or require a very complex version control process. Many companies circumvent these issues by keeping machine learning solutions up-to-date using automated systems that continuously evaluate and retrain models, a technique known as continuous learning. Note, however, that this may take huge storage space, due to the need to save all the new incoming information. Hence, in the best case scenario, re-training is an expensive and difficult approach that assumes a certain level of knowledge that is not always guaranteed. In what follows we consider other approaches to implement *differential replication* in its attempt to solve the problem of *environmental adaptation*.

1.3.1 Differential replication mechanisms

The notion of *differential replication* is built on top of two concepts. First, there is some inheritance mechanism that is able to transfer key aspects from the previous generation to the next. That would account for the name of *replication*. Second, the next generation should display new features or traits not present in their parents. This corresponds to the idea of *differential*. These new traits should make the new generation more fit to the environment to enable *environmental adaptation* of the offspring.

Particularizing to machine learning models, implementing the concept of *differential* may involve a fundamental change in the substratum of the given model. This means we might need to define a new hypothesis space that fulfills the constraints of the new environment $\mathcal{C}_t$. Consider, for example, the case of a large ensemble of classifiers. In highly time demanding tasks, this model may be too slow to provide real time predictions when deployed into production. Differential replication enables moving from this architecture to a simpler, more efficient one, such as that of a shallow neural network [40]. This "child" network can inherit the decision behavior of its predecessor while at the same time being better adapted to the new environment. Conversely, *replication* requires that some behavioral aspect be inherited by the next generation. Usually, it is the model's decision behavior that is inherited, so that the next generation will replicate the parent decision boundary. Replication can be attained in many different ways. As shown in Fig. 1.2, depending on the amount of knowledge that is assumed about the existing model and its training data, mechanisms for inheritance can be grouped under different categories.

[1]The objective function in this scenario corresponds to $\mathsf{P}(y|x;h)$.

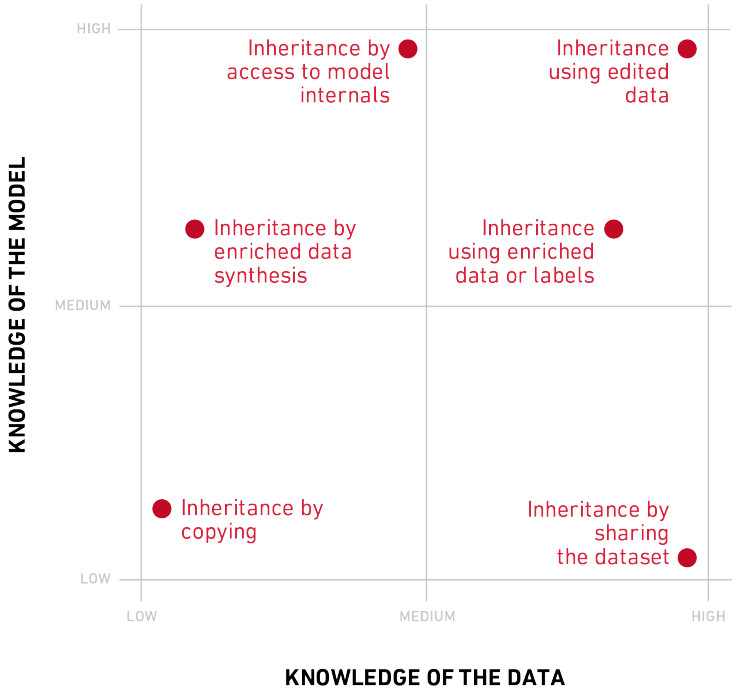


Fig. 1.2 Inheritance mechanisms in terms of their knowledge of the data and the model internals.

Inheritance by sharing the dataset Two models trained on the same data are bound to learn similar decision boundaries. This is the weakest form of inheritance possible, were no actual information is transferred from source to target. Instead, the decision behavior is reproduced indirectly and mediated through the data themselves. Re-training falls under this category [18]. This form of inheritance requires no access to the parent model, but assumes knowledge of its training data. In addition to re-training, model *wrappers* can also be used to envelope the existing solutions with an additional learnable layer that enables adaptation [170][171][2].

Inheritance using edited data Editing is the methodology that allows data selection for training purposes [25][24][176]. Editing can be used to preserve data that are relevant to the decision boundary learned by the parent solution and use them to train the next generation. Take, for example, the case where the source hypothesis space corresponds to the family of support vector machines. In training a differential replica, one could retain only those data points that were identified as support vectors [219]. This mechanism assumes full access to the model internals, as well as to the training data.

Inheritance using model driven enriched data Data enrichment adds new information to the training dataset through either the features or the labels. In this scenario, each sample in the training set is

[2]Note that model wrappers may require access to the model internals. In this study we classify them in this category by considering the most agnostic and general case.

augmented using information from the parent decision behavior. For example, a sample can be enriched by adding additional features using the prediction results of a set of classifiers. Alternatively, if instead of learning hard targets one considers using the output of the parent's class probability outputs or logits as soft-targets, this richer information can be exploited to build a new generation that is closer in behavior to the parent. Under this category fall methods like model distillation [40][112][225][258], as well as techniques such as label regularization [177][261] and label refinery [13]. In general, this form of inheritance requires access to the source model and is performed under the assumption of full knowledge of the training data.

Inheritance by enriched data synthesis A similar scenario is that where the training data are not accessible, but the model internals are open for inspection. In this situation, the use of synthetic datasets has been explored [40][262]. In some cases, intermediate information about the representations learned by the source model are also used as a training set for the next generation. This form of inheritance can be understood as a zero-shot distillation[179].

Inheritance of internals model's knowledge In some cases, it is possible to access the internal representations of the parent model, so that more explicit knowledge can be used to build the next generation [4][47]. For example, if both parent and child are neural networks, one can force the mid-layer representations to be shared among them [239]. Alternatively, one could use the second level rules of a decision tree to guide the next generation of rule-based decision models.

Inheritance by copying In certain environments access to the training samples or to the model internals may not be possible. In this context, experience can also be transmitted using synthetic data points labelled according to the hard predictions of the source model. This has been referred to as copying [236][240].

Note that on top of a certain level of knowledge about either the data or the model, or both, some of the techniques listed above often impose additional restrictions. Techniques such as distillation, for example, assume that the original model can be controlled by the data practitioner, i.e. internals of the model can be tuned to force specific representations of the given input throughout the adaptation process. In certain environments this may be possible, but generally it is not.

1.4 Differential replication in practice

In what follows we describe seven different scenarios where differential replication can be exploited to ensure a devised machine learning solution adapts to changes in its environment. In all seven of them we assume an initial model has already been trained and served into production. This model and its characteristics correspond to *Scenario 1*, as defined above. We describe how the constraints that apply to

Scenario 2 differ from the original scenario and discuss different techniques and approaches to adapting the existing solution to the new requirements. Note that, while specific specific examples are given here, other solutions based on differential replication may also be possible.

1.4.1 Moving to a different software environment

Model deployment is often costly in company environments [212][83][220][265]. Common issues include the inability to maintain the technological infrastructure up-to-date with latest software releases, conflicting versions or incompatible research and deployment environments. Indeed, in-company infrastructure is subject to continuous updates due to the rapid pace with which new software versions are released to the market. At any given time, changes in the organizational structure of a company may drive the engineering department to change course. Say, for example, that a company whose products were originally based on Google's Tensorflow package [1] makes the strategic decision of moving to Pytorch [187]. In doing so, they might decide to re-train all models from scratch in the new environment. This is a long and costly process that can result in a potential loss of performance. Especially if the original data are not available or the in-house data scientists are new to this framework. Alternatively, using differential replication the knowledge acquired by the existing solutions could be exploited in the form of hard or soft labels or as additional data attributes for the new generation.

Equivalently, consider the opposite case, where a company previously relying on other software now decides to train its neural network models using Tensorflow. Despite the library itself provides detailed instructions on how to serve models in production [97], this typically requires several third-party components for docker orchestration, such as Kubernetes or Elastic Container Service [260], which are seldom compatible with on-premise software infrastructure. Instead, exploiting the knowledge acquired by the neural network to train a child model in a less demanding environment may help bridge the gap between the data science and engineering departments.

1.4.2 Adding uncertainty to prediction outputs

In applications where machine learning models are used to aid in high-stakes decisions, producing accurate predictions may not always be enough. In those applications information about the risks or confidence associated with predictions may be required. This is the case, for example, of medical diagnosis [22]. Consider a case where an existing machine learning solution produces only hard predictions. In this situation, doctors and data practitioners have very little information on what the level of confidence is behind each output. Yet, a new protocol may require refraining from making predictions in cases of large uncertainty. To meet this new requirement, a new learnable algorithmic component can be added to wrap the original solution and endow it with a layer of uncertainty to measure the confidence in prediction [170][171][163].

1.4.3 Mitigating the bias learned by trained classifiers

Machine learning models tend to reproduce existing patterns of discrimination [17][105]. Some algorithms have been reported to be biased against people with protected characteristics like ethnicity [9][41][128][191], gender [33][42] or sexual orientation [100]. As a model is tested against new data throughout its lifespan, some of its learned biases may be made apparent [256]. Consider one of such scenarios, where a deployed model is found to be biased in terms of a sensitive attribute. Under such circumstances, one may wish to transit to a new model that inherits predictive performance but which avoids discriminatory outputs. A possible option is to edit the sensitive attributes to remove any bias, therefore reducing the disparate impact in the task $\mathcal{T}$, and then training a new model on the edited dataset [133][208]. Alternatively, in very specific scenarios where the sensitive information is not leaked through additional features, it is possible to build a copy by removing the protected data variables [235], as discussed in *Chapter 7* of this document. Or even, to redesign the hypothesis space considering a loss function that accounts for the fairness dimension when training subsequent generations.

1.4.4 Evolving from batch to online learning

In general, companies train and deploy batch learning models. However, these are very rapidly rendered obsolete by their inability to adapt to a change in the data distribution. When this happens, the most straightforward solution is to wait until there are enough samples of the new distribution and re-train the model. Yet, this approach is timely and often expensive. A faster solution to ensure adaptation to the new data distribution is to use the idea of differential replication to create a new enriched dataset able to detect the data drift. For example, including the soft targets and a timestamp attribute in the target domain, $\mathcal{D}_t$. One may then use this enriched dataset to train a new model that replicates the decision behavior of the existing classifier. To allow this new model to also learn from new incoming data samples we may additionally incorporate the online requirement in the constraints $\mathcal{C}_t$ for the differential replication process [234].

1.4.5 Preserving the privacy of deployed models

Developing good machine learning models requires abundant data. The more accessible the data, the more effective a model will be. In real applications, training machine learning models requires collecting a large volume of data from users, often including sensitive information. When models trained on user data are released and made accessible through specific APIs, there is a risk of leaking sensitive information. Differential replication can be used to avoid this issue by training another model, usually a simpler one, that replicates the learned decision behavior but which preserves the privacy of the original training

samples by not being directly linked to these data. The use of distillation techniques in the context of teacher-student networks, for example, has been reported to be successful in this task [44][254].

In order to minimize the risk of leaking personal data through models, the European General Data Protection Regulation [51] recognizes the principle of data minimization, which dictates that personal data shall be limited to what is necessary in relation to the purposes for which they are processed. However, it is often difficult to determine the minimum amount of data required. Differential replication has been shown to be successful in this task by producing a generalization of the model that reduces the amount of personal data needed to obtain accurate predictions [93].

1.4.6 Intelligible explanations of non-linear phenomena

A widely established technique in many industrial applications to ensure model remain explainable [74][147] is to use linear models, such as logistic regression. Model parameters, *i.e.* the linear coefficients associated to the different attributes, can then be used to provide explanations to different audiences. Although this approach works in simple scenarios where the variables do not need to be modified nor pre-processed, this is seldom the case for real life applications, where variables are usually redesigned before training and new more complex features are introduced. This is even worse when, in order to improve model performance, data scientists create a large set of new variables, such as bi-variate ratios or logarithm scaled variables, to capture non-linear relations between original attributes that linear models cannot handle during the training phase. This results in new variables being obfuscated and therefore often not intelligible for humans.

Recent papers have shown that the knowledge acquired by black-box solutions can be transferred to interpretable models such as trees [21][45][87], rules [102][198] and decision sets [137]. Hence, a possible solution to the problem above is to replace the whole predictive system, composed by both the pre-processing/feature engineering step and the machine learning model by a copy that considers both steps as a single black box model [233]. This option is further developed in *Chapter 6* of this document. Doing this, we are able to deobfuscate model variables by training copies to learn the decision outputs of trained models directly from the raw data attributes without any pre-processing. Another possible approach is using wrappers. This is, for example, the case of LIME [197], where a local interpretable *proxy* model is learned by perturbing the input in the neighborhood of a prediction and using the original solution as a query oracle.

1.4.7 Model standardization for auditing purposes

Auditing machine learning models is no easy task. When an auditor wants to audit several models under the same constraints all models need to fulfill an equivalent set of requirements. Those requirements may limit the use of certain software libraries, or of certain model architectures. Usually, even within the

same company, each model is designed and trained on its own basis. As research in machine learning grows, new models are continuously devised. However, this fast growth in available techniques hinders the possibility of having a deep understanding of the mechanisms underlying the different options and makes the assessment of some auditing dimensions a nearly impossible task.

In this scenario, differential replication can be used to establish a small set of canonical models into which all others can be translated. In this sense, a deep knowledge of these set of canonical models would be enough to conduct auditing tests. Say, for example, that we define the canonical model to be a deep learning architecture with a certain configuration. Any other model can be translated into this particular architecture using differential replication[3]. The auditing process need then only consider how to probe the canonical deep network to report impact assessment.

[3]Provided the capacity of the network is large enough to replicate the given decision boundary.

Lessons learned

- Machine learning models are often deployed to highly constrained environments where conditions are liable to change at any time. As a result, there is a pressing need to devise mechanisms that ensure adaptation of models throughout their life-cycle.

- *Environmental adaptation* is the process through which knowledge acquired by an existing model is reused from generation to generation to extend the useful life of machine learning models by adapting them to their changing environment.

- Such adaptation can be mediated by a projection operator able to translate the decision behavior of a machine learning model into a new hypothesis space with different characteristics. As a result, traits of a given classifier can be inherited by another, more suitable under the new premises. This is what we refer to as *differential replication*.

- There exist different inheritance mechanisms to achieve this goal, depending on specific knowledge availability scenarios, ranging from the more permissive *inheritance by sharing the dataset* to the more restrictive *inheritance by copying*, which is the solution requiring less knowledge about the parent model and training data.

- In this work we are primarily interested in studying the theoretical and practical mechanisms that allow inheritance by copying.

Part II

Theory

"For imitating is connatural to men from childhood and by it they differ from the other animals, because man is the most imitative animal and forms his first apprehensions through imitation. It is also connatural that they all enjoy imitations. A sign of this is what happens in practice: for we enjoy looking at the most accurate images of things which are themselves painful to look at"

– Aristotle, *Poetics*

The second part of this thesis is devoted to introducing the notion of copying from a theoretical perspective. For this purpose, in *Chapter 2* we provide a review of the existing literature on knowledge representation in general, and more particularly on how the knowledge stored in one form of representation can be transferred to another. Once having introduced this background, in *Chapter 3* we present the mathematical derivation for copying and its implementation through either the single-pass or the dual optimization approaches. Finally, in *Chapter 4* we discuss how copying can be validated in practice. We introduce a set of performance metrics and conduct extensive experiments on a series of well-known datasets to demonstrate the feasibility of the described strategies.

Chapter 2
Building the conceptual framework

2.1 An imitation game

Machine learning is an imitation game. Machine learning models are designed to imitate the behavior of the unknown function that governs a given data generation process. The use of machine learning is motivated by the need to transform one form of hypothesis representation, which is usually not accessible to us, to another, which we can control and which is therefore more suitable under certain circumstances.

The replacement of one model with another is a well-known practice in many scientific disciplines when the complexity of the phenomena of interest poses severe limitations to the extent to which they may be understood, studied or interacted with [98][125][216]. Surrogates or meta-models are of special relevance for simplifying systems or reducing the computational burden of simulations in many engineering tasks that involve expensive analysis codes, such as most optimization processes [193][217].

In the case of machine learning, this replacement is often twofold. We treat the data mechanisms as unknown and replace them with models that approximate their behavior [37]. As the complexity of these models increases, however, they become less accessible to our human mind. Hence, we find ourselves once again in need of replacing these complex structures with simpler representations that allow us to better comprehend their inner mechanisms. This need for comprehension, however, is generally the primary motivation for this replacement, there are also many other reasons why we might want to obtain yet more refined representations of a given process. These can be related, for example, to a need to increase utility by reducing the requirements for storage and computation [40, 112]; or more generally to a need

to adapt models to their changing environment by adding new features and characteristics [237], as previously discussed. In any event, independently of what the particular reason for this may be, we often wish to represent a model's acquired knowledge in a more suitable form.

Conceptually, we often identify the knowledge acquired by a trained model with its learned parameter values. Given a certain model architecture and its corresponding set of tunable parameters, we assume the knowledge learned by this model to be engrained into the specific parameter configuration that results from the learning process. This view assumes that knowledge is dependent of a particular instantiation, and therefore explicitly represented by its container. A more abstract approach, however, allows us to decouple content and container, by assuming that the knowledge is instead encoded in the learned relationship between the input and output dimensions. This idea is in line with studies on knowledge representation where they propose to solve the knowledge acquisition bottleneck using different methods that externalise and make explicit the tacit knowledge inside an expert's head [113]. In particular, we highlight a modelling technique known as "mimetism" [26], where an expert is queried to obtain answers on as many cases as possible and the resulting data is used to train a machine learning model which imitates the expert's behaviour.

This broader understanding of what knowledge accounts for in machine learning has allowed the development of large areas of research. Fig. 2.1 presents an overview of the scientific publications throughout the years in some of such areas. These include studies related to extracting rules from trained models or, more recently, distilling the knowledge of large cumbersome systems to train smaller models to perform complex pattern recognition tasks. Other approaches include those involving learners that actively participate in the learning process or those where malicious adversaries seek to acquire knowledge about a system to compromise it. The notion of copying also builds on this idea of knowledge to replicate the behavior of a given model using another.

2.2 Initial attempts at rule extraction

The idea of extracting symbolic representations from trained machine learning models dates back to the 1980s, when the first concept extraction algorithms were proposed and applied in practice [6][88][228]. By then, there was already an increasing awareness of the need to understand the representations learned by machine learning models. Es pecially in domains such as medical diagnosis where a start was made at using these systems. Already, it was not enough for models to be accurate. They also needed to be understood by their human users, if they were to trust them and deem them acceptable [72].

Most studies dated from this time proposed rule-extraction methods that operated on a search-based basis: a breadth-first search was conducted through a space of conjunctive rules to extract propositional *if-then* rules from trained neural networks. In cases where the resulting representations were comprehensible, they could be made accessible to human review. As a result, users of a given learning system were better capable of understanding its classification behavior. Ultimately, this enforced trust. A limitation of these

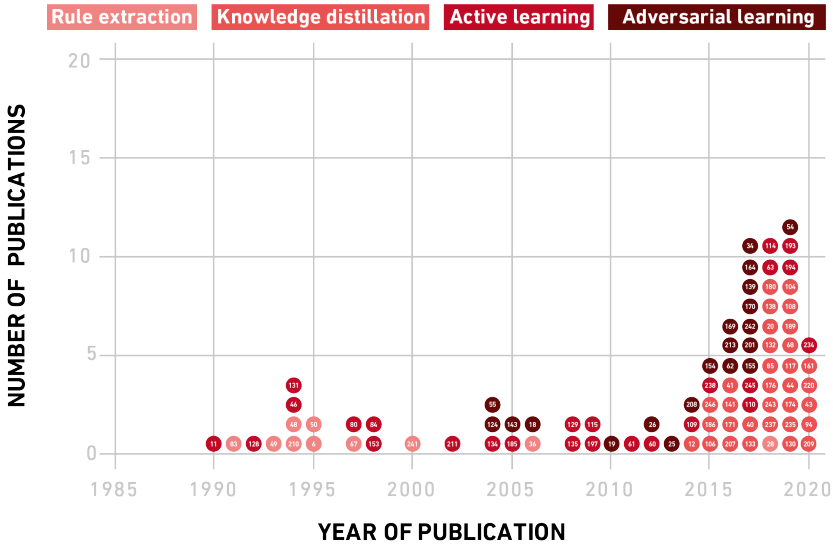


Fig. 2.1 Number of publications per year for each of the four disciplines: rule extraction, knowledge distillation, active learning and adversarial learning. Numbers correspond to paper references as shown in this document.

methods, however, was the computational complexity of the search, which increased exponentially with the number of input features.

Craven et al. presented in 1994 a novel approach to symbolic rule extraction to overcome this limitation. In their proposal, rule extraction was not framed as a search task, but, as a supervised learning task [52][53] instead. The target concept was the decision function learned by the network and the input features were the training attributes. Information about the target concept was acquired through oracle-based query algorithms, a learning technique that had been extensively described in the literature of the previous decade [7][107][242]. Two different oracles, *examples* and *subset*, produced new training samples and the target network itself was used to answer queries about the concept being learned. This procedure required less computation than search-based approaches and yet obtained rule sets with a comparable degree of fidelity.

These early ideas gave rise a few years later to TREPAN, a query algorithm to extract tree-structured representations of trained neural networks [54]. As before, TREPAN exploited queries to induce a decision tree that provided a close approximation to the function represented by the network. For this purpose, it amplified the dataset with new samples, generated by randomly selecting values from the marginal distribution of each attribute and labeling them according to the predictions of the net.

A particularity of TREPAN was that, contrary to most decision tree algorithms that followed a depth-first approach, it grew trees in a best-first manner. This ensured that each new split was defined in terms of the gain in the fidelity of the extracted tree to the considered network. Therefore maximizing this metric at each step. Moreover, because new data were available at each split, the tree-induction process did not degrade with depth. In general, the amount of training data available when growing a tree decreases with depth, so that the quality of a split depends heavily on the depth at which it is located.

Exploiting the neural net to label new data at each split ensured high quality splits also for the deeper levels. After a thorough literature review, we consider TREPAN to be the first clear predecessor of a copying algorithm.

While TREPAN focused primarily on extracting comprehensible outputs from neural networks, other authors explored concept extraction for different model architectures. In particular, many research from this time was devoted to approximating ensembles of models. It was around those years that techniques such as bagging [36], boosting [85], stacking [257] or error-correcting output coding [130] were first introduced. These approaches consisted on learning several different component classifiers by introducing variations on either the training data or the learners, and then combining all the individual predictions to obtain a final classification decision. These methods were very rapidly shown to improve the accuracy of individual algorithms in many domains. Despite being more accurate and robust, however, they hardly complied with the time and space requirements of many applications. Hence, on top of the comprehensibility issue, the appearance of ensembles introduced a need to deal with an increasing demand for storage and computation. A variety of papers from the late 1990s and early 2000s tackled the issue of compressing this ensembles into more compact models that nonetheless retained most of the predictive performance and stayed comprehensible.

Domingos [72] proposed CMM, which used a single base-learner to recover the decision behavior of a bagged ensemble of the same type of models. This was done by presenting the base-learner with a new training set, composed of the original training examples plus a large number of data points artificially created and labelled using the bagged model. CMM used decision rule sets as the base learner. New training examples were generated by randomly sampling the hyperspace classified by the decision rules. Another method aimed at approximating the behavior of a given ensemble was proposed by Zeng et al. [262], who used pseudo training sets based on the distribution of the original data to compress complex ensemble classifiers into multilayer neural nets that maintained a similar accuracy. Given the complexity of the hypothesis formed by ensemble classifiers, the authors enriched the original training set with additional data. These data were generated by sampling the marginal distribution of each individual attribute and then labelling each sample using the the ensemble as an oracle.

These initial efforts for ensemble compression crystallized in 2006 in a famous paper by Buciluǎ et al. [40]. This work explored different methods for generating pseudo training data to transfer the knowledge acquired by a large, complex ensemble of models to a faster, more compact neural network architecture.

2.3 The notion of Knowledge Distillation

Most commercial machine learning applications deal with non-trivial problems, such as speech recognition or computer vision. During training, machine learning models need to extract structure from very large, generally redundant datasets. A task that usually involves using high capacity models, such as ensembles

or deep neural networks. This incurs high costs for companies, both in terms of the required computational resources and the training time, and bears the question of whether simpler models could be deployed instead. In fact, empirical work has shown that shallower models can approximate decision boundaries of arbitrary complexity [58]. However, they tend to be more difficult to train than deeper ones, which have the added benefit of ensuring a generally improved predictive performance on raw data [63][77].

The training stage of any model takes up a great deal of computational and time resources. In contrast, models are required to operate in almost real time during deployment, a stage when there exist much more stringent time limitations. Predictions are expected to be fast and efficient. Deployed models therefore have to provide answers in a matter of seconds. And to do so accurately. When it comes to space, several constraints also apply. While more powerful infrastructures are available at the training stage, during deployment machine learning models are often served to limited memory devices, such as mobile phones or tablets. Hence, additional restrictions appear in terms of the required memory allocation, the available computational power or the acceptable running time, among others. This gap between the training and deployment stages has been the subject of a large amount of research. The general belief is that there exists a complex trade-off between the representational capacity of a model to extract structure from the data, its complexity, and its requirements for latency and computational resources that can cause severe bottlenecks at the time of prediction. Overcoming such deployment barriers has been the focus of extensive research during the past 10 to 15 years.

The seminal work by Buciluǎ et al [40] showed in 2006 that it is possible to compress the knowledge acquired by a complex ensemble of models into a single smaller model, better suited to operate under the demands of a stringent deployment environment. This paper pioneered a vast field of research that later developed under the more general framework of model distillation [112]. Authors of this paper showed that it is possible to decouple the training and deployment stages by using different models for each task. During training, a larger, more complex architecture, such as that of an ensemble, can be used to recognize the patterns in the data. This architecture can then be exploited to guide a smaller model to an equivalent solution and use this simpler system at test time. This proposal was evaluated on eight binary classification problems and it was shown that the loss in performance due to compression was generally negligible. The more compact models provided results that were almost as accurate as those of the larger ensembles, while at the same taking 1000 times less space and being 1000 times faster.

These initial findings were borne out by a follow-up paper in 2014 [12]. Authors showed that shallow neural networks could be trained to perform similarly to more intricate models on the CIFAR-10 image recognition and TIMIT phoneme recognition tasks. These results were obtained by guiding a simple model, referred to as the student, to approximate the function learned by a deeper model, the teacher. To do so, the student model was not trained on the original labels. Instead, it was passed the data labeled according to the scores produced by the teacher. Assuming that the teacher was a neural net, these scores were matched to the logits z_i, *i.e.* the inputs to the softmax output layer for each data point i. The best results where obtained when expressing the learning objective function for the student as a

regression problem of the form

$$\ell = \frac{1}{2N} \sum_{i=1}^{N} ||v_i(\boldsymbol{x}_i; \boldsymbol{\omega}, \boldsymbol{\beta}) - z_i||_2^2 \tag{2.1}$$

for training data $\mathscr{D} = \{(\boldsymbol{x}_i, z_i)\}_{i=1}^{N}$, N the total number of data points, $v_i(\boldsymbol{x}_i; \boldsymbol{\omega}, \boldsymbol{\beta})$ the logits of the student model and $\boldsymbol{\omega}$ and $\boldsymbol{\beta}$ its weight and offset matrices. In the simplest case, the optimal values for $\boldsymbol{\omega}$ and $\boldsymbol{\beta}$ were obtained by back-propagating the error throughout all the layers of the student and updating each value using stochastic gradient descent. Results showed that shallow student nets were able to exploit the knowledge conveyed by the teacher signals to achieve performances that were previously only achievable by deeper models; thus effectively compressing this knowledge.

The choice of logits as labels was motivated by the belief that using the logits allowed training of the student to be conducted with greater ease. In a regular neural net architecture, the final softmax layer transforms the logits z_i into class probabilities p_i, by comparing each z_i with the other logits as

$$p_i = \frac{e^{z_i/T}}{\sum_k e^{z_k/T}} \tag{2.2}$$

for T a temperature value that governs the distribution of the resulting probabilities over classes. This transformation smoothens the distribution of labels, so that information about the internal configuration of the teacher is lost when passing through logits to probability space. Using the logits as training targets ensured that this information could be exploited by the student to better learn the behavior of the teacher. A year later, Hinton et al. [112] demonstrated that this approach to model compression was actually a special case of their more general solution: *knowledge distillation*.

In distillation, class probabilities produced by the teacher are used as "soft targets". These soft targets are obtained by setting the temperature T in (2.2) to a high value. This temperature is such that when $T \to \infty$ all classes share the same probability, whereas when $T \to 0$ the soft targets collapse into hard labels. Hence, by choosing a large value for T, Hinton and his collaborators ensured a softer probability distribution over the different classes to help the student generalize in the same way as the teacher. In this setting, the cross-entropy distilled loss can be expressed as follows

$$\mathcal{C}_D = -\sum_i p_i \log(q_i) \tag{2.3}$$

for p_i the class probabilities output by the teacher and q_i those predicted by the student. Each data point i then contributes a cross-entropy gradient given by

$$
\begin{aligned}
\frac{\partial \mathcal{C}_D}{\partial v_i} &= -\sum_k p_k \frac{\partial \log(q_k)}{\partial v_i} \\
&= -\frac{1}{T}\sum_k \frac{p_k}{q_k} q_k(\delta_{ki} - q_i) \\
&= \frac{1}{T}(q_i - p_i) \\
&= \frac{1}{T}\left(\frac{e^{v_i/T}}{\sum_k e^{v_k/T}} - \frac{e^{z_i/T}}{\sum_k e^{z_k/T}}\right)
\end{aligned}
$$

For high values of T the exponent tends to zero, since the temperature term dominates over the logits. The expression above can therefore be approximated by the truncated Taylor expansion as

$$
\frac{\partial \mathcal{C}_D}{\partial v_i} \approx \frac{1}{T}\left(\frac{1 + v_i/T}{N + \sum_k v_k/T} - \frac{1 + z_i/T}{N + \sum_k z_k/T}\right) \tag{2.4}
$$

If we assume the transfer training data to be normalized, then the logits are separately zero-meaned, so that $\sum_k v_k = \sum_k z_k = 0$. As a result, the contribution of each value v_i can be written as

$$
\begin{aligned}
\frac{\partial \mathcal{C}_D}{\partial v_i} &\approx \frac{1}{T}\left(\frac{1 + v_i/T}{N} - \frac{1 + z_i/T}{N}\right) &(2.5)\\
&\approx \frac{1}{T}\left(\frac{1 + v_i/T - 1 - z_i/T}{N}\right) &(2.6)\\
&\approx \frac{1}{NT^2}\left(v_i - z_i\right) &(2.7)
\end{aligned}
$$

In the high temperature, regime distillation can be assimilated to minimizing $1/2\left(v_i - z_i\right)^2$, as seen for model compression in (2.1). In this regime, the teacher can convey useful information to the student through very positive logit values. By tuning the temperature parameter, distillation can also function in other regimes. In the case of lower values of this term, for example, large negative logits, which tend to be noisy, are ignored. In the intermediate temperature regime, some of the knowledge captured by the relative probabilities of incorrect answers is kept to train the student. The optimal regime is that were a balanced trade-off is obtained between the different effects. In general, this regime is found empirically.

Hinton and his collaborators observed that, in the optimal regime, distillation significantly improved results on datasets such as MNIST, by exploiting teacher feedback to guide simple models to solutions that were not accessible when learning directly from the raw training data. This was a remarkable observation, since it showed that it is often easier to train a simple model on the real-valued scores output by a pre-trained complex model than it is by using the actual ground-truth labels as targets.

In general, we can define the student loss as the cross-entropy between the class probability outputs of the student and the ground truth labels for each data point in the set $\mathscr{D}$. This loss can be expressed

as follows

$$\mathcal{C}_S = \sum_{i=1}^{N} t_i \log(p_i) \tag{2.8}$$

for t_i the true labels corresponding to each instance x_i and p_i the student model trained on the teacher-labelled set $\mathscr{D}$. Hence, the overall loss of knowledge distillation can be expressed as the joint of the distilled loss and the student loss

$$\mathcal{C} = \alpha \mathcal{C}_D + \beta \mathcal{C}_S \tag{2.9}$$

for regularization parameters α and β. The combined architecture of such a problem is shown in Fig. 2.2.

As mentioned before, a major result of distillation is that when comparing the student loss as defined in (2.8) to that obtained for a model belonging to the same class but trained directly on the ground truth labels, the former tends to be lower. This is, a student trained on teacher signals can improve the baseline performance for that model class. In light of this finding, distillation has received increasing attention from the machine learning community in recent years. Papers in this field have explored different forms of supervision from the teacher [225], training the same network in generations [90] or inducing teacher signals with a softened label distribution to convey useful task-dependent information to students [258]. Distillation has been found to work well across a wide range of applications too, including mutual learning [264], distributed learning [190], learning from noisy labels [144] or training stabilization [200]. In a few cases, it has also been extended to other tasks, such as defending from adversarial attacks [185], data augmentation [140] or data privacy [44][254].

In spite of its success, however, there is still a very limited understanding of the theoretical and empirical foundations behind knowledge distillation. Early works attributed this success to the encoding of "dark knowledge" [112]. This dark knowledge was assumed to be encoded in the class probabilities assigned by the teacher to the wrongly classified samples. Other authors have also pointed in this same direction, by stating that soft targets from the teacher may be more informative to the student than the original hard labels [12]. Partly because the teacher acts as a filter for complexity, by eliminating some of the errors in the original label set or by smoothing the corresponding distribution. In this regard, rich information outputs by the teacher can be understood as a form of sample weighting that makes learning easier for the student [90][227]. Hence, mechanisms related to knowledge distillation can be seen as a form of regularization that reduces the generalization gap between teacher and student by preventing overfitting of the latter. Yet, there is still little consensus as to why these mechanisms work [99].

Lopez-Paz et al. [153] related distillation to a form of learning using privileged information, a setting where an intelligent teacher provides additional per-instance information to support the learning process [244]. Unification of these two techniques, however, is limited to cases where the teacher's supervision is noise-free. Moreover, this parallelism does not suffice to answer the question of why this supervision works in the first place. A more recent contribution by Phuong and Lampert has provided

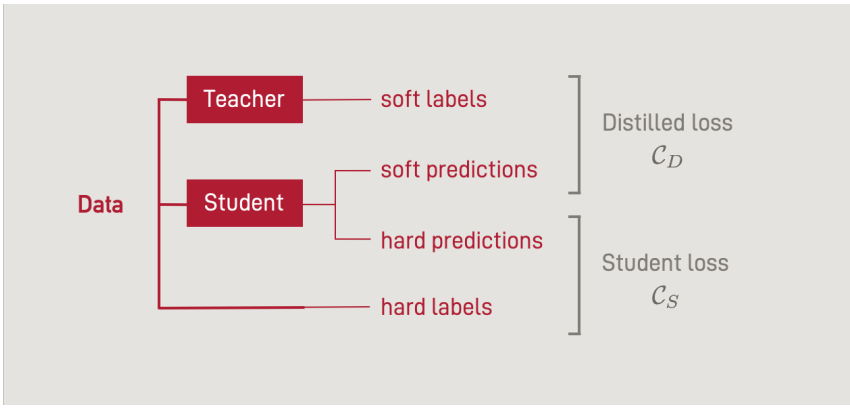


Fig. 2.2 Diagram for knowledge distillation.

more insights into the inner mechanisms of knowledge distillation in the scenario of linear and deep linear binary classifiers [188]. Notably, they proof a generalization bound for fast convergence of student learning and reveal three key factors that determine the success of distillation: (1) data geometry, and particularly the angular alignment between the data distribution and the teacher signals, (2) optimization bias, specified as the specific generalization properties of the different learning algorithms, and (3) strong monotonicity of the student classifier, through which an increased size of the student training set leads to a better approximation of the teacher's knowledge. Further, Cheng et al. [47] have proposed interpreting distillation by studying the visual concepts encoded in the intermediate layers of deep neural networks. Their results suggest that distillation enables students to learn more visual concepts than when learning from raw data and that this learning can be done simultaneously. In line with previous findings, authors of this paper also report that guidance from a teacher ensures a more stable optimization of the student concept representation.

In absence of a deeper understanding of the mechanisms underlying knowledge distillation, most authors have studied the impact of specific student and teacher architectures. During the last year, several articles have been aimed at identifying the characteristics that make a good teacher [48][173]. Results show that a higher accuracy of the teacher does not necessarily ensure a better performance for the student. Instead, it has been demonstrated that teachers trained to model the true class probability distribution can significantly aid learning [173]. Conversely, using larger models as teachers may result in an increased capacity mismatch between teacher and student [48]. In those cases where this mismatch is substantial it may prevent the student from converging to the desired solution [124]. Proposals to solve this problem include early-stopping during teacher training [73] or performing knowledge distillation sequentially in several steps. Mobahi et al [175] circumvent this issue by using student and teacher models belonging to the same class. In some cases, a very large capacity gap may prevent the student from reaching a reasonable performance altogether. It has been observed that in problems related to feature distillation, student models reach their parametric modelling limit before being able to converge to the appropriate solution [203]. Sometimes performing even worse than baseline. These results suggest

that certain knowledge distillation strategies may only succeed for specific architectures and training settings, being non-generalizable to the average learning task.

2.4 Generating pseudo training data

An important degree of freedom in distillation is the so-called transfer set used to train the student. Traditionally, knowledge transfer has been treated as a standard learning process, where the training data are relabelled and extended to learn an alternative model [32]. When it comes to the relabelling step, different methods have been proposed. See, for example, the work by Li et al. where dynamic importance sampling is used to relabel the original training data according to classes sampled from a proposal distribution [143]. This method avoids computation of the full matrix multiplication at the teacher softmax layer, which is generally costly. Instead, it approximates the teacher loss function using mixture of Laplace distributions. This leads to an important reduction in the required computational resources to train competitive students.

In general, the original training set is relabelled for distillation, either in its raw form [45][112] or enriched with additional synthetic data [21][149]. In [149], GANs are used to approximate the training data and enrich the transfer set with additional data points. Alternatively, Bastani et al. propose an algorithm that actively samples new points to avoid overfitting when approximating a neural network with a decision tree [21]. There are also cases where teachers and students with the same task have different access to training data. See, for example, the case of RDPD settings, where rich multimodal training data are leveraged to transfer the knowledge from the teacher to a student operating on poor data [114]. As a result, the student mimics not only the performance but also the behaviour of the teacher.

Conversely, some authors advocate for the use of unlabelled data, extracted from the estimated density of the attributes [40][262]. The main reason for using unlabelled sets of synthetic data is that, whereas the size of the given training set may be small, it is possible to generate as many synthetic samples as needed. When the teacher is a complex ensemble of classifiers, for example, a large transfer set may be required for approximating its decision boundary with a simpler model. Hence, in many occasions, students are trained on large amounts of data. Indeed, in cases where the student has access to a large pool of unlabelled samples, distillation can be understood as a means of semi-supervised learning [194]. However, generating unlabelled data is a non-trivial task and generally requires access to the training data distribution.

In [40] three different methods are proposed to this end: RANDOM draws independent samples from the marginal distribution for each attribute, while NBE and MUNGE sample from an estimate of the joint attribute density. While RANDOM generates new samples in a task-agnostic manner, both NBE and MUNGE exploit those areas of the space where the original training data are located. This ensures a certain coherence in the resulting transfer set and forces the student to focus on the defined regions of

interest. As a result, performance is significantly improved.

A prior work by Zeng and Martinez used pseudo training sets labeled using class probability vectors output by a large ensemble of classifiers [262], where each vector component was defined in terms of the number of votes received from the ensemble for each distinct class label. Points in the pseudo training set were obtained by sampling the marginal distribution of each individual attribute. For nominal features, this distribution was directly computed from the original training data. In the case of continuous features, values were first discretized into equally-sized intervals and then the marginal contribution of each separate interval was computed. Results on 16 datasets demonstrated success of this approach to approximate bagging classifiers.

A recent contribution by Heo et al. trains student classifiers on adversarial samples supporting the decision boundary of the teacher [109]. Authors of this paper assume that a teacher's knowledge is embodied in its learned decision boundary. Hence, an adversarial attack is exploited to discover samples supporting this boundary and then transfer this information to the student. The resulting classifiers achieve state-of-the-art performance leveraging solely the information conveyed by these samples. As described in later sections other approaches using a adversarial attacks have also been successful in retrieving relevant information form trained machine learning classifiers.

Obtaining a reduced set of highly informative samples is expensive as well as complex. Over the years, different disciplines have evolved in relation to this issue. See, for example, works on machine teaching, where a human teacher hand-picks as small a training set as possible to train a machine learning system [267]. Or, alternatively, the numerous contributions to the field of active learning, where a desired hypothesis is learned by reducing the number of queries to a human oracle [214].

2.5 Sample selection in Active learning

Active learning focuses on developing learning strategies in settings where unlabeled data are abundant, but there is a high cost associated with labelling. In such cases, a human annotator is used as an oracle to which a learner may pose queries. Queries in this context are assumed to be costly and, hence, the learner usually aims at achieving the maximum accuracy using as few labeled instances as possible. Besides, as opposed to the regular machine learning environment, where the learner remains passive throughout the learning process, in active learning the learner has some control over the inputs on which it trains [50]. Hence the name active learner. Active learning strategies focus on query optimization to minimize the cost of sample annotation, while at the same time ensuring the active learner achieves a good performance in the considered task.

There exist many different scenarios where an active learner may pose queries to an oracle. The three most common settings are those of membership query synthesis, stream-based selective sampling and pool-based sampling, as shown in Fig. 2.3. In all three of them, there exists an initial set of labeled data, which is known and which is used as guidance during learning. The case of pool-based sampling [141], also

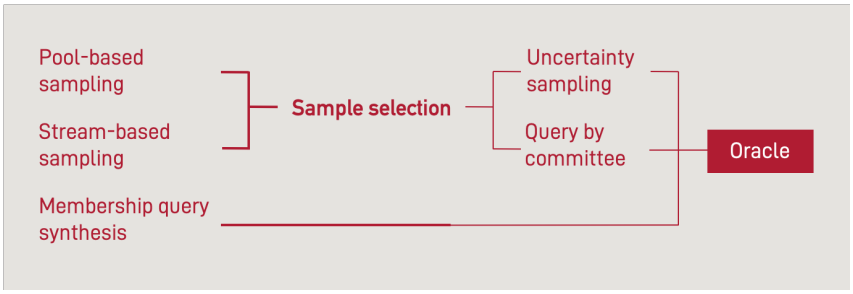


Fig. 2.3 Diagram of the different sampling and sample selection techniques in active learning.

known as batch-mode active learning, is perhaps the most well-known scenario. Here, there exists a small set of labelled data and a large pool of unlabeled data, a situation that is common to many real-world problem domains such as text [141][167] and image classification [66][122] or speech recognition [199]. Queries in this setting are selectively drawn from the pool by evaluating their informativeness and added to the training set. Informativeness is quantified based on a user-selected metric that depends on the considered application. Stream-based settings work similarly to pool-based scenarios, but here the learner samples new instances from a stream of data rather than from a data pool.

In membership query synthesis the learner generates new samples sequentially, instead of selecting them from an existing collection. For each new query, it observes the class label predicted by the oracle. This approach decreases the amount of time required for learning when compared to pool-based methods [146]. In membership query synthesis the learner queries the oracle using samples generated following a given probability distribution [7]. At each step, it constructs a new instance and queries the oracle to obtain a label. The labelled instance is then incorporated to the training set for subsequent steps. A main drawback of this approach is that arbitrary queries are often hard to understand for humans, as demonstrated by Lang and Baum [138]. Especially in highly structured environments, such as those of image or text. Labelling in this context is hard, even when using more advances methods such as GANs [94] to generate new instances [116][266]. It has been only recently that advances have been reported in this regard [211].

In all three scenarios above, a measure of the informativeness is used to select the most appropriate sample at each step. When using membership queries, this notion of informativeness is embedded into the probability distribution defined during sampling. This distribution defines how the resulting samples are distributed throughout the input space and therefore governs the intensity with which the different regions are explored. In the cases of stream- and pool-based sampling, new samples are individually chosen each time from an already available set. When using stream data, the learner evaluates samples sequentially to decide whether to query or to reject them. In pool-based active learning the whole collection is evaluated before selecting the most suitable sample for querying. Hence, regardless of the particular setting, active learning tasks require some form of sample selection strategy [11][50].

Most existing sample selection strategies fall under one of the two categories: uncertainty sampling [141] and query by committee [86]. In uncertainty sampling, the learner selects the examples

to be labelled from those for which the predicted label is most uncertain. Based on this information, specific regions of interest are defined from which the new samples are drawn or selected for querying. Many strategies for uncertainty sampling use entropy as an uncertainty measure [259]. Related to this approach are also additional strategies for margin sampling [210]. See, for example, the work by Tong et al, who experiment with an uncertainty sampling strategy for support vector machines that involves querying the instance closest to the linear decision boundary [229].

In general, sample selection through uncertainty sampling relies on the class probability estimations output by the learner. However, many learning algorithms produce classifiers lacking such class probability outputs. To overcome this issue, the query by committee approach suggests consulting several classifiers, instead of using a single learner. Information from the different classifiers is used to evaluate the uncertainty of each prediction [50][86] and select those samples for which the disagreement is largest. This idea has also been extended to settings where learning is conducted using single [139] or multiple teachers [65], so that at each time step the learner can choose which teacher to query. Training specialized teachers, each focusing on different aspects of the given task, ensures that the most optimal samples are chosen each time. An additional approach that circumvents the issue of needing class probability outputs in committee-based settings uses nearest-neighbor classifiers [89][145], and is often referred to as memory-based or instance-based learning. Here, each neighbor is allowed to vote on the class label of a given data point. The proportion of votes for each label is then used to represent the posterior label probability for each sample.

Finally, a modern approach to active learning proposes to select unlabeled samples at each iteration based on their overall representativeness [68]. As opposed to informativeness, which measures the uncertainty reduction that results from incorporating a given sample to the training set, representativeness is a measure of how well a sample represents the overall structure in the data. Using representative samples avoids redundancy by reducing the size of the training set, and can highly decrease learning time. In recent years, several contributions have proposed to combine measures of informativeness and representativeness to conduct sample selection in active learning settings [115][120].

While the final objective of knowledge distillation and active learning differ from one another, learnings from one discipline are often relevant to the other. Particularly when it comes to generating a suitable training set to transfer the knowledge from teacher to student. While this transfer is absent in the active learning setting, strategies for query selection in this scenario can be applied to guide the student to a desired solution [253]; as well as to reduce the time required for this process by optimizing training data generation. A third discipline which also bears similarities to these two is that of adversarial learning, where a malicious adversary exploits knowledge of a model to compromise it.

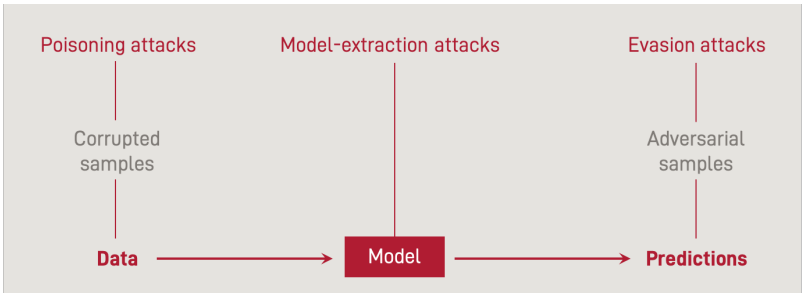


Fig. 2.4 Different types of adversarial threads.

2.6 An overview of Adversarial learning

As the presence of machine learning models increases, so does the incentive for defeating them. Today, adversaries actively manipulate the data to avoid detection across a wide range of domains. This is the case, for example, of email spam filtering and detection, where adversaries insert non-spam words into corrupt emails or break up spam words using spurious punctuation [59]. Failure to detect these attacks can have severe economic consequences for users unaware of the threat posed by spam. Other domains that are prone to attacks are those of fraud detection, where perpetrators employ increasingly sophisticated strategies to circumvent the defenses put in place [132], web search, where adversaries manipulate pages to improve placement, or counter-terrorism through image and video surveillance, where wrongdoers conceal their identity by fooling face recognition software. In all cases, the need of renewal is never ending, since classifiers very rapidly become obsolete as adversaries learn to defeat them.

The field of adversarial learning studies such situations where an external attacker with access to a system seeks to compromise its security. The objective of any adversarial learning strategy is to gain a better understanding of potential attacks to adapt to adversaries' evolving manipulations, in hope of producing more robust predictive systems [60][155]. In general, external attackers have some short of access to the target model, either through a query interface or trough direct manipulation, and use this access to gain information about its behavior. They then exploit this information to take advantage of the vulnerabilities surrounding the decision boundary to craft targeted attack vectors.

The first credited contribution to the field of adversarial learning appeared in 2004 by Dalvi et al. [60]. This paper suggested possible countermeasures in spam detection scenarios where the attacker had full knowledge of the system. Results clearly outperformed previous baselines and led to further research into this issue. A follow-up paper by Lowd and Meek demonstrated in 2005 the feasibility of adversarial attacks even in cases of partial knowledge [155]. They introduced ACRE, an adversarial classifier reverse engineering strategy to determine whether an adversary could learn sufficient information about a system to launch successful attacks against it and design defense protocols accordingly. These findings formed the basis of research on adversarial learning. More recently, the discovery of *adversarial examples* against deep networks in 2014 brought new attention to the field [226]. In their ground-breaking paper Szegedy

et al. showed that deep neural networks can be easily fooled using samples with minimal perturbations that are non-discernible to the human observer. On the basis of these initial results, much research has been conducted in the past decade with a view to understanding the vulnerabilities of classifiers in face of different forms of adversarial attacks [20]. A diagram with the different forms of attacks is shown in Fig. 2.4. Studies have mainly focused in two thread types: evasion and poisoning attacks [19].

Evasion attacks consist of manipulating data at test time to produce false negatives. Depending on the assumptions made on the adversary's knowledge of the system, different strategies are possible. Those based on gradient descent, for example, have been demonstrated to effectively mislead classification of different model architectures, including linear classifiers, support vector machines and neural networks, in malware detection tasks [27]. Similar techniques have also been successful in fooling systems in other application domains, such as image classification [67][169]. Poisoning attacks are also aimed at increasing the number of misclassified samples at test time, but contrary to evasion attacks, they occur at training time. During this phase, an adversary with access to the input data injects one or more corrupt instances into the training set to decrease a model's classification accuracy. Biggio et al. showed that gradient ascend procedures could significantly increase test error in support vector machines [28] in this manner. More recently, attacks of this form have been formulated as bilevel optimization problems, where the outer optimization maximizes the attacker's objective, while the inner optimization amounts to learning the classifier on the poisoned training data [168].

In addition to fooling them, attackers may also seek to steal the models. Model extraction attacks can be particularly harmful in cases where a model is protected by industrial secrecy or when it is deemed confidential due to the sensitive nature of its training data. Tramèr et al. demonstrated the efficiency of different extraction attacks against commercial services such as BigML and AWS [232]. They showed that a qualified adversary could steal models by interacting with their prediction APIs through highly informative queries. Equivalently, Wang and Gong have reported that a similar procedure can enable adversaries to steal model hyperparameters [253], which are critical to business revenue. Moreover, extraction attacks have succeeded in retrieving subsets of a model's training data. Even for neural networks [218]. This is possible because learning in high capacity models largely relies on data memorization [263].

Initially, it was assumed that attackers had full access to the targeted system. They knew of the underlying learning algorithm and could draw samples from the training data distribution. Today, different attack scenarios are defined in terms of the adversary's level of knowledge [27], as shown in Fig. 2.5. Perfect-knowledge attacks, also referred to as white-box attacks, correspond to the classical approach, where the attacker is assumed to know everything. In contrast, in limited-knowledge attacks some level of restriction is assumed on the knowledge of the adversary. In general, attacks of this form can be classified into two different types. In gray-box attacks, attackers may know the learning algorithm and have a certain intuition about the problem's feature representation. Finally, black-box attacks correspond to the most restrictive scenario, where the attacker can interact with the model through a query interface, but lacks any substantial knowledge of the underlying data and parameter distribution.

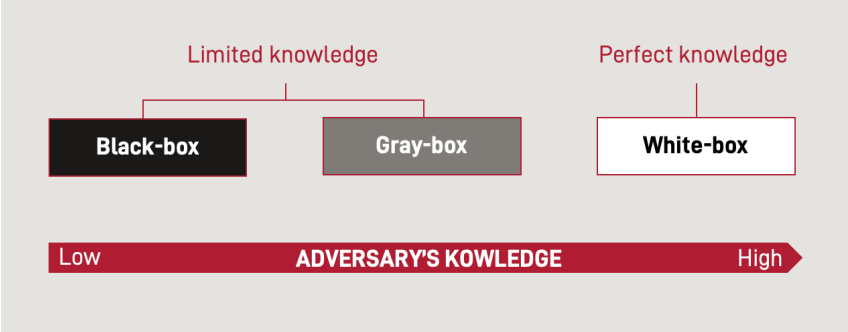


Fig. 2.5 Different forms of adversarial learning in terms of the adversary's level of knowledge.

In recent years, a commonly used approach to overcome the knowledge gap in limited-knowledge attacks has been using surrogate models. Adversarial examples are first crafted for the surrogate classifier and then transferred to the original system. This approach is often referred to as *transferability-based adversarial learning* [183]. The main idea is, even in cases were the adversary has very little information about a model, it is possible to train a surrogate to perform the same task, craft adversarial examples against this substitute and then transfer them to the original classifier [226]. . For this purpose, it is generally assumed that the adversary can query the original system and observe the labels assigned to a set of synthetically generated inputs [184]. These inputs are generally drawn from regions of the space close to the decision boundary, so that they can later be used to build a single or multiple local surrogate models that ensure a good representation of the system's behavior in those areas.

In trying to stay undetected, the adversary may seek to minimize the number of queries. Hence, in theory, the problem of learning a surrogate model in this context could be casted as an active learning problem. However, no such comparison has been formally established to our knowledge. Instead, recent contributions have studied transferability of different types of adversarial examples. While some articles have succeeded in performing targeted attacks on large scale models [150], it has been reported that targeted adversarial examples do not always transfer from the surrogate to the target model [178]. Of particular interest to our work are *decision-based attacks*. These attacks differ from the general setting, where one assumes the adversary has access to the training data, by relying solely on knowledge of the final model decision [38].

All in all, the three disciplines of distillation, active learning and adversarial learning propose different approaches to understanding how a machine learning model acquires and represents knowledge. This understanding is essential to teach another model to learn an equivalent representation, to optimize the learning process and to prevent adversarial attacks. Lessons from these fields should therefore be considered when developing any knowledge inheritance technique. In the following chapter we leverage these insights to introduce copying as a mechanism to build differential replicas of trained classifiers without making any assumptions on the available information.

Lessons learned

- Starting from the first works on *rule extraction*, many research has been conducted on the issue of replacing one form of hypothesis representation with another by replicating a machine learning system's behavior using a different structure.

- Particularly successful in this task has been the extensive research on *model distillation*, where the knowledge acquired by a large cumbersome model, the teacher, is transferred to a simpler architecture, the student, which is more suitable under stringent deployment conditions. For this purpose, a transfer training set is labelled using the class probability outputs of the teacher as soft targets.

- In building such a transfer set different approaches have been assayed with the aim of obtaining a reduced set of highly informative samples to train the student.

- Particularly relevant in this regard are the advances in the field of *active learning*. Here, the learner actively selects samples from an available collection based on their informativeness and annotated them by making queries to a human oracle. While such query based learning has been mostly exploited to accelerate training of machine learning models, it can also be employed for malicious purposes.

- Examples of such use are numerous in the field of *adversarial learning*, where an external attacker exploits access to model's query interface to compromise its security. In this context, research has focused on anticipating potential attacks in scenarios were the attacker has different levels of knowledge on the system, in order to design the appropriate countermeasures.

- Altogether, the fields of knowledge distillation, active learning and adversarial learning approach the issue of knowledge representation in machine learning models from different perspectives and demonstrate the feasibility of transferring the knowledge from one form of representation to another.

- In what follows, we build from these findings to study how the knowledge codified in the decision behavior of a classifier can be inherited by another, which achieves a similar predictive performance and which may display additional characteristics, in situations where there is no access to the original model's internals, nor to its training data. We formalize this idea under the notion of copying.

Chapter 3

A theory for copying

3.1 Introduction

Depending on the amount of knowledge available about a system, different inheritance mechanisms are possible for differential replication. Here, we envisage the most restrictive scenario, where we make the minimum number of required assumptions about the amount of information that is accessible. We assume the model internals to be unknown and access to the model to be limited to a membership query interface that produces only hard predictions. In addition, we also assume the training data to be unknown or, simply, lost. This is a common scenario in highly regulated environments, such as that of a production infrastructure where access to the data may only be temporary or require specific permissions and models may be hosted in external servers.

Inheritance in this context can be understood as a form of zero-knowledge distillation, where the decision behavior of a larger model is transferred to a simpler one in circumstances where no knowledge is assumed about the training data or the model internals. Effectively, this corresponds to an scenario where the larger model is a black-box and distillation is conducted in a data-free way. Here, we refer to this mechanism as *copying*. In what follows we develop the theoretical background behind this concept and discuss how it can be implemented in practice. We begin by introducing the different elements that conform the copying pipeline.

3.2 Copying machine learning classifiers

Let us define a machine learning model as a function $f : \mathcal{X} \to \mathcal{T}$ from samples to labels, for $\mathcal{X}$ the input space and $\mathcal{T}$ the target space. We introduce the *training data* as a set $\mathscr{D} = \{(\boldsymbol{x}_i, t_i)\}_{i=1}^{M}$, where $\boldsymbol{x}_i \in \mathcal{X}$ and $t_i \in \mathcal{T}$ and M refers to the total number of samples. For the remaining of this work, we restrict to the case where the input space $\mathcal{X}$ is such that $\mathcal{X} = \mathbb{R}^d$ and the target space $\mathcal{T}$ is such that $\mathcal{T} = \mathbb{Z}_k$ for k the number of classes. This is, classification of real-valued attributes. Hence, we define $f_{\mathcal{O}}$ as a classifier trained on this set of labelled data points.

A *copy* is a new classifier $f_{\mathcal{C}}(\theta) \in \mathcal{H}_{\mathcal{C}}$, parameterized by θ, whose decision function mimics $f_{\mathcal{O}}$ all over the sample space $\mathcal{X}$. The *copy hypothesis space* $\mathcal{H}_{\mathcal{C}}$ includes all the possible model families the copy can belong to. As seen in *Chapter 1*, this new hypothesis space needs not coincide with that of the original classifier; this is, $f_{\mathcal{O}}$ and $f_{\mathcal{C}}$ need not belong to the same family of models. On the contrary, they usually don't. Yet, ideally, both models should display the same decision behavior.

3.2.1 The copy hypothesis space

The space $\mathcal{H}_{\mathcal{C}}$ defines a new form of knowledge representation. Copying therefore amounts to finding a suitable representation of $f_{\mathcal{O}}$ in this new space. We can understand this process as projecting the decision function $f_{\mathcal{O}}$ onto $\mathcal{H}_{\mathcal{C}}$, as shown in Fig. 3.1. Since $\mathcal{H}_{\mathcal{C}}$ may contain infinite individual models, there exist multiple possible projections. We identify the optimal copy $f_{\mathcal{C}}^*$ as the projection, or model, for which the distance to $f_{\mathcal{O}}$ is smallest. This optimal model is such that given a new, unseen sample $\boldsymbol{x}^*$ it predicts the output $y^* = f_{\mathcal{O}}(\boldsymbol{x}^*)$. The problem of copying is therefore characterized by the predictive distribution $\mathsf{P}(y^*|f_{\mathcal{O}}, \boldsymbol{x}^*)$. We can marginalize this distribution with respect to the copy parameters θ and write it as

$$\mathsf{P}(y^*|f_{\mathcal{O}}, \boldsymbol{x}^*) = \int_{\theta \in \tilde{\mathcal{H}}_{\mathcal{C}}} \mathsf{P}(y^*|\theta, f_{\mathcal{O}}, \boldsymbol{x}^*)\mathsf{P}(\theta|f_{\mathcal{O}}, \boldsymbol{x}^*)d\theta \tag{3.1}$$

for $\tilde{\mathcal{H}}_{\mathcal{C}}$ the equivalent hypothesis space corresponding to the copy parameters.

When building the copy, knowledge about the unseen data point $\boldsymbol{x}^*$ is not available, so that we can assume that $\mathsf{P}(\theta|f_{\mathcal{O}}, \boldsymbol{x}^*) = P(\theta|f_{\mathcal{O}})$. Conversely, once the optimal parameter set θ is obtained, interaction with $f_{\mathcal{O}}$ is no longer required. Hence, we can also assume that $\mathsf{P}(y^*|\theta, f_{\mathcal{O}}, \boldsymbol{x}^*) = \mathsf{P}(y^*|\theta, \boldsymbol{x}^*)$ and rewrite (3.1) as follows

$$\mathsf{P}(y^*|f_{\mathcal{O}}, \boldsymbol{x}^*) = \int_{\theta \in \tilde{\mathcal{H}}_{\mathcal{C}}} \mathsf{P}(y^*|\theta, \boldsymbol{x}^*)\mathsf{P}(\theta|f_{\mathcal{O}})d\theta \tag{3.2}$$

The equation above evaluates the probability distribution for all the possible values of θ and, consequently, for all the possible copies $f_{\mathcal{C}}(\theta) \in \mathcal{H}_{\mathcal{C}}$. However, we are only interested in the optimal copy $f_{\mathcal{C}}^*$, defined by the optimal parameter set θ^*. We take a *winner takes it all* approach to force the posterior

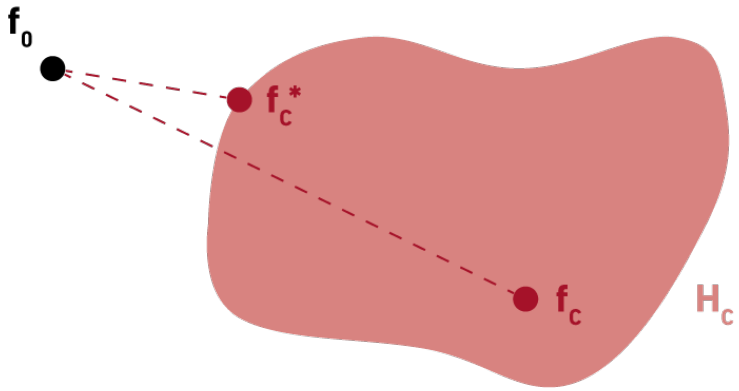


Fig. 3.1 Copying as a projection of a decision function $f_{\mathcal{O}}$ onto a new hypothesis space $\mathcal{H}_C$. The optimal copy f_C^* is the projection which is closest to $f_{\mathcal{O}}$.

to have the form of a point mass density $\mathsf{P}(\theta|f_{\mathcal{O}}) = \delta(\theta - \theta^*)$, where $\delta(.)$ corresponds to the *Dirac delta function*[1]. When doing so, we force all the probability mass to be placed onto θ^*, so that (3.2) can be rewritten in terms of the optimal copy parameters as

$$\mathsf{P}(y^*|f_{\mathcal{O}}, \boldsymbol{x}^*) = \mathsf{P}(y^*|\theta^*, \boldsymbol{x}^*) \tag{3.3}$$

Hence, we can conclude that the problem of copying can be understood as that of finding the optimal parameter values θ^* that maximize the posterior probability

$$\theta^* = \arg\max_{\theta} \mathsf{P}(\theta|f_{\mathcal{O}}) \tag{3.4}$$

The above expression depends solely on the form of the decision function $f_{\mathcal{O}}$. In the most general scenario, where we make no prior assumptions about the availability of the data or the accessibility of the model, neither the training set $\mathcal{D}$ nor the model internals are known. This means that we have no access to the training data points nor can we estimate their distribution throughout the input space. Equivalently, the specific architecture of the learned decision function $f_{\mathcal{O}}$ is unknown and our knowledge about its form only implicit. We can query $f_{\mathcal{O}}$ to obtain predictions on new data, but we lack a mathematical expression to represent it. Under these circumstances, we need to generate new data in order to gain information about this boundary and explore the decision behavior of $f_{\mathcal{O}}$ throughout $\mathcal{X}$.

---

[1] The *Dirac Delta function* is a generalized function that models the density of an idealized point mass. It is formally defined as

$$\delta(x - x_0) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{i\omega(x_o - x)} d\omega$$

This function is equal to zero everywhere except for $x = x_0$, where its value is infinitely large, and its integral over the entire real line is equal to 1.

3.2.2 The need for unlabelled data

We introduce a set of unlabelled data points $\boldsymbol{Z} = \{\boldsymbol{z}_j\}_{j=1}^{N}$, such that $\boldsymbol{z}_j \in \mathcal{X}$ and rewrite the expression (3.4) in terms of this set as follows

$$\theta^* = \arg\max_{\theta} \int_{\boldsymbol{z}\sim P_Z} \mathsf{P}(\theta|f_{\mathcal{O}}(\boldsymbol{z}))dP_Z, \tag{3.5}$$

for an arbitrary generating probability distribution P_Z, from which the new samples are independently drawn.

This distribution defines the spatial support for the copy, *i.e.* its plausible operational space. In the considered scenario, neither the training data $\mathcal{D}$ nor its distribution P are accessible. Hence, we cannot match P_Z to our estimate of P. This contrasts with most of the situations described in *Chapter 2*, where the training data distribution P is usually directly [112] or indirectly [40] known and this information can be exploited to choose P_Z accordingly. Note, however, that even in the absence of this information, it is still possible to define a suitable form for P_Z. Indeed, while this distribution can be chosen to strictly follow P, this is not always necessary, nor advisable.

Consider, for example, the problem shown in Fig. 3.2. The training distribution P defines a completely separable binary problem. The data points corresponding to each class are drawn from a Gaussian distribution and the learned decision boundary lies in a low density area of the space. Assuming we had full access to both the learned predictive system and its training data, it seems reasonable to define P_Z to be as close to P as possible. This would ensure that $\boldsymbol{Z}$ contains new data points located in the vicinity of the original samples. Indeed, by forcing $P_Z = P$ we ensure that the copy replicates the learned decision behaviour in those areas where the training data lie. However, the copy may display a completely different behaviour around the decision boundary itself, where these data are scarce. Alternatively, by forcing P_Z to place a higher density in the area around the boundary, we ensure a better fit in this region.

In general, defining P_Z to resemble the form of P ensures that the copy generalizes well in the original training data domain. However, this can also be achieved by other methods, such as updating the form of P_Z as we gain more information about $f_{\mathcal{O}}$, or forcing P_Z to adapt to the form of the copy hypothesis space. In any event, choosing P_Z adequately can be difficult, given that we have no intuition about which are the regions of interest where the training data are located. A more in depth discussion of how the generating distribution P_Z is chosen is presented in the following sections. Additionally, *Appendix A* reports results for a set of experiments with different choices of P_Z. For now, let us just say that P_Z can take the form of any arbitrary probability distribution, as long as it allows us to explore the form of $f_{\mathcal{O}}$ over $\mathcal{X}$.

Then, if we assume an arbitrary form for the probability distribution P_Z in (3.5) and because maximizing the posterior is equal to maximizing the log-posterior, we can rewrite this expression as

$$\theta^* = \arg\max_{\theta} \left[\log \left(\int_{\boldsymbol{z}\sim P_Z} \mathsf{P}(\theta|f_{\mathcal{O}}(\boldsymbol{z}))dP_Z \right) \right] \tag{3.6}$$

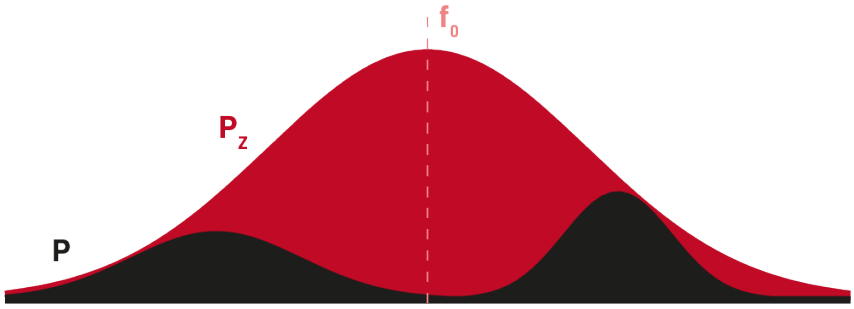


Fig. 3.2 Gaussian training data distribution P (in black), learned decision boundary $f_{\mathcal{O}}$ (in light red) and alternative gaussian distribution for P_Z (in red).

We can use *Jensen's inequality*[2] to move the logarithm inside the integral. When doing so, we obtain a lower bound[3] for θ^* of the form

$$\theta^* \geq \arg\max_{\theta} \int_{\boldsymbol{z}\sim P_Z} \log\left(\mathsf{P}(\theta|f_{\mathcal{O}}(\boldsymbol{z}))dP_Z\right)dP_Z \tag{3.7}$$

For simplicity, we assume equality and apply *Bayes' rule*. If we operate then with the terms inside the integral, we can develop the expression above as

$$\begin{aligned}
\theta^* &= \arg\max_{\theta} \int_{\boldsymbol{z}\sim P_Z} \log\left(\frac{\mathsf{P}(f_{\mathcal{O}}(\boldsymbol{z})|\theta)\mathsf{P}(\theta)}{P(f_{\mathcal{O}}(z))}\right)dP_Z \\
&= \arg\max_{\theta} \left[\int_{\boldsymbol{z}\sim P_Z} \log\mathsf{P}(f_{\mathcal{O}}(\boldsymbol{z})|\theta)dP_Z - \int_{\boldsymbol{z}\sim P_Z} \log\mathsf{P}(f_{\mathcal{O}}(\boldsymbol{z}))dP_Z + \log\mathsf{P}(\theta)\right] \\
&= \arg\max_{\theta} \left[\int_{\boldsymbol{z}\sim P_Z} \log\mathsf{P}(f_{\mathcal{O}}(\boldsymbol{z})|\theta)dP_Z + \log\mathsf{P}(\theta)\right]
\end{aligned} \tag{3.8}$$

where we drop the term $\int_{\boldsymbol{z}\sim P_Z} \log\mathsf{P}(f_{\mathcal{O}}(\boldsymbol{z}))dP_Z$, which has no dependence on θ.

---

[2]Let f be a convex function, and let X be a random variable. *Jensen's inequality* states that

$$E[f(X)] \geq f(E[X])$$

Moreover, if f is strictly convex, then $E[f(X)] = f(E[X])$ holds true if and only if $X = E[X]$ with probability 1. The inequality also holds for concave functions f, but with the direction of all the inequalities reversed; so that given a concave function f, then $E[f(X)] \leq f(E[X])$. Specifically, note that the function $f(x) = log(x)$ is a concave function, since $f''(x) = -1/x2 < 0$ over its domain $x \in \mathbb{R}^+$.

[3]Note that even though maximization of the lower bound also maximizes the original function, the optimal value of the lower bound may be different from that of the original objective function.

3.2.3 Copying under the empirical risk minimization framework

The solution to the expression above depends, among other things, on the specific form of the considered models expressed through the two probability distributions $\mathsf{P}(f_{\mathcal{O}}(\boldsymbol{z})|\theta)$ and $\mathsf{P}(\theta)$. In this work, we are interested in building hard decision copies. We can therefore recover the regularized empirical risk minimization framework [246] by approximating these two distributions with an exponential family, so that

$$\mathsf{P}(f_{\mathcal{O}}(\boldsymbol{z})|\theta) \propto e^{-\gamma_1 \ell_1(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z}))}; \quad \mathsf{P}(\theta) \propto e^{-\gamma_2 \ell_2(\theta,\theta^+)}$$

for $\ell_i(a,b)$ a measure of disagreement between a and b, γ_1 and γ_2 normalization parameters and θ^+ our prior about θ. Using this approximation we can rewrite (3.8) as

$$\theta^* = \arg\min_{\theta} \left[\int_{\boldsymbol{z}\sim P_Z} \gamma_1 \ell_1(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z}))dP_Z + \gamma_2 \ell_2(\theta,\theta^+) \right] \tag{3.9}$$

The first term in this expression is the expected value of the disagreement between the model $f_{\mathcal{O}}$ and the copy $f_{\mathcal{C}}$ over the set $\mathcal{Z}$. Under the empirical risk minimization framework, we can identify this first term as the expected loss particularized to the copying problem and defined it as

$$\begin{aligned} R^{\mathcal{F}}(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z})) &= \mathbb{E}_{\boldsymbol{z}\sim P_Z}[\ell_1(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z}))] \\ &= \int_{\boldsymbol{z}\sim P_Z} \ell_1(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z}))dP_Z \end{aligned} \tag{3.10}$$

over the probability distribution P_Z.

We refer to this term as the *fidelity error*. This error captures all the loss of copying. In its most general form, it corresponds to the integral $\int_{\boldsymbol{z}\sim P_Z} \log P(f_{\mathcal{O}}(z)|\theta)dP_Z$ in (3.8), *i.e.* the probability that the decision behavior of the copy resembles that of the model. To numerically solve this integral we need to gather knowledge about how $f_{\mathcal{O}}$ behaves for any given $\boldsymbol{z} \sim P_Z$, *i.e.* for the whole input space. However, this is not possible in practice, because it requires that we draw an infinite number of samples from this distribution. Hence, we approximate the term above using the empirical risk instead.

The particularization of the empirical risk to the copying setting is the *empirical fidelity error*, $R^{\mathcal{F}}_{emp}$, which we define to be the empirical version of the fidelity error

$$R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z})) = \frac{1}{N}\sum_{j=1}^{N} \ell_1(f_{\mathcal{C}}(\boldsymbol{z_j},\theta), f_{\mathcal{O}}(\boldsymbol{z_j})) \tag{3.11}$$

for N the total number of samples.

The first term in (3.9) can therefore be approximated by the empirical fidelity error as above. The second term, in turn, refers to the fit of the parameters to the prior and can be identified as the regularization term

$$\Omega(\theta) = \ell_2(\theta, \theta^+) \tag{3.12}$$

Hence, in the discrete case, we can approximate the optimal copy parameter values by rewriting the two terms above as follows

$$(\theta^*, \boldsymbol{Z}^*) = \arg\min_{\theta, \boldsymbol{z}_j \in \boldsymbol{Z}} \left[R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z}, \theta), f_{\mathcal{O}}(\boldsymbol{z})) + \Omega(\theta) \right] \tag{3.13}$$

$$= \arg\min_{\theta, \boldsymbol{z}_j \in \boldsymbol{Z}} \left[\frac{1}{N} \sum_{j=1}^{N} \gamma_1 \ell_1(f_{\mathcal{C}}(\boldsymbol{z}_j, \theta), f_{\mathcal{O}}(\boldsymbol{z}_j)) + \gamma_2 \ell_2(\theta, \theta^+) \right] \tag{3.14}$$

where we introduce $\boldsymbol{Z}^* = \{\boldsymbol{z}_j^*\}_{j=1}^N$ as an optimal set of artificially generated samples. We label this set using the predictions of the model $f_{\mathcal{O}}$ as hard targets. This is, for each new sample $\boldsymbol{z}_j$ we assign a label that corresponds to the hard prediction output by $f_{\mathcal{O}}$ for this point. We refer to the resulting set of labelled data points $\mathcal{Z}^* = \{(\boldsymbol{z}_j^*, f_{\mathcal{O}}(\boldsymbol{z}_j^*)\}_{j=1}^N$ as the *synthetic set*.

The expression above can be understood as a dual optimization problem, where we simultaneously optimize the model parameters θ and the synthetic set $\boldsymbol{Z}$ used to explore the decision behavior of $f_{\mathcal{O}}$ over the sample space $\mathcal{X}$. This dual optimization lies at the very core of copying. The same set of samples $\mathcal{Z}$ is used both to build the copy and to evaluate it by measuring how faithfully it replicates the decision function $f_{\mathcal{O}}$. As a result, copying requires not only that we optimize the parameters θ, but also that we refine the set $\boldsymbol{Z}$ to ensure a reliable representation of the learned decision behavior.

From the perspective, once again, of regularized empirical risk minimization, we can interpret (3.14) as a way of applying the concentration of measure inequalities to bound the generalization error in terms of the empirical risk in the form

$$R^{\mathcal{F}} \leq R^{\mathcal{F}}_{emp} + \mathcal{O}\left(\sqrt{\frac{C}{N}}\right), \tag{3.15}$$

for C a parameter that governs the classifier capacity and N the size of the synthetic dataset. Although not all generalization bounds have this same form, we find this trade-off between a capacity measure and the number of samples in all of them, including VC-dimension [245][247] or covering numbers approaches [57], Rademacher complexity frameworks [172], PAC-Bayes bounds on distributions of hypothesis [166] and compression bounds [10].

Assuming this formulation, the dual optimization in (3.14) can be understood as the scalarization of a multi-objective optimization function $(R_{emp}(), \Omega)$, for γ_2 the parameter that controls the trade off between the empirical risk and the capacity term $C \approx \Omega(\theta)$. The solution to this optimization problem

defines a Pareto's optimal surface, from which the optimal point is usually chosen using a validation dataset. Many algorithms, including SVMs, neural networks, boosting or Bayesian models are examples of problems of this form. In the following, we describe how the specific characteristics of copying can be exploited to reach the optimal operation point.

3.2.4 Solving the copying problem

When generating the labelled synthetic dataset $\mathcal{Z}^*$, the class membership predictions output by the model $f_{\mathcal{O}}$ define a hard classification boundary. The resulting problem, represented by $\mathcal{Z}^* = \{(\boldsymbol{z}_j^*, f_{\mathcal{O}}(\boldsymbol{z}_j^*)\}_{j=1}^N$, has two important characteristics: (1) it is always separable and (2) we can potentially increase the sample size N indefinitely. The model $f_{\mathcal{O}}$ acts as a form of regularizer that notably simplifies the problem for the copy. Hence, if we assume a copy with enough capacity it is always possible to achieve zero empirical fidelity error, so that $R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z}, \theta), f_{\mathcal{O}}(\boldsymbol{z})) = 0$. The error then only depends on the generalization gap for the synthetic dataset. Plus, because we control the synthetic data generation process, we can have an infinite stream of samples at our disposal when learning this simplified problem. The generalization error can therefore be asymptotically reduced to zero. This means that, in theory, copying can be performed without loss and redefined as an unconstrained optimization problem of the form

$$\underset{\theta, \boldsymbol{Z}}{\text{minimize}} \quad R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z}, \theta), f_{\mathcal{O}}(\boldsymbol{z})). \tag{3.16}$$

Yet, in practice, the synthetic dataset is always finite. Moreover, as the dimensionality of $\mathcal{X}$ increases, the task of obtaining a representative set $\mathcal{Z}^*$ becomes harder. As a result, despite the problem being separable, we might not reach the optimal operation point given our limited knowledge of its form. It therefore stands to reason to impose that the copy has small capacity, $\Omega(\theta)$, and instead rewrite the copying problem as

$$\begin{aligned} &\underset{\theta, \boldsymbol{Z}}{\text{minimize}} \quad \Omega(\theta) \\ &\text{subject to} \quad \|R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z}, \theta), f_{\mathcal{O}}(\boldsymbol{z})) - R^{\mathcal{F}}_{emp}(f^{\dagger}_{\mathcal{C}}(\boldsymbol{z}, \theta), f_{\mathcal{O}}(\boldsymbol{z}))\| < \epsilon, \end{aligned} \tag{3.17}$$

for $f^{\dagger}_{\mathcal{C}}(\boldsymbol{z}, \theta)$ the solution to the unconstrained problem (3.16) and ϵ a defined tolerance. The term $\|R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}, f_{\mathcal{O}}) - R^{\mathcal{F}}_{emp}(f^{\dagger}_{\mathcal{C}}, f_{\mathcal{O}})\| < \epsilon$ defines a feasible set of parameters. Hence, the solution to (3.17) achieves the smallest capacity $\Omega(\theta)$ while keeping $R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z}, \theta), f_{\mathcal{O}}(\boldsymbol{z}))$ within a tolerance of the unconstrained optimal value of the empirical fidelity error, $R^{\mathcal{F}}_{emp}(f^{\dagger}_{\mathcal{C}}(\boldsymbol{z}, \theta), f_{\mathcal{O}}(\boldsymbol{z}))$. We argue that there exists a set of parameters θ that fulfill this constraint.

In this definition of copying, the optimal loss value is known in advance. There are also other classification problems where we find this. Consider, for example, the case of SVMs, where the hinge-loss value is defined to be zero. However, this is not always true. See, for example, least-square errors in

classification[4] where the global minimum of the error function is not known. Or the cross-entropy loss in artificial neural networks. In general, when training a model, one does not know what the optimal values of both the loss and the regularization term are. Copying therefore differs from the standard multi-objective optimization in a pure learning setting. Instead of having a *Pareto's surface* of plausible optimal solutions, as long as $\Omega(\theta)$ is convex, the solution to (3.17) is unique.

Moreover, in cases where the capacity is directly modelled, this optimization can be straightforwardly solved. This holds, for example, for SVMs and neural networks, where we can use a regularization function, or for Bayesian models, where we can appropriately select the priors. For other model architectures, such as decision trees, the complexity control must be done by either early stopping or by an external process, such as post- or pre-pruning of the leaves. Finally, when using techniques such as boosting or deep learning, which exhibit a delayed overfitting effect [209][181][39], we can exploit this property to our advantage to directly solve the problem (3.16) instead of (3.17).

3.3 The single-pass approach

Copying involves the dual optimization of the synthetic dataset $\mathcal{Z}$ and the copy parameters θ through (3.17), or its simplified form (3.16). Solving this optimization directly requires that the copy hypothesis space have certain properties, such as online updating. We study this case in the following section. Here, we propose a more straightforward solution. We consider the simplest approach to solving the dual copying problem: the *single-pass approach*. We formulate this approach in theory and bridge the gap between theory and practice by providing meaningful insights on how it works in a series of toy examples.

In the single-pass approach we cast the simultaneous optimization problem into one where only a single iteration of an alternating projection optimization scheme is used. In other words, we effectively split the problem in two independent sub-problems: (1) finding the optimal synthetic dataset $\mathcal{Z}^*$ and (2) optimizing for θ^*. The single-pass approach works as follows:

1. *Synthetic sample generation.* The first step of the single-pass approach involves finding the optimal set of synthetic data points $\boldsymbol{Z}^*$. This set is that for which the empirical fidelity error is minimal,

$$\boldsymbol{Z}^* = \arg\min_{\boldsymbol{Z}} R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z}, \theta), f_{\mathcal{O}}(\boldsymbol{z}))$$

   In obtaining this set we can define the optimal synthetic dataset $\mathcal{Z}^*$, by labelling all the samples using $f_{\mathcal{O}}$.

[4]Instead of tracking the empirical risk we can track the empirical error, which can be set to zero due to the separability property.

2. *Optimal parameter set.* Once having defined an optimal set of labelled synthetic data points, we use it to train the copy. We do so by looking for the optimal parameter set θ^* that minimizes the constrained problem

$$\begin{aligned} \underset{\theta}{\text{minimize}} \quad & \Omega(\theta) \\ \text{subject to} \quad & \|R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z})) - R^{\mathcal{F}}_{emp}(f^{\dagger}_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z}))\| < \epsilon, \end{aligned}$$

or its simplified version (3.16), provided that the adequate conditions hold.

An example of the single-pass copy is shown in Fig. 3.3, where the binary decision function learned by a fully-connected neural network is copied using a decision tree classifier. The tree-based copy learns from a set of synthetic samples drawn from a uniform distribution and labelled according to the hard predictions output by the neural net.

3.3.1 Meaningful insights

In order to build an intuition on how the single-pass approach works in practice, we focus on the two steps described above. We begin by studying the synthetic sample generation process and then discuss certain properties of the copying framework in a practical setting.

Synthetic sample generation

For the sake of this discussion, let us consider a binary classification problem and let $f_{\mathcal{O}}(\boldsymbol{z}) \in \{-1, +1\}$ and $f_{\mathcal{C}}(\boldsymbol{z}, \theta) \in \{-1, +1\}$, for any $\boldsymbol{z}_j \in \mathcal{X}$ stand for the model and copy decision functions, respectively. Let us also particularize ℓ_1 to the 0/1 loss. For this case, the empirical fidelity error in (3.11) can be rewritten as

$$\begin{aligned} R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z})) &= \frac{1}{2N}\sum_{j=1}^{N}\Big|f_{\mathcal{O}}(\boldsymbol{z}_j)) - f_{\mathcal{C}}(\boldsymbol{z}_j,\theta)\Big| \\ &= \frac{1}{2N}\sum_{j=1}^{N}\Big|f_{\mathcal{O}}(\boldsymbol{z}_j)\Big|\Big|1 - \frac{f_{\mathcal{C}}(\boldsymbol{z}_j,\theta)}{f_{\mathcal{O}}(\boldsymbol{z}_j)}\Big| \\ &= \frac{1}{2N}\sum_{j=1}^{N}\Big(1 - \frac{f_{\mathcal{C}}(\boldsymbol{z}_j,\theta)}{f_{\mathcal{O}}(\boldsymbol{z}_j)}\Big) \\ &= \frac{1}{2N}\sum_{j=1}^{N}1 - \frac{1}{2N}\sum_{j=1}^{N}\frac{f_{\mathcal{C}}(\boldsymbol{z}_j,\theta)}{f_{\mathcal{O}}(\boldsymbol{z}_j)} \end{aligned}$$

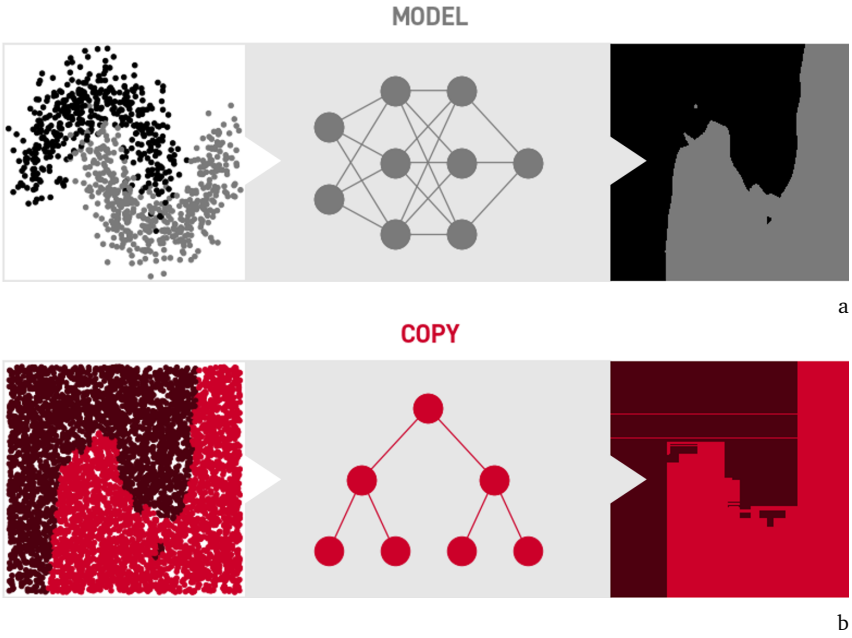


Fig. 3.3 Example of the single-pass copy approach. (a) Training data, model architecture and resulting decision boundary. (b) Generated synthetic data, copy architecture and copy decision function.

$$= \frac{1}{2} - \frac{1}{2N}\sum_{j=1}^{N} f_{\mathcal{C}}(\boldsymbol{z}_j, \theta) f_{\mathcal{O}}(\boldsymbol{z}_j); \quad \boldsymbol{z}^{(N)} \sim P_Z$$

Let us now define a partition of the space such that $\mathcal{X} = \mathcal{X}_+ \cup \mathcal{X}_-$ and $\mathcal{X}_+ \cap \mathcal{X}_- = \emptyset$, where $\mathcal{X}_+ = \{\boldsymbol{z} | \boldsymbol{z} \in \mathcal{X}, f_{\mathcal{O}}(\boldsymbol{z}) = 1\}$ and $\mathcal{X}_- = \{\boldsymbol{z} | \boldsymbol{z} \in \mathcal{X}, f_{\mathcal{O}}(\boldsymbol{z}) = -1\}$ are the two sub-spaces defined by the model. We can rewrite the equation above in terms of this partition as

$$R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z}, \theta), f_{\mathcal{O}}(\boldsymbol{z})) = \frac{1}{2} - \frac{1}{2N_+}\sum_{j=1}^{N_+} f_{\mathcal{C}}(\boldsymbol{z}_j, \theta) + \frac{1}{2N_-}\sum_{j=1}^{N_-} f_{\mathcal{C}}(\boldsymbol{z}_j, \theta)$$

for N_+ and N_- the number of samples lying in $\mathcal{X}_+$ and $\mathcal{X}_-$, respectively.

We define the probability of a sample lying in $\mathcal{X}_+$ as $p_+ = \mathbb{P}(\boldsymbol{z} \in \mathcal{X}_+)$ and the probability of a sample lying in $\mathcal{X}_-$ as $p_- = \mathbb{P}(\boldsymbol{z} \in \mathcal{X}_-)$. These two probabilities depend on the *size* of the positive and negative domains. This is,

$$p_+ = \int_{\boldsymbol{z} \in \mathcal{X}_+} P_Z(\boldsymbol{z}) dz, \quad p_- = \int_{\boldsymbol{z} \in \mathcal{X}_-} P_Z(\boldsymbol{z}) dz$$

so that it holds that $N_+ = Np_+$ and $N_- = Np_-$. Thus, once again, we can rewrite the empirical fidelity error for this case as

$$R^{\mathcal{F}}_{emp}(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z})) = \frac{1}{2} - \frac{1}{2Np_+}\sum_{j=1}^{Np_+} f_{\mathcal{C}}(\boldsymbol{z}_j,\theta) + \frac{1}{2Np_-}\sum_{j=1}^{Np_-} f_{\mathcal{C}}(\boldsymbol{z}_j,\theta).$$

The minimization of this expression to obtain the optimal set $\boldsymbol{Z}^*$ explicitly depends on the form of the generating probability distribution P_Z. In the simplest case, we can assume this distribution to be flat on the domain $\mathcal{X}$, so that $\boldsymbol{z} \sim \mathcal{U}(\mathcal{X})$. In other words, we can assume the synthetic samples to be drawn from a uniform probability distribution across the space. Under this assumption, p_+ and p_- directly correspond to the fraction of volume for each of the two classes. Then, recalling the form of the error for the Monte Carlo estimator under this distribution, we can express the standard error associated to the evaluation of the empirical fidelity error $R^{\mathcal{F}}_{emp}$ as

$$\sigma(R^{\mathcal{F}}_{emp}) \propto \mathcal{O}\Big(\frac{1}{\sqrt{Np_+}} + \frac{1}{\sqrt{Np_-}}\Big). \tag{3.18}$$

We use this expression to extract relevant insights for the synthetic sample generation process. First, we confirm the need to define an attribute representation $\mathcal{X}$. This is a reasonable assumption, since we need to have an approximate idea of the dynamic range of all variables in order to build meaningful queries[5].

Second, we note that in some situations there might be a mismatch between the decision boundary achievable by the copy and $f_{\mathcal{O}}$. This issue is related to a capacity gap between both models and can prevent the copy from being able to converge to the desired solution altogether. As a consequence, a given synthetic dataset may not perform equally for different copy hypotheses. Consider, for example, a non-linear decision function and a linear copy hypothesis space. Exploring the twists of the decision boundary during the synthetic sample generation process may not be relevant in this situation. Hence, in order to effectively exploit each generated sample, we should consider the specific properties and assumptions of the copy hypothesis space everytime.

Another important issue that emerges from the derivation above is *volume imbalance*, which appears when one or more of the classes occupy a region of the space much smaller than the rest.

The issue of volume imbalance The empirical fidelity error depends on the fraction of volume occupied by each decision region. If the spatial support of one class is small with respect to the total volume, it may be difficult to have a meaningful number of samples on that region, which may result in large approximation errors. Importantly, this issue is independent of the original label distribution, *i.e.* it is a different problem from that of learning with unbalanced data.

In Fig. 3.4(a), we show a binary dataset with a balanced label distribution. Despite the number

[5]Note that even in those cases where this information is not known beforehand, it is still possible to infer it by interacting with the model's query interface. In general, most model APIs display a default error message when input queries fall out of the defined problem domain. Hence, coming up with the appropriate domain is more an issue of time that it is of skill. Only in adversarial scenarios, where the user may be limited to a given number of queries, might this impose a serious restriction. Still, there exist ways to circumvent this issue, even in the most stringent scenarios.

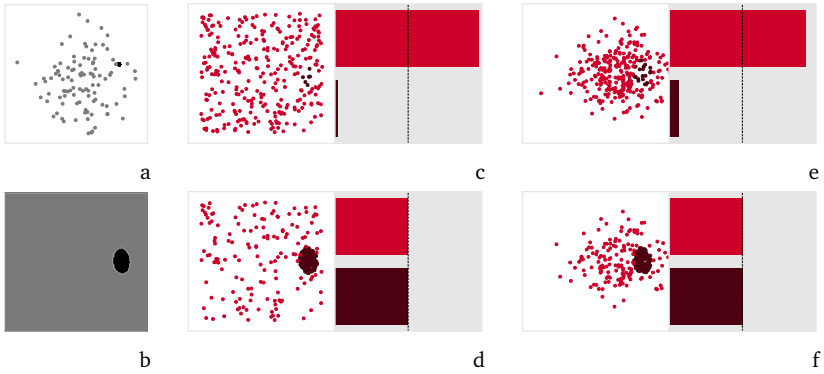


Fig. 3.4 (a) Training dataset. (b) Decision boundary learned by a Gaussian Process classifier. (c) Raw and (d) balanced synthetic datasets generated from a uniform distribution. (e) Raw and (f) balanced synthetic datasets generated from a uniform distribution and a standard normal distribution.

of instances per label being equal, there are notable differences in the volume occupied by each of the classes. While points belonging to one of the classes (in gray) are spread out throughout the space, those belonging to the other (in black) are concentrated in a very small region. The resulting decision boundary is displayed in Fig. 3.4(b). The form of this boundary reflects this disparity in the size of both classes.

To copy this model, we assay two different forms for the probability distribution P_Z. In a preliminary approach, we generate samples uniformly at random until we reach the desired number of points. In Fig. 3.4(c) we plot the resulting set, together with its corresponding label distribution. In addition, we also generate samples using a standard normal distribution. The resulting set is shown in Fig. 3.4(e). In both cases, the resulting synthetic datasets, are notably unbalanced: there is one class for which we only recover a few number points, whereas we generate numerous samples for the other. Again, recall that this result is unrelated to class distribution: both classes had originally the same number of samples. Instead, it is related to how the different classes are distributed.

Solving the issue of volume imbalance is no easy task. In scenarios where we can assume a certain knowledge of the training data distribution, this information can be used to focus sampling on specific regions of the space. However, in the most restrictive cases, where there exists no access to these data, this is not possible. Fortunately, the volume imbalance effect can be alleviated by a good choice of P_Z. For example, we can try to infer a sampling distribution that allocates a large amount of the probability mass around the unknown decision boundary. Or we can continuously update the form of P_Z to conduct a guided search of the space by incorporating new knowledge at each new step. In Appendix A we include a comparison of different sampling algorithms for the copying setting, including a technique that focuses on boundary exploration, a Bayesian-based optimizer, a modified version of the Jacobian approach proposed by [184] and raw random sampling.

Alternatively, we can also alleviate the issue of volume imbalance by imposing that the synthetic set be balanced with respect to the class labels. This can be done using heuristics that balance a general

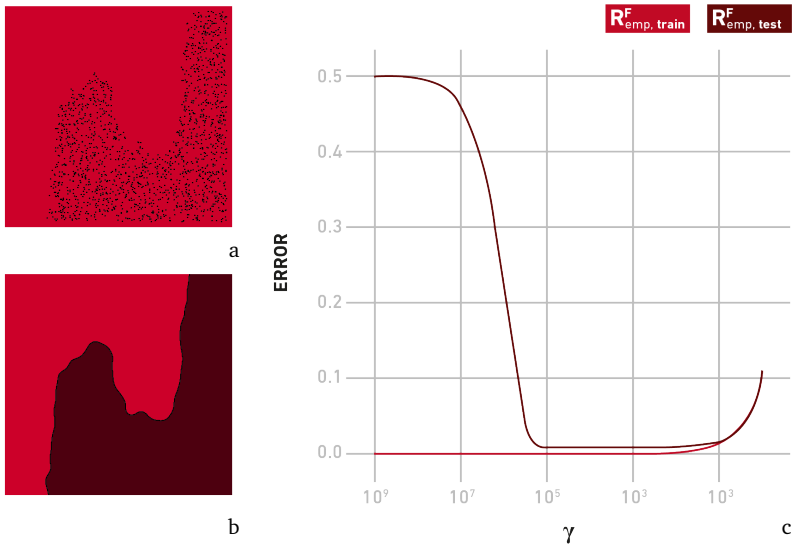


Fig. 3.5 Decision boundaries learned by copies with (a) a maximal and (b) an optimal γ. (c) Empirical risk and generalization error for decreasing values of γ.

exploration of the space with exploitation around the areas of interest[6]. In Fig. 3.4(d) and Fig. 3.4(f) we display the distribution of synthetic data points obtained when forcing the data generator to focus on those areas where the misrepresented class is located, for both the uniform and the normal distribution. The resulting label distributions are now balanced. Copies trained on these data are more likely to recover the original decision boundary.

Optimal parameter set

The second part of the alternating projection scheme in the single-pass approach corresponds to finding the optimal parameter set for the copy. This set is that for which the copy capacity is minimized while maintaining the empirical fidelity error reasonably close to the unconstrained value in (3.16). This can be attained in practice without applying any regularization technique to prevent overfitting of the copy.

For illustration purposes, consider a radial basis function kernel SVM. This model is defined by a kernel function of the form $\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}') = e^{-\gamma||\boldsymbol{x}-\boldsymbol{x}'||^2}$, where $||\boldsymbol{x}-\boldsymbol{x}'||^2$ corresponds to the squared Euclidean distance, and γ is the inverse of the radius of influence of the support vectors, *i.e.* the width of the kernel. This means, in essence, that γ controls the capacity: the larger its value, the higher the complexity of the model. Hence, minimizing the model capacity in (3.17) amounts to minimizing γ. In Fig. 3.5 we show how this can be exploited in practice to copy the neural net in Fig. 3.3 using synthetic samples drawn at random from a uniform distribution.

[6]Note that this approach is not guaranteed to provide good results in large input spaces with high dimensionality. In these cases, the synthetic sample generator might not be able to recover one or more of the classes. Again, in cases where the training data are known, this knowledge can be used to sample the appropriate regions intensively. For every other case, finding at least one sample of the minority class can become extremely hard.

In particular, Fig. 3.5(a) shows the copy decision function for a maximal value of γ, such that the second term in (3.17) is satisfied and the empirical error is zero. Fig. 3.5(b) shows the decision boundary for a copy with optimal capacity γ, computed for a tolerance $\epsilon = 1e-4$. This solution results from sequentially reducing the value of γ and monitoring the change in accuracy until the error deviation is greater than ϵ. When comparing both plots we observe the improvement in generalization performance. This improvement is also seen in Fig. 3.5(c), where train and generalization errors of the copy are shown for decreasing values of γ. For a bounded value of the empirical error, the generalization error is reduced as we decrease the capacity of the copy. This result shows that, unlike the classical machine learning setting, where capacity is optimized during the validation step, in copying it is possible to optimize the capacity during training. This has a profound impact on how copying is performed, since we may think of a more effective design of algorithms than using standard machine learning pipelines and assumptions.

Finally, note that the specific choice of copy architecture has a significant impact on performance. Depending on the chosen hypothesis space, copies may behave very differently when confronted with the same set of synthetic data points. We measure this effect using the *capacity error*. In what follows, we provide an intuition of what the effect of this error is in practice. We refer the reader to *Chapter 4* for a more in-depth discussion.

Capacity error The capacity of a classifier is a measure of its complexity. A mismatch of capacity between model and copy can lead to poor performance results, even in cases where the synthetic dataset properly covers the input space. Moreover, this can also hold in cases where the optimal parameter set θ^* is obtained. Take the case of a linear logistic regression and a support vector machine. The decision functions learned by copies based on these two architectures are notably different. Given the same set of synthetic points, a logistic model may not be able to fully recover the form of the considered decision boundary if, for example, this is non-linear. This is because in this case the original classifier is not contained in the new hypothesis space. For the SVM, the mismatch in capacity may presumably not be so pronounced and therefore the copy decision boundary may be much more precise in this case.

3.4 The dual-pass approach

In the copying setting, learning is mediated by a synthetic dataset specifically generated to represent the decision behavior of the target model throughout the space. Hence, the adequacy of this dataset is crucial to ensure a good result. In the single-pass approach optimality of the synthetic dataset is not necessarily guaranteed. Synthetic data are gathered in a single run and no corrective mechanism is established to guide sampling in future iterations. In most cases, unless the choice of the distribution P_Z is optimal, chances are that the resulting dataset is in need of further refinement. Moreover, when it comes to the copy parameters, these are optimized to fit the given synthetic data. Hence, while the copy may satisfy (3.14), there is no guarantee of a good generalization performance. Unless, of course, we assume we have

access to an infinite stream of synthetic data. This condition, however, is never met in practice. In cases where the size of the synthetic dataset is very large, the single-pass can ensure a good fit of the copy. Yet, there will still be several regions of the space where no synthetic data are generated and where the copy will need to infer the missing knowledge. If these knowledge gaps are located in areas close to the decision boundary, this may prevent the copy from reaching a good performance altogether. Moreover, even in the best case scenario, dealing with a large number of synthetic data may cause memory errors. This is because the computational resources of any company are finite. Hence, while it provides a valid approach to solving the dual optimization problem, the single-pass is generally not optimal in practice.

In this section, we move on from the single-pass approach to propose a strategy to tackle the dual optimization problem in (3.14) directly: the *dual-pass*. The dual-pass is based on an alternating projection scheme, where the objective function is optimized first in terms of one set of parameters and then the other. During each iteration, we first optimize the copy parameters with respect to the synthetic data and then optimize the generated set of synthetic data points with respect to the copy.

The main idea behind the dual-pass approach is that it is possible to train copies incrementally, by updating the model parameters step by step as the set of synthetic samples are refined with every new iteration. For simplicity, let us define P_Z to be a uniform distribution throughout the domain and set the number of iterations to $\mathcal{T}$. Let us also define the error buffer as a set ε, which at any point in time stores information about the errors of the copy and the set $\mathcal{I}$ which stores the iteration t each of these errors correspond to. We proceed as follows:

1. We begin by drawing a random set of samples of size N and labelling them according to $f_{\mathcal{O}}$. We identify the resulting set as $\mathscr{Z}_0$ and use it to train $f_{\mathcal{C}}$.

2. For every iteration t, we identify those instances in $\mathscr{Z}_{t-1}$ where the copy predicts an incorrect class label. We store these samples in the set of errors $\varepsilon = \{\boldsymbol{z} \in \mathscr{Z}_{t-1} | f_{\mathcal{O}}(\boldsymbol{z}) \neq f_{\mathcal{C}}(\boldsymbol{z})\}$ and the corresponding iteration in the indexing buffer $\mathcal{I} = \{t | f_{\mathcal{O}}(\boldsymbol{z}) \neq f_{\mathcal{C}}(\boldsymbol{z})\}$. The size of the error set is governed by a memory parameter m, which governs the number of iterations a given sample is stored in memory. Samples that exceed this value are removed from ε.

3. We draw a new set of synthetic samples $\mathscr{Z}_t^{new}$ of size $M = N - |\varepsilon|$, for $|\varepsilon|$ the size of the error buffer.

4. We update $\mathscr{Z}_t$ as $\mathscr{Z}_t = \mathscr{Z}_t^{new} \cup \varepsilon$ and use it to update $f_{\mathcal{C}}$.

5. We repeat steps 2 and 4 until all the iterations are complete.

In Alg. 1 we propose a possible algorithm for the dual-pass as described in the methodological procedure above.

Algorithm 1 Dual-pass(**int** N, **int** $\mathcal{T}$, **int** m, **Classifier** $f_{\mathcal{O}}$)

$\varepsilon \leftarrow \emptyset$ ▷ Error buffer
$\mathcal{I} \leftarrow \emptyset$ ▷ Iteration buffer
$f_{\mathcal{C}} \leftarrow$ Classifier() ▷ Instantiation of the copy
for $t = 1$ to $\mathcal{T}$ **do**
    $Z^{new} \leftarrow \emptyset$
    M = N - $|\varepsilon|$
    while $|Z^{new}| < M$ **do**
        $z_a \sim Uniform(\mathcal{X}), \quad y_a \leftarrow f_{\mathcal{O}}(z_a)$
        $Z^{new} \leftarrow Z^{new} \cup \{(z_a, y_a)\}$
    end while
    $Z \leftarrow Z^{new} \cup \varepsilon$
    train_model($f_{\mathcal{C}}, Z$)
    for $z \in Z$ **do**
        if $f_{\mathcal{C}}(z) \neq f_{\mathcal{O}}(z)$ **then**
            $\varepsilon \leftarrow z$
            $\mathcal{I} \leftarrow t$
        end if
    end for
    for $e, i \in \varepsilon, \mathcal{I}$ **do**
        if $t - i > m$ **then**
            $\varepsilon \leftarrow \varepsilon - e$
            $\mathcal{I} \leftarrow \mathcal{I} - i$
        end if
    end for
end for

3.4.1 Meaningful insights

To provide an intuition of how this works in practice, we assay the dual-pass approach in a series of toy problems that serve as an example of real-world classification systems. In particular, we use the datasets displayed in Fig. 3.6, where each plot corresponds to the training data for four different binary classification problems. For each of these problems, we fit a different classifier. We use adaboost to fit the data in Fig. 3.6(a), a gaussian-kernel svm to fit the data in Fig. 3.6(b), a multilayer perceptron to fit the data in Fig. 3.6(c) and a random forest to fit the data in Fig. 3.6(d). The decision functions learnt by these classifiers are shown in the top row of Fig. 3.7.

We copy these classifiers using incremental decision trees, which can be trained in different runs by adding new samples each time. We set N, the number of synthetic data points for each iteration to 100 and allow a total of 1000 iterations. For simplicity, we use the most straightforward implementation of the dual-pass approach, where we set the memory parameter m to 1. This means that errors are only kept in buffer for a single iteration and then the buffer is emptied again.

The middle row of Fig. 3.7 shows the copy decision functions resulting from the first iteration, *i.e.* corresponding to copies trained on the initial draw of 100 synthetic samples. We show these results for

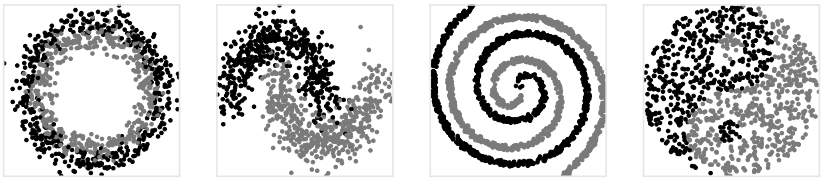


Fig. 3.6 Training data for (a) circles, (b) moons, (c) spirals and (d) yin-yang binary classification problems.

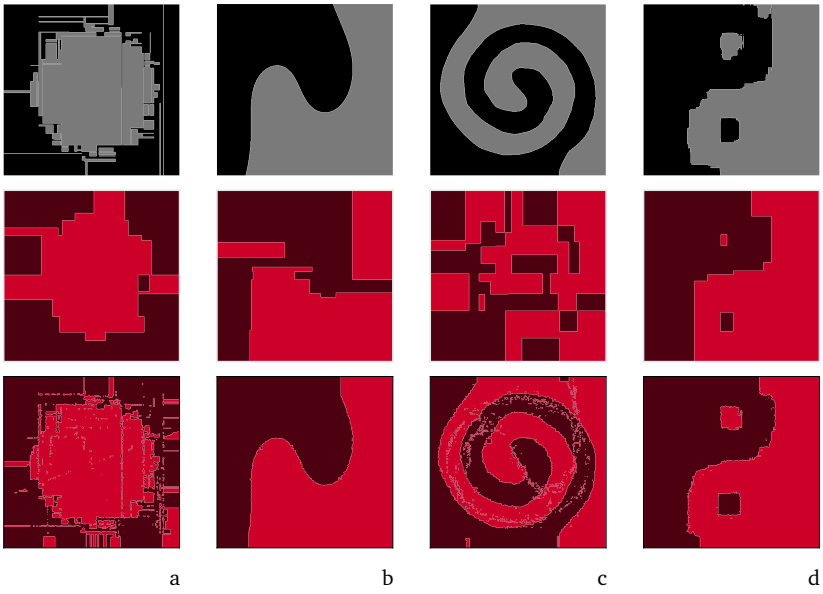


Fig. 3.7 From top to bottom, original decision functions and decision functions for copies based on incremental trees using a single iteration and 1000 iterations with a budget of 100 synthetic data points for (a) circles, (b) moons, (c) spirals and (d) yin-yang binary classification problems.

comparison. Overall, the learned decision functions display the general decision behavior of the target models shown in the top row. They correctly identify the general area where each of the classes are located. However, they fail to replicate the fine-grained form of the decision boundaries for each of the problems.

The copies resulting from the whole 1000 iterations are shown in the bottom row of Fig. 3.7 for the different datasets. In all cases, copies replicate the original decision boundaries to a high level of accuracy. This is particularly relevant because these boundaries are based in different classifiers and therefore display very different forms. Note in particular the cases of Fig. 3.7(b) and Fig. 3.7(c), which corresponds to a gaussian-kernel svm and a multilayer-perceptron, respectively. In both cases, the resulting decision functions have a smooth form. Yet, the copy incremental trees are able to recover these forms almost completely.

This is possible because the dual-pass approach exploits overfitting to ensure the copy fits every point in the synthetic dataset to perfection. To do so, it exploits access to the original predictions to force the

copy to focus on those regions where it outputs incorrect labels. Mistakes in one iteration are therefore rectified in the following iterations. In general, errors are located close to the decision boundary, so that this method ensures a good coverage around this region. At the same time, adding new samples ensures that we keep a balance between exploitation of the problematic areas and exploration of the input space. The trade-off between these two regimes is controlled by the memory storage parameter m, which defines the maximum number of iterations during which a given sample can be stored in memory.

Tuning the value of this parameter allows us to alternate between exploration and exploitation. The larger the value of m, the more times the copy is confronted with the same data. In contrast, the lower the value of m, the more new samples are allowed during each run. In general, the initial iterations should be oriented to exploring the decision behavior of the model freely. This is because we want the copy to capture the general behavior of $f_{\mathcal{O}}$. Hence, the value of m should be small or even 0 during this phase. As the number of iterations increases, however, we want the copy to refine its behavior by focusing on those regions where there exists a greater disagreement with the original model. Hence, towards the end of the process, the value of the memory storage parameter should be maximal, to ensure a good fit also for the hardest samples.

A good choice for the memory parameter is shown in Fig. 3.8, which displays how the value of m changes with increasing number of iterations. Here, we assume a sigmoid form for m. In the first half of the process, the memory parameter is kept close to zero. This allows for a proper exploration of the space to gather general knowledge about how the original model behaves. At approximately 300 iterations the value of m starts increasing. This is where exploitation starts. After having recovered the general form of the original decision function, we force the copy to focus on the details, while we keep searching the space for new data points. At around 700 iterations, the plot collapses. We are now only interested in those areas where the copy struggles the most. This ensures that we smoothly transition from learning of the general to focusing on the particular. As a result, we obtain a better fit also in the most difficult area,s so that the copy decision boundary fully resembles the original function.

The insights here presented correspond to preliminary experiments conducted using the dual-pass approach. Yet, while the results here discussed are promising, this strategy is still in need of further refinement. In the following chapters we restrict ourselves to discussing practical implementations of the single-pass approach exclusively.

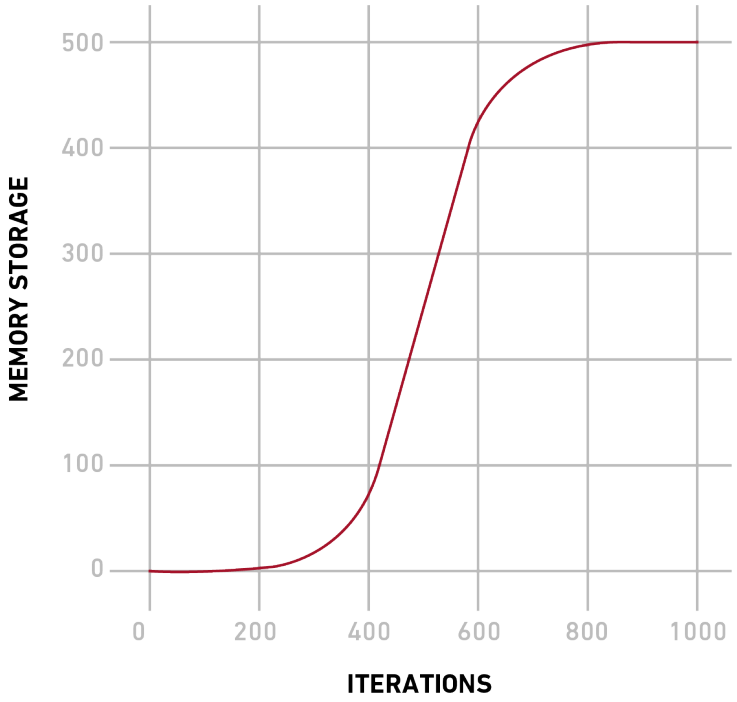


Fig. 3.8 Evolution of the memory storage parameter m with the number of iterations.

Lessons learned

- Copying refers to the problem of replicating the decision behavior of one classifier using another, in conditions where we have no access to the original training data nor to the target model's internals. This problem can be understood as the projection of a given decision function onto a new hypothesis space that defines a set of constraints for the copy.
- In practice, this projection involves a dual optimization. On the one hand, we need to optimize the set of points we use to build the copy. Given that the training data are unknown, copying requires that we generate a set synthetic data points to gain information about the decision behavior of the given model throughout the input space. On the other hand, we need to obtain the optimal set of copy parameters that fit these data.
- We present two different approaches to solving this dual optimization problem. In the simplest case, we decouple both optimizations and use a single iteration of an alternating projection scheme instead. We refer to this approach as the *single-pass*. Alternatively, we also sketch a solution for the *dual-pass approach*, where we exploit online capabilities to iteratively refine the copy decision boundary by guiding sampling towards those areas where errors are more substantial. This method, however, is still in need of further revision.
- In both cases, the problem of copying presents certain characteristics that we can exploit at our advantage. In particular, copies can be built without any regards for over-fitting, since the original model acts as a form of regularizer and we can generate as many synthetic samples as needed.
- In the following chapter we explore these and other specificities of copying in practice. We introduce a set of experimental metrics for copy validation and discuss additional insights of our proposed methodology in a set of well-known datasets.

Chapter 4

Experimental validation

4.1 Introduction

In the previous chapter we have developed the mathematical background for copying and discussed several qualitative insights that can be extracted from it. Here, we are interested in validating this methodology in practice. Indeed, while the theoretical basis for copying may be known by now, we need to also devise appropriate mechanisms for evaluation. This evaluation requires, among other things, that we establish clear and reliable metrics. In doing so, it is vital that we clearly identify all the different sources of error that may appear along the process. In the case of copying, these sources are related to our choice of copy hypothesis space, which is usually dependent on external constraints, our devised synthetic sample generation process, which depends on our knowledge of the data, and the interaction between these two.

This chapter begins with discussing the three most relevant error contributions to the copying loss. Namely, the capacity error, the coverage error and the interaction error, which are collectively defined by the fidelity error. Based on this knowledge, we present a series of performance metrics to approximate these errors in practice and report experimental results for these metrics over a heterogeneous set of problems. Throughout the next sections, we assume varying levels of knowledge over the different elements of the copying process and discuss how the available information can be exploited in each particular case.

4.2 Identifying the sources of error

We begin by studying the different sources of error that emerge during the copying process. In *Chapter 3* we introduced the expected loss for the copying problem through (3.10). We refer to this loss as the *fidelity error*. The fidelity error captures all the error of the copying process. Effectively, it measures the overall disagreement between the original decision function $f_{\mathcal{O}}$ and the copy $f_{\mathcal{C}}$. As we show below, this is a theoretical measure that can be divided into different parts.

For simplicity, let us particularize the fidelity error for the 0/1 loss and rewrite it as

$$R^{\mathcal{F}}(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z})) = \mathsf{P}\big(f_{\mathcal{C}}(\boldsymbol{z},\theta) \neq f_{\mathcal{O}}(\boldsymbol{z})\big) \tag{4.1}$$

$$= \mathbb{E}_{\boldsymbol{z}\sim P_Z}[\mathbb{I}_{\{\boldsymbol{z}: f_{\mathcal{C}}(\boldsymbol{z},\theta)\neq f_{\mathcal{O}}(\boldsymbol{z})\}}(\boldsymbol{z})] \tag{4.2}$$

for $\mathbb{I}$ the indicator function. Without loss of generality, let us also assume a binary classification problem so that $f_{\mathcal{O}}(\boldsymbol{z}) \in \{-1, +1\}$ and $f_{\mathcal{C}}(\boldsymbol{z},\theta) \in \{-1, +1\}$. The fidelity error for this case is given by

$$R^{\mathcal{F}}(f_{\mathcal{C}}(\boldsymbol{z},\theta), f_{\mathcal{O}}(\boldsymbol{z})) = \frac{1}{2}\int_{\boldsymbol{z}\sim P_Z} |f_{\mathcal{O}}(\boldsymbol{z}) - f_{\mathcal{C}}(\boldsymbol{z},\theta)| dP_Z \tag{4.3}$$

We can use the expression above to obtain an unbounded measure of the differences in the decision outputs for $f_{\mathcal{O}}$ and $f_{\mathcal{C}}$ throughout the whole attribute domain. As before, this would requires full access to the generating probability distribution P_Z. Moreover, to solve this integral directly, the form of both $f_{\mathcal{O}}$ and $f_{\mathcal{C}}$ should be explicitly known. Since this is not the case, *i.e.* we cannot compute the fidelity error directly, we develop the expression above to gain a better intuition on where this error arises from.

We begin by introducing the optimal copy model, $f^*_{\mathcal{C}}$, as defined in *Section 3.2.1*. In Fig. 4.1 we show a modified version of Fig. 3.1. The shaded region corresponds to the copy hypothesis space, which encompasses all the possible copy models of the same family. The optimal copy model $f^*_{\mathcal{C}}$ is that which is closest to the original $f_{\mathcal{O}}$ in this space. As opposed to the actual attainable copy $f_{\mathcal{C}}$, which is built on a finite synthetic set $\mathscr{Z}$ of size N, the optimal copy model assumes an infinite stream of synthetic data is available. Hence, we can refer to this model as the optimal projection of $f_{\mathcal{O}}$ in the new hypothesis space. Notably, when $f_{\mathcal{O}}$ belongs to $\mathcal{H}_{\mathcal{C}}$, *i.e.* when original and copy belong to the same family of models, then $f_{\mathcal{O}} = f^*_{\mathcal{C}}$. Note that this case has been previously discussed in *Chapter 1*, as displayed in Fig. 1.2(b).

We use the optimal copy model to expand (4.3) as follows

$$R^{\mathcal{F}} = \frac{1}{2} \int\limits_{\boldsymbol{z} \sim P_Z} \Big| f_{\mathcal{O}}(\boldsymbol{z}) + f^*_{\mathcal{C}}(\boldsymbol{z}, \theta) - f^*_{\mathcal{C}}(\boldsymbol{z}, \theta) - f_{\mathcal{C}}(\boldsymbol{z}, \theta) \Big| dP_Z \tag{4.4}$$

$$= \frac{1}{4} \int\limits_{\boldsymbol{z} \sim P_Z} \Big(f_{\mathcal{O}}(\boldsymbol{z}) - f^*_{\mathcal{C}}(z) + f^*_{\mathcal{C}}(\boldsymbol{z}, \theta) - f_{\mathcal{C}}(\boldsymbol{z}, \theta) \Big)^2 dP_Z \tag{4.5}$$

$$= \frac{1}{4} \int\limits_{\boldsymbol{z} \sim P_Z} \Big(f_{\mathcal{O}}(\boldsymbol{z}) - f^*_{\mathcal{C}}(\boldsymbol{z}, \theta) \Big)^2 dP_Z + \tag{4.6}$$

$$\frac{1}{4} \int\limits_{\boldsymbol{z} \sim P_Z} \Big(f^*_{\mathcal{C}}(\boldsymbol{z}, \theta) - f_{\mathcal{C}}(\boldsymbol{z}, \theta) \Big)^2 dP_Z + \tag{4.7}$$

$$\frac{1}{2} \int\limits_{\boldsymbol{z} \sim P_Z} \Big(f_{\mathcal{O}}(\boldsymbol{z}) - f^*_{\mathcal{C}}(\boldsymbol{z}, \theta) \Big) \Big(f^*_{\mathcal{C}}(\boldsymbol{z}, \theta) - f_{\mathcal{C}}(\boldsymbol{z}, \theta) \Big) dP_Z \tag{4.8}$$

where we drop the square root in the first expression, thanks to the co-domain of the functions involved. Note that, for the sake of simplicity, we here use the shorter form $R^{\mathcal{F}}$ and drop the explicit dependence on both $f_{\mathcal{C}}(\boldsymbol{z}, \theta)$ and $f_{\mathcal{O}}(\boldsymbol{z})$.

The expression above contains three different contributions to the fidelity error. The first term appears in (4.6) and corresponds to the error we incur when replacing the original model $f_{\mathcal{O}}$ with the optimal copy model $f^*_{\mathcal{C}}$, the theoretically attainable copy. This error quantifies the capacity mismatch between the copy hypothesis space and the original hypothesis space. As mentioned previously in *Section 3.3.1* we refer to this mismatch as the *capacity error*, $R_{\mathcal{C}}$. In Fig. 4.1 it is displayed as the distance between $f_{\mathcal{O}}$ and $f^*_{\mathcal{C}}$. When $f_{\mathcal{O}} \in \mathcal{H}_{\mathcal{C}}$, this distance is zero.

Mathematically, we can define the capacity error as

$$R_{\mathcal{C}} = \frac{1}{2} \int_{\boldsymbol{z} \in P_Z} \Big| f_{\mathcal{O}}(\boldsymbol{z}) - f^*_{\mathcal{C}}(\boldsymbol{z}, \theta) \Big| dP_Z. \tag{4.9}$$

The second contribution to the fidelity error is shown in (4.7) and arises from the the fact that we are limited to a finite number of synthetic samples. For a given copy hypothesis space, this error measures the disagreement between the decision boundary output by the optimal copy model and that obtained when building a copy based on N samples. We call this source of error *coverage error*, $R_{\mathcal{CV}}$. Graphically, we can depict it as the distance between the optimal copy model $f^*_{\mathcal{C}}$ and the copy $f_{\mathcal{C}}$, as show in Fig. 4.1. We define the coverage error as

$$R_{\mathcal{CV}} = \frac{1}{2} \int_{\boldsymbol{z} \in P_Z} \Big| f^*_{\mathcal{C}}(\boldsymbol{z}) - f_{\mathcal{C}}(\boldsymbol{z}, \theta) \Big| dP_Z \tag{4.10}$$

Finally, there is also a third error term that appears in (4.8). This error accounts for a certain coupling

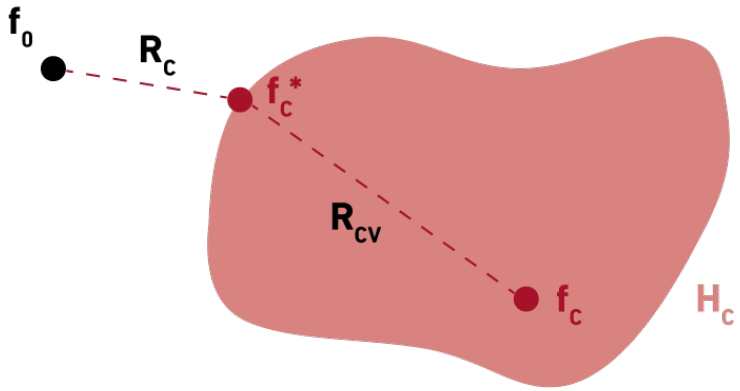


Fig. 4.1 Original, copy and optimal copy models in relation to the copy hypothesis space. Both the capacity and the coverage errors are displayed in terms of the distance they refer to in this space.

between the capacity and coverage errors. We refer to this term as the *interaction error*, $R_{\mathcal{I}}$, defined as

$$R_{\mathcal{I}} = \frac{1}{2} \int_{\boldsymbol{z} \in P_Z} \Big(f_{\mathcal{O}}(\boldsymbol{z}) - f_{\mathcal{C}}^{\infty}(\boldsymbol{z}, \theta) \Big) \Big(f_{\mathcal{C}}^{\infty}(\boldsymbol{z}, \theta) - f_{\mathcal{C}}(\boldsymbol{z}, \theta) \Big) dP_Z \tag{4.11}$$

The interaction error has three main properties. First, the only cases in which it is non-zero is when $f_{\mathcal{O}}(\boldsymbol{z})$ agrees with $f_{\mathcal{C}}(\boldsymbol{z}, \theta)$, while both differ from $f_{\mathcal{C}}^{*}(\boldsymbol{z}, \theta)$. Whenever this happens, the net effect of the interaction error is negative, meaning that it acts in the direction of decreasing the total error. This means that the interaction error is only relevant when we select a copy hypothesis space very distant from the family of models the original belongs to and at the same time we conduct a poor synthetic sample generation. Under these circumstances, the interaction term corrects errors due to a mismatch in capacity when the coverage error is also high. Second, as the number of synthetic samples increases, $N \to \infty$, the copy approaches the optimal model, $f_{\mathcal{C}} \to f_{\mathcal{C}}^{*}$. This reduces the coverage error contribution and, consequently, also the value of the interaction term. Finally, note that we would generally choose a large capacity copy, so that ideally either $f_{\mathcal{O}}$ is in the copy hypothesis space or $f_{\mathcal{C}}^{*}$ and $f_{\mathcal{O}}$ are close. As a result, the capacity and interaction terms are usually low and the largest contribution to the fidelity error is the coverage term.

Altogether, the three error terms above represent all the different sources of error that appear during the copying process. We can express the fidelity error in terms of the capacity, coverage and interaction error contributions as

$$R^{\mathcal{F}}(f_{\mathcal{C}}(\boldsymbol{z}, \theta), f_{\mathcal{O}}(\boldsymbol{z})) = R_{\mathcal{C}} + R_{\mathcal{CV}} + R_{\mathcal{I}}. \tag{4.12}$$

A main issue with copy evaluation is that, although the different sources of error are known and understood, they cannot all be measured in practice. In the general setting, for example, the synthetic dataset is finite. Hence, we do not have access to the optimal copy model. Measuring the capacity and coverage errors can therefore be tricky. Plus, even if we did have access to this model, we would still need to devise a reliable evaluation framework. In the next section we present a series of performance metrics

that can be used to validate the copying process under different assumptions.

4.3 Performance metrics

When evaluating copies in practice, we may ask questions of the form: *"what does the performance on a synthetic validation set tell us about the generalization of the copy?"*, *"does the copy have enough capacity to replicate the decision function?"* or, more generally, *"what metrics should we use to evaluate copies in terms of the available information?"*. In what follows we introduce a set of definitions aimed at answering these questions. First, we propose a measurable approximation to the empirical fidelity error. Further, we define a set of performance metrics that serve as sanity checks when the original accuracy or the original dataset or both are accessible.

Empirical fidelity error in practice

The empirical fidelity error corresponds to the empirical risk in the copying setting, as defined in (3.11). As we did before for the overall fidelity error, we can particularize the empirical fidelity error for the 0/1 loss as

$$R_{emp}^{\mathcal{F},\mathscr{Z}} = \frac{1}{N}\sum_{j=1}^{N} \mathbb{I}[f_{\mathcal{O}}(\boldsymbol{z}_j) \neq f_{\mathcal{C}}(\boldsymbol{z}_j)] \tag{4.13}$$

where, again, $\mathbb{I}$ refers to the the indicator function. Note the chosen notation. We specify the super-index $\mathscr{Z}$ to highlight the fact that this error is measured over the generated synthetic dataset.

The empirical fidelity error as above defined measures the ratio of the N synthetic data points that are equally classified by the model $f_{\mathcal{O}}$ and the copy $f_{\mathcal{C}}$. How well this error represents the total error of the copy depends, among other things, on the specific choice of $\mathscr{Z}$, *i.e.* on the specific choice of P_Z. Following the discussion in *Section 3.3.1*, we observe that, in resorting to Monte Carlo integration to explore the original decision behavior we necessarily incur in an approximation error. This error depends on how representative the set $\mathscr{Z}$ is of the original model's behavior. This is, on the coverage error R_{CV}. As a result, a low $R_{emp}^{\mathcal{F},\mathscr{Z}}$ is no absolute guarantee of a good copy. For this value to be a valid assessment of the total error, the synthetic dataset must be large enough to ensure coverage of the input space and the volume imbalance effect needs to be controlled for. If these conditions are met and $R_{emp}^{\mathcal{F},\mathscr{Z}}$ is still different from zero, the theoretical sources of error should be considered. The error either arises from a low capacity of the copy (large capacity error) or because there is a mismatch between the optimal achievable model and the one obtained (large coverage error). In the first case, we should change the model to a larger capacity option. In the second, more samples or better quality samples may be needed. In this sense, we stress that although we cannot measure the coverage, capacity and interaction error directly, they all contribute to our estimation of the fidelity error through $R_{emp}^{\mathcal{F},\mathscr{Z}}$.

In the most general setting, where we assume no access to the training data, the empirical fidelity error is the only metric we can report when building a copy. Conversely, in cases where the constraints of the copying scenario are relaxed and the training data $\mathscr{D}$ are accessible, we could also evaluate the empirical fidelity error over this set as

$$R_{emp}^{\mathcal{F},\mathscr{D}} = \frac{1}{M}\sum_{i=1}^{M} \mathbb{I}[f_{\mathcal{O}}(\boldsymbol{x}_i) \neq f_{\mathcal{C}}(\boldsymbol{x}_i)]. \tag{4.14}$$

for M the total number of data points in that set. Again, the expression above quantifies the ratio of instances over which the predictions output by $f_{\mathcal{O}}$ and $f_{\mathcal{C}}$ disagree. But this time, these instances are those of the original training data.

Usually, the empirical fidelity errors $R_{emp}^{\mathcal{F},\mathscr{D}}$ and $R_{emp}^{\mathcal{F},\mathscr{Z}}$, computed over the two datasets, yield very different values. This difference arises mainly from the mismatch between the probability density functions P and P_Z. Depending on the application, the overlap between these two distribution may be smaller or larger.

Copy accuracy

In cases where the original training data are accessible, we can introduce an additional safety check and evaluate the copy generalization performance over $\mathscr{D}$. We define the *copy accuracy*, $\mathcal{A}_{\mathcal{C}}$, as the ratio of original instances that the copy classifies correctly. It can be expressed as follows

$$\mathcal{A}_{\mathcal{C}} = \frac{1}{M}\sum_{i=1}^{M} \mathbb{I}[t_i = f_{\mathcal{C}}(\boldsymbol{x}_i)], \tag{4.15}$$

for $t_i \in \mathscr{T}$ the true labels.

Additionally, when the performance of the original model is also known, we can explore this information to our advantage. We refer to this value as the *original accuracy*, $\mathcal{A}_{\mathcal{O}}$. Assuming the copying process is conducted successfully, the original accuracy is an upper bound to the performance the copy can achieve in the original data environment $\mathscr{D}$. Since we assume all the loss to be captured by the fidelity error, we can write $\mathcal{A}_{\mathcal{C}}$ in terms of the original accuracy as

$$A_{\mathcal{C}} = A_{\mathcal{O}}(1 - R^{\mathcal{F}})$$

According to this expression, if we are able to achieve a perfect copy, *i.e.* fidelity error equal to zero, then it follows that $A_{\mathcal{C}} = A_{\mathcal{O}}$. Unfortunately, this is not usually the case. On the contrary, there is generally a certain error, if negligible. Moreover, since the fidelity error is not directly measurable, we need to substitute it with the empirical fidelity error. In situations where the original training data are not known but the original accuracy is, we can obtain an estimation of the copy accuracy as

$$\widehat{\mathcal{A}_{\mathcal{C}}} = \mathcal{A}_{\mathcal{O}}(1 - R_{emp}^{\mathcal{F},\mathscr{Z}}) \tag{4.16}$$

We refer to this value as the *estimated copy accuracy*. As ever, this estimation becomes more trustworthy, the smaller all the error contributions are. In particular, note that in cases where there is little, or none, overlap between P and P_Z, this value may not provide a good approximation to $\mathcal{A}_C$.

In what follows we validate copies in a practical setting and evaluate their performance using the metrics above. In all cases, we report one metric or the other in terms of the level of knowledge that is assumed of the system. These experiments allow us to study what the advantages and shortcomings of copying are in practice and how the latter may be overcome.

4.4 UCI classification

We use 60 datasets from the UCI Machine Learning Repository database [75]. We do so by following [81], who present a comparison of 122 datasets from this source. We refer the reader to this paper for a specific description of initial data selection and preprocessing. We discard 62 datasets due to several reasons: we do not consider those datasets which contain less than 100 samples and remove those with at least one class label with a frequency smaller than 10% of the total size of the dataset. We also require the number of inputs to be greater than double the number of attributes. Among the selected datasets 42 correspond to binary classification problems and 18 are multiclass.

For any given dataset, we train a model using a generic machine learning pipeline. Since this model is only used as a baseline for copying, we are not interested in optimizing performance at this step. Instead, we study different model architectures and forms to have a better understanding of how copying works under different conditions. Even so, we discuss initial model training and accuracy. The real experiment, however, begins once these models are trained. For each case, we build copies based on three different model families and discuss the characteristics of each case. Altogether, we build 180 copies with the methods described below.

4.4.1 Experimental set up

Given the raw data for the 60 UCI dataset, we convert nominal attributes to numerical and re-scale variables to zero mean and unit variance. We split data into stratified 80/20 training and test sets. We use 6 state-of-the-art classification algorithms, including adaboost (*adaboost*), artificial neural networks (*ann*), random forest (*rfc*), linear SVM (*linear_svm*), a radial basis kernel SVM (*rbf_svm*) and gradient-boosted trees (*xgboost*). We use standard methods from Python's *scikit-learn* module to train the original model for the first 5 algorithms and *xgboost* library to fit the gradient-boosting trees. To avoid bias regarding the choice of algorithm for each particular problem, we sort datasets in alphabetical order, group them in sets of 10 and randomly assign a classifier to each group. A full description of the 60 datasets, including general data attributes and their assigned classifier, can be found in Tables 4.1 and

4.2.

To further avoid any interference from our part, we build a generic pipeline and train all models using a cross-validated grid-search over a fixed parameter grid. We do so using a 3-fold cross-validation. Three classifiers learn decision functions that exclude at least one of the class labels. This occurs for *pittsburg-bridges-REL-L*, for which only two of the three classes are learned, and *planning* and *statlog-australian-credit*, for which a single class label is assigned to all data points. Besides, because we use a fixed pipeline, not all models yield an optimal performance. See, for example, the case of *echocardiogram*, where original accuracy is equal to 0.3. We keep this result for two reasons. First, to ensure we get an optimal understanding of copy performance under different circumstances, we want the experimental setup to be as agnostic as possible and hence the random pairing of models and datasets. Second, it reinforces an important idea: a copy can only be as good as the model it aims to replicate. Or in the other words, the baseline for the copy performance is the original model performance. Non-optimal models lead to poorly performing copies. We stress, nonetheless, that in a real setting one would be interested in copying only those models that perform reasonably well.

We generate balanced synthetic sets by sampling the input space of each problem using a uniform distribution an labelling the resulting samples according to the predictions of the trained classifiers. We use synthetic sets composed of $1e6$ random samples. We identify three cases of volume imbalance: *congressional-voting*, *ilpd-indian-liver* and *statlog-image*. In all cases, despite the training data being balanced with respect to class distribution, we only recover a small fraction of samples for one or more of the labels. As previously mentioned, this could lead to sub-optimal results, given that the copy tends to wrongly classify points that belong to the subsampled classes. Imposing that the synthetic dataset be balanced mitigates this issue to a great extent and ensures that the copy treats all labels equally.

To evaluate the impact of heuristics, we assay different copy model hypotheses. We use decision trees because they provide a rule-based decision path that is generally interpretable[1], logistic regression because it is an easily understandable linear model and random forest as an example of a more intricate bagging method. We copy using no cross-validation or hyper-parameter tuning: trees are grown until each leaf contains a single sample and neural networks and boosting methods are trained with no regard for generalization. For validation purposes, we run each experiment 100 times and report averages over all repetitions for the true and the estimated copy accuracy. We also report the mean empirical fidelity error measured over both the training and the synthetic data.

4.4.2 Results

In Fig. 4.2 the averaged performance metrics for all datasets are plotted against each other. In particular, Fig. 4.2(a), Fig. 4.2(b) and Fig. 4.2(c) show the distribution of the copy accuracy $\mathcal{A}_{\mathcal{C}}$ against the

[1]Note the term *generally* here. While decision trees are widely accepted as interpretable models, this assertion does not always hold. In the following chapters we explore this issue in greater depth.

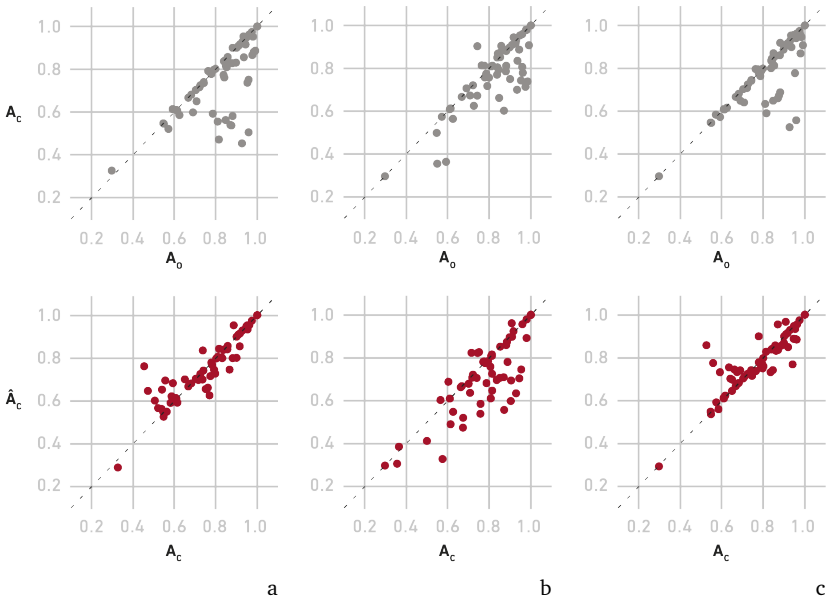


Fig. 4.2 From top to bottom, distribution of average copy accuracy against original accuracy and distribution of average estimated copy accuracy against average true copy accuracy for all datasets and for copies based on (a) decision trees, (b) logistic regression and (c) random forest.

original accuracy $\mathcal{A}_{\mathcal{O}}$ (top) and the estimated copy accuracy $\widehat{\mathcal{A}_{\mathcal{C}}}$ (bottom) for copies based on decision trees (*decision_tree*), logistic regression (*logistic_regression*) and random forest (*rfc*), respectively. The distribution of results for both *decision_tree* and *rfc* are mostly scattered around the main diagonal, whereas copies based on *logistic_regression* show a greater dispersion; especially when comparing $\mathcal{A}_{\mathcal{C}}$ to $\widehat{\mathcal{A}_{\mathcal{C}}}$. In general, the value of $\widehat{\mathcal{A}_{\mathcal{C}}}$ is smaller than that of $\mathcal{A}_{\mathcal{C}}$, which means that the empirical fidelity error tends to overestimate the real error. This is partly due to the difference between P and P_Z. When measuring $R^{\mathscr{Z}}_{\mathcal{F}}$, we evaluate performance on the space defined by P_Z, which is usually bigger than that of P. As a result, we penalize the copy for errors in regions where there might not be any actual training data.

The complete summary of results for all problems and copy algorithms is shown in Tables 4.1 and 4.2, where all the relevant results are highlighted. Blank spaces correspond to cases where models learn a single class label. In most problems, results show the ability of copies to replicate the target decision behaviour. Overall, copy accuracy is competitive for the proposed synthetic dataset size and the estimated copy accuracy provides a reliable approximation to the accuracy of the copy in the training data. The empirical fidelity error over the synthetic dataset generally yields values close to 0, which indicates that copies are correctly built. And, while there are some exceptions, in general the empirical fidelity error over the training dataset stays reasonably low too.

Notably, there are several datasets where there is no degradation when using a *logistic_regression* to copy higher capacity models such as *ann* or *xgboost*. This is the case, for example, for *breast-cancer-wisc* and *wine*, where $\mathcal{A}_{\mathcal{C}}$ is reasonably close to $\mathcal{A}_{\mathcal{O}}$, even while the logistic model can only learn linear

Table 4.1 Experimental results for the first 30 UCI datasets.

Dataset	$\lvert\mathcal{T}\rvert$	M	d	Model	$\mathcal{A}_\mathcal{O}$	decision_tree			logistic_regression			rfc		
						$\mathcal{A}_\mathcal{C}$	$\widehat{\mathcal{A}}_\mathcal{C}$	$R^{\mathcal{D}}_{\mathcal{F}}$	$\mathcal{A}_\mathcal{C}$	$\widehat{\mathcal{A}}_\mathcal{C}$	$R^{\mathcal{D}}_{\mathcal{F}}$	$\mathcal{A}_\mathcal{C}$	$\widehat{\mathcal{A}}_\mathcal{C}$	$R^{\mathcal{D}}_{\mathcal{F}}$
abalone	3	3341	8	*adaboost*	0.57	0.52 ± 0.02	0.565 ± 0.001	0.27 ± 0.02	0.57 ± 0.00	0.327 ± 0.001	0.41 ± 0.00	0.58 ± 0.01	0.559 ± 0.011	0.27 ± 0.01
acute-inflammation	2	96	6	*adaboost*	1	1.00 ± 0.00	1.000 ± 0.000	0.00 ± 0.00	1.00 ± 0.00	1.000 ± 0.000	0.00 ± 0.00	1.00 ± 0.00	1.000 ± 0.000	0.00 ± 0.00
acute-nephritis	2	96	6	*adaboost*	1	1.00 ± 0.00	1.000 ± 0.000	0.00 ± 0.00	1.00 ± 0.00	0.999 ± 0.000	0.00 ± 0.00	1.00 ± 0.00	1.000 ± 0.000	0.00 ± 0.00
bank	2	3616	16	*adaboost*	0.85	0.82 ± 0.03	0.843 ± 0.001	0.10 ± 0.03	0.87 ± 0.00	0.556 ± 0.001	0.12 ± 0.00	0.87 ± 0.00	0.835 ± 0.002	0.07 ± 0.00
blood	2	598	4	*adaboost*	0.71	0.65 ± 0.04	0.700 ± 0.001	0.14 ± 0.05	0.67 ± 0.00	0.519 ± 0.001	0.38 ± 0.00	0.64 ± 0.02	0.702 ± 0.023	0.16 ± 0.04
breast-cancer	2	228	9	*adaboost*	0.74	0.74 ± 0.01	0.741 ± 0.000	0.00 ± 0.01	0.67 ± 0.00	0.473 ± 0.001	0.31 ± 0.00	0.74 ± 0.00	0.741 ± 0.000	0.00 ± 0.00
breast-cancer-wisc	2	559	9	*adaboost*	**0.93**	0.93 ± 0.00	0.929 ± 0.000	0.00 ± 0.00	**0.93 ± 0.00**	0.633 ± 0.001	0.06 ± 0.00	0.93 ± 0.00	0.929 ± 0.000	0.00 ± 0.00
breast-cancer-wisc-diag	2	455	30	*adaboost*	0.95	0.95 ± 0.00	0.947 ± 0.000	0.00 ± 0.00	0.95 ± 0.00	0.744 ± 0.000	0.04 ± 0.00	0.95 ± 0.00	0.947 ± 0.000	0.00 ± 0.00
breast-cancer-wisc-prog	2	158	33	*adaboost*	0.73	0.72 ± 0.00	0.725 ± 0.000	0.00 ± 0.00	0.62 ± 0.00	0.547 ± 0.001	0.35 ± 0.00	0.72 ± 0.00	0.725 ± 0.000	0.00 ± 0.00
breast-tissue	6	84	9	*adaboost*	0.59	0.61 ± 0.02	0.591 ± 0.000	0.16 ± 0.04	0.36 ± 0.00	0.384 ± 0.001	0.59 ± 0.00	0.57 ± 0.02	0.591 ± 0.022	0.18 ± 0.01
chess-krvkp	2	2556	36	*ann*	0.99	0.89 ± 0.02	0.953 ± 0.001	0.11 ± 0.02	0.91 ± 0.00	0.960 ± 0.000	0.10 ± 0.00	0.91 ± 0.01	0.967 ± 0.009	0.09 ± 0.01
congressional-voting	2	348	16	*ann*	0.61	0.61 ± 0.00	0.609 ± 0.000	0.00 ± 0.00	0.61 ± 0.00	0.609 ± 0.000	0.00 ± 0.00	0.61 ± 0.00	0.609 ± 0.000	0.00 ± 0.00
conn-bench-sonar	2	166	60	*ann*	0.88	0.58 ± 0.08	0.590 ± 0.001	0.40 ± 0.07	0.81 ± 0.01	0.808 ± 0.000	0.12 ± 0.01	0.69 ± 0.03	0.741 ± 0.033	0.21 ± 0.04
connect-4	2	54045	42	*ann*	0.87	0.54 ± 0.04	0.652 ± 0.001	0.46 ± 0.04	0.60 ± 0.00	0.687 ± 0.000	0.39 ± 0.00	0.66 ± 0.00	0.744 ± 0.003	0.32 ± 0.00
contract	3	1178	9	*ann*	0.55	0.55 ± 0.01	0.525 ± 0.001	0.07 ± 0.02	0.50 ± 0.00	0.411 ± 0.000	0.40 ± 0.00	0.55 ± 0.00	0.536 ± 0.004	0.03 ± 0.01
credit-approval	2	552	15	*ann*	0.79	0.80 ± 0.02	0.727 ± 0.001	0.10 ± 0.02	0.72 ± 0.00	0.707 ± 0.000	0.19 ± 0.00	0.79 ± 0.01	0.753 ± 0.007	0.03 ± 0.01
cylinder-bands	2	409	35	*ann*	0.69	0.60 ± 0.05	0.608 ± 0.001	0.37 ± 0.05	0.71 ± 0.00	0.635 ± 0.000	0.25 ± 0.00	0.65 ± 0.01	0.645 ± 0.012	0.35 ± 0.01
echocardiogram	2	104	10	*ann*	**0.3**	0.33 ± 0.04	0.288 ± 0.000	0.05 ± 0.03	0.30 ± 0.00	0.296 ± 0.000	0.00 ± 0.00	0.30 ± 0.00	0.292 ± 0.000	0.00 ± 0.00
energy-y1	3	614	8	*ann*	**0.96**	**0.96 ± 0.01**	0.954 ± 0.001	0.00 ± 0.01	**0.78 ± 0.00**	0.774 ± 0.001	0.23 ± 0.00	**0.96 ± 0.00**	0.955 ± 0.004	0.00 ± 0.00
energy-y2	3	614	8	*ann*	0.84	0.84 ± 0.00	0.840 ± 0.001	0.00 ± 0.00	0.79 ± 0.00	0.784 ± 0.001	0.09 ± 0.00	0.84 ± 0.01	0.841 ± 0.007	0.01 ± 0.00
fertility	2	80	9	*rfc*	0.9	0.90 ± 0.00	0.899 ± 0.000	0.00 ± 0.00	0.75 ± 0.00	0.826 ± 0.000	0.15 ± 0.00	0.90 ± 0.00	0.899 ± 0.000	0.00 ± 0.00
haberman-survival	2	244	3	*rfc*	0.61	0.61 ± 0.01	0.613 ± 0.000	0.01 ± 0.01	0.61 ± 0.00	0.489 ± 0.001	0.26 ± 0.00	0.61 ± 0.01	0.612 ± 0.007	0.00 ± 0.01
heart-hungarian	2	235	12	*rfc*	0.76	0.79 ± 0.04	0.745 ± 0.001	0.07 ± 0.03	0.81 ± 0.00	0.646 ± 0.001	0.05 ± 0.00	0.80 ± 0.00	0.750 ± 0.000	0.03 ± 0.00
hepatitis	2	124	19	*rfc*	0.74	0.74 ± 0.05	0.699 ± 0.001	0.05 ± 0.04	0.90 ± 0.00	0.597 ± 0.000	0.16 ± 0.00	0.75 ± 0.01	0.715 ± 0.009	0.00 ± 0.01
ilpd-indian-liver	2	466	9	*rfc*	0.62	0.59 ± 0.02	0.621 ± 0.001	0.36 ± 0.02	0.56 ± 0.00	0.602 ± 0.000	0.47 ± 0.00	0.62 ± 0.01	0.622 ± 0.009	0.37 ± 0.01
ionosphere	2	280	33	*rfc*	0.94	0.92 ± 0.03	0.854 ± 0.001	0.07 ± 0.03	0.89 ± 0.00	0.779 ± 0.000	0.11 ± 0.00	0.95 ± 0.01	0.887 ± 0.014	0.05 ± 0.01
iris	3	120	4	*rfc*	0.93	**0.95 ± 0.02**	0.933 ± 0.000	0.02 ± 0.02	**0.70 ± 0.00**	0.678 ± 0.001	0.30 ± 0.00	**0.95 ± 0.02**	0.933 ± 0.016	0.02 ± 0.02
magic	2	15216	10	*rfc*	0.88	0.83 ± 0.01	0.802 ± 0.001	0.10 ± 0.01	0.79 ± 0.00	0.681 ± 0.001	0.17 ± 0.00	0.86 ± 0.00	0.832 ± 0.002	0.06 ± 0.00
mammographic	2	768	5	*rfc*	0.8	0.80 ± 0.00	0.797 ± 0.001	0.01 ± 0.00	0.76 ± 0.00	0.538 ± 0.001	0.19 ± 0.00	0.80 ± 0.00	0.797 ± 0.004	0.01 ± 0.00
miniboone	2	104051	50	*rfc*	0.94	0.86 ± 0.01	0.840 ± 0.001	0.12 ± 0.01	0.84 ± 0.00	0.695 ± 0.001	0.15 ± 0.00	0.90 ± 0.00	0.868 ± 0.001	0.07 ± 0.00

Dataset	$\lvert\mathcal{S}\rvert$	M	d	Model	$\mathcal{A}_{\mathcal{O}}$	decision_tree			logistic_regression			rfc		
						$\mathcal{A}_{\mathcal{C}}$	$\hat{\mathcal{A}}_{\mathcal{C}}$	$R^2_{\mathcal{F}}$	$\mathcal{A}_{\mathcal{C}}$	$\hat{\mathcal{A}}_{\mathcal{C}}$	$R^2_{\mathcal{F}}$	$\mathcal{A}_{\mathcal{C}}$	$\hat{\mathcal{A}}_{\mathcal{C}}$	$R^2_{\mathcal{F}}$
molec-biol-splice	3	2552	60	*linear_svm*	0.84	0.77 ± 0.01	0.715 ± 0.001	0.17 ± 0.01	0.77 ± 0.00	0.780 ± 0.001	0.14 ± 0.00	0.80 ± 0.00	0.759 ± 0.004	0.15 ± 0.00
mushroom	2	6499	21	*linear_svm*	0.98	0.95 ± 0.02	0.953 ± 0.001	0.03 ± 0.02	0.98 ± 0.00	0.978 ± 0.000	0.00 ± 0.00	0.87 ± 0.05	0.955 ± 0.054	0.11 ± 0.06
musk-1	2	380	166	*linear_svm*	**0.88**	0.54 ± 0.04	0.562 ± 0.001	0.46 ± 0.05	0.88 ± 0.00	0.873 ± 0.000	0.01 ± 0.00	**0.67 ± 0.06**	0.732 ± 0.058	0.32 ± 0.04
musk-2	2	5278	166	*linear_svm*	**0.96**	0.50 ± 0.05	0.601 ± 0.001	0.50 ± 0.05	0.96 ± 0.00	0.956 ± 0.000	0.00 ± 0.00	**0.56 ± 0.04**	0.775 ± 0.038	0.44 ± 0.04
oocytes_merl_nucl_4d	2	817	41	*linear_svm*	0.82	0.47 ± 0.06	0.646 ± 0.001	0.52 ± 0.06	0.81 ± 0.00	0.814 ± 0.000	0.00 ± 0.00	0.59 ± 0.03	0.732 ± 0.026	0.38 ± 0.03
oocytes_tris_nucl_2f	2	729	25	*linear_svm*	0.81	0.56 ± 0.05	0.695 ± 0.001	0.43 ± 0.07	0.81 ± 0.00	0.808 ± 0.000	0.00 ± 0.00	0.63 ± 0.03	0.754 ± 0.030	0.32 ± 0.03
parkinsons	2	156	22	*linear_svm*	0.9	0.83 ± 0.04	0.799 ± 0.001	0.11 ± 0.06	0.90 ± 0.00	0.895 ± 0.000	0.00 ± 0.00	0.89 ± 0.01	0.856 ± 0.013	0.02 ± 0.02
pima	2	614	8	*linear_svm*	0.72	0.72 ± 0.01	0.697 ± 0.001	0.04 ± 0.01	0.72 ± 0.00	0.720 ± 0.000	0.00 ± 0.00	0.71 ± 0.01	0.710 ± 0.005	0.02 ± 0.01
pitts-bridges-MATERIAL	3	84	7	*linear_svm*	0.91	0.91 ± 0.00	0.909 ± 0.000	0.00 ± 0.00	0.91 ± 0.00	0.897 ± 0.000	0.00 ± 0.00	0.91 ± 0.00	0.909 ± 0.000	0.00 ± 0.00
pitts-bridges-REL-L	3	82	7	*linear_svm*	**0.67**	0.67 ± 0.00	0.667 ± 0.000	0.00 ± 0.00	0.67 ± 0.00	0.667 ± 0.000	0.00 ± 0.00	0.67 ± 0.00	0.667 ± 0.000	0.00 ± 0.00
pitts-bridges-T-OR-D	2	81	7	*rbf_svm*	0.86	0.86 ± 0.00	0.852 ± 0.000	0.00 ± 0.00	0.90 ± 0.00	0.693 ± 0.001	0.24 ± 0.00	0.86 ± 0.00	0.857 ± 0.000	0.00 ± 0.00
planning	2	145	12	*rbf_svm*	**0.7**	0.70 ± 0.00	0.703 ± 0.000	0.00 ± 0.00	-	-	-	0.70 ± 0.00	0.703 ± 0.000	0.00 ± 0.00
ringnorm	2	5920	18	*rbf_svm*	0.98	0.88 ± 0.01	0.799 ± 0.001	0.11 ± 0.01	0.74 ± 0.00	0.705 ± 0.000	0.26 ± 0.00	0.95 ± 0.00	0.888 ± 0.002	0.05 ± 0.00
seeds	3	168	7	*rbf_svm*	0.88	0.90 ± 0.02	0.801 ± 0.001	0.06 ± 0.03	0.88 ± 0.00	0.858 ± 0.001	0.00 ± 0.00	0.92 ± 0.02	0.849 ± 0.016	0.04 ± 0.02
spambase	2	3680	57	*rbf_svm*	0.93	0.45 ± 0.10	0.761 ± 0.001	0.55 ± 0.11	0.92 ± 0.00	0.923 ± 0.000	0.02 ± 0.00	0.53 ± 0.08	0.858 ± 0.082	0.48 ± 0.09
statlog-australian-credit	2	552	14	*rbf_svm*	**0.68**	0.68 ± 0.00	0.681 ± 0.000	0.00 ± 0.00	-	-	-	0.68 ± 0.00	0.681 ± 0.000	0.00 ± 0.00
statlog-german-credit	2	800	24	*rbf_svm*	0.79	0.59 ± 0.03	0.682 ± 0.001	0.36 ± 0.04	0.78 ± 0.00	0.782 ± 0.000	0.02 ± 0.00	0.76 ± 0.01	0.737 ± 0.012	0.07 ± 0.01
statlog-heart	2	216	13	*rbf_svm*	0.85	0.81 ± 0.02	0.805 ± 0.001	0.05 ± 0.01	0.85 ± 0.00	0.851 ± 0.000	0.00 ± 0.00	0.81 ± 0.01	0.827 ± 0.006	0.04 ± 0.01
statlog-image	7	1848	18	*rbf_svm*	0.95	0.74 ± 0.01	0.835 ± 0.001	0.25 ± 0.02	0.74 ± 0.00	0.822 ± 0.001	0.25 ± 0.00	0.78 ± 0.01	0.899 ± 0.008	0.21 ± 0.01
statlog-vehicle	4	676	18	*rbf_svm*	0.85	0.56 ± 0.05	0.549 ± 0.001	0.40 ± 0.04	0.66 ± 0.00	0.662 ± 0.000	0.33 ± 0.00	0.65 ± 0.02	0.697 ± 0.019	0.29 ± 0.02
synthetic-control	6	480	60	*xgboost*	0.96	0.75 ± 0.02	0.655 ± 0.001	0.23 ± 0.03	0.81 ± 0.00	0.610 ± 0.001	0.22 ± 0.00	0.94 ± 0.01	0.768 ± 0.010	0.02 ± 0.01
teaching	3	120	5	*xgboost*	0.55	0.55 ± 0.02	0.548 ± 0.000	0.01 ± 0.01	0.35 ± 0.00	0.305 ± 0.001	0.65 ± 0.00	0.55 ± 0.01	0.548 ± 0.010	0.00 ± 0.01
tic-tac-toe	2	766	9	*xgboost*	0.97	0.97 ± 0.00	0.974 ± 0.000	0.00 ± 0.00	0.71 ± 0.00	0.822 ± 0.001	0.26 ± 0.00	0.97 ± 0.00	0.974 ± 0.000	0.00 ± 0.00
titanic	2	1760	3	*xgboost*	0.78	0.78 ± 0.00	0.778 ± 0.000	0.00 ± 0.00	0.76 ± 0.00	0.584 ± 0.000	0.10 ± 0.00	0.78 ± 0.00	0.778 ± 0.000	0.00 ± 0.00
twonorm	2	5920	20	*xgboost*	0.98	0.87 ± 0.01	0.745 ± 0.001	0.13 ± 0.01	0.98 ± 0.00	0.891 ± 0.000	0.02 ± 0.00	0.96 ± 0.00	0.884 ± 0.002	0.02 ± 0.00
vertebral-column-2clases	2	248	6	*xgboost*	0.77	0.78 ± 0.02	0.767 ± 0.001	0.05 ± 0.02	0.81 ± 0.01	0.724 ± 0.001	0.14 ± 0.01	0.80 ± 0.01	0.769 ± 0.007	0.02 ± 0.01
vertebral-column-3clases	3	248	6	*xgboost*	0.84	0.84 ± 0.02	0.838 ± 0.000	0.02 ± 0.01	0.80 ± 0.00	0.758 ± 0.000	0.10 ± 0.00	0.84 ± 0.00	0.838 ± 0.000	0.00 ± 0.00
waveform	3	4000	21	*xgboost*	0.84	0.77 ± 0.01	0.625 ± 0.001	0.18 ± 0.01	0.85 ± 0.00	0.705 ± 0.000	0.09 ± 0.00	0.83 ± 0.00	0.732 ± 0.004	0.08 ± 0.00
waveform-noise	3	4000	40	*xgboost*	0.84	0.76 ± 0.01	0.661 ± 0.001	0.19 ± 0.01	0.87 ± 0.00	0.708 ± 0.000	0.08 ± 0.00	0.85 ± 0.00	0.742 ± 0.004	0.07 ± 0.00
wine	3	142	11	*xgboost*	**0.92**	0.92 ± 0.00	0.915 ± 0.000	0.00 ± 0.00	**0.94 ± 0.00**	0.703 ± 0.001	0.08 ± 0.00	0.92 ± 0.00	0.915 ± 0.000	0.00 ± 0.00

Table 4.2 Experimental results for the final 30 UCI datasets.

relationships among attributes. We take this to be an indication that the initial classifiers were too complex for the relatively simple problems considered. Hence, copying here allows us to move to a more suitable solution, with less parameters and training requirements.

On the other hand, we identify a number of cases where copies based on *decision_tree* and *rfc* clearly outperform *logistic_regression*. See, for example, *energy-y1* and *iris*. This is because when the decision function is not linear[2], non-linear copies are needed. Here, the capacity error dominates, because the copy hypothesis space, the logistic family, does not contain $f_{\mathcal{O}}$. Hence, given the same set of synthetic data points, *i.e.* similar coverage error, it is this mismatch that dominates the overall fidelity error.

Finally, in some instances the copy hypothesis space is well chosen and yet the empirical fidelity error is high. See for example *musk_1* and *musk_2*, which are both high dimensional problems where a *linear_svm* is copied using a *rfc*. In both cases, $\mathcal{A}_{\mathcal{C}}$ is notably lower than $\mathcal{A}_{\mathcal{O}}$. This happens in complex datasets, where $1e6$ synthetic data points sampled uniformly at random are probably not enough to ensure a small coverage error. Increasing the size of the synthetic dataset would probably alleviate this issue. Alternatively, depending on the topology of the considered problem, other sampling techniques can also be considered to ensure a suitable exploration of the attribute domain in full. A discussion of the performance of different such techniques is presented in Appendix A for a subset of the problems here presented.

4.4.3 Discussion

The different error contributions of the copying process are collectively defined by the fidelity error and approximated through the empirical fidelity error. However, the condition that empirical fidelity error be small is necessary, but not sufficient to ensure a good copy. Having significant errors in certain regions and none in others may lead to a low error, while altogether not ensuring a good generalization performance. The opposite is also true: a large empirical fidelity error may not lead to a low copy accuracy. Take, for example, errors distributed around the boundary. This may happen when trying to copy a smooth function using linear decision cuts. If errors are very substantial, this may be seen as a problem. However, if the training data are distributed far away from the boundary, errors in this region would have no real impact. No effective error would therefore be measured when substituting the model with the copy.

To a large extent, copy evaluation depends on the available information. The more information we have, the more reliable our estimates will be. If the training data were accessible, we could obtain a direct estimate of copy generalization. Furthermore, assuming this was necessary for the considered application, we could choose P_Z to be as close to P as possible, *i.e.* redefine the copy operation space to match P. If the form of the model $f_{\mathcal{O}}$ was also known, we could refine the choice of copy hypothesis space. In those cases where model and copy have similar decision boundary shapes, copying is conducted with greater ease. That is, when the decision function is formed of cuts perpendicular to the axes, as in the case of

[2]Despite the training data being linearly separable, the learned decision boundary may be non-linear.

a random forest classifier, it is easier to copy it with a decision tree than it is with a radial basis kernel SVM. Conversely, those models with smooth decision functions are better copied using classifiers other than trees.

At this stage, we may again ask ourselves the question: *in case were the training data are available, why should we copy the given model instead of learning a new classifier?* This is a question that was already posed in *Part I*, but which is relevant to revisit here. There exist scenarios where a new training may not be advisable. A new model may display very different behaviour and decision properties. This is unacceptable in production environments where performance has to be preserved and controlled. Moreover, training a new classifier with the same training data involves having to take care of the overfitting effect, whereas when copying we avoid hyper-parameter optimization. Another reason to use copies is that when training a new model we might not be able to recover the same operation point as before. In contrast, a copy can help bias the parameter optimization process towards a desired solution so that we can ensure a smoother transition.

4.5 Further considerations

As discussed in *Chapter 1*, one of the main benefits of copying is that it enables differential replication of machine learning models. This means that copying can be used to enhance existing solutions or to build next generation models that are better fit to meet the requirements of an stringent ecosystem. Copying can, for example, be used to evolve from batch to online learning schemes [35]. This extends a model's lifespan as it enables adaptation to data drifts or performance deviations. Equivalently, when new class labels appear during a model's deployment in the wild, copying can account for the new data points and evolve from binary to multiclass classification settings [79]. More generally, there are numerous examples were differential replication through copying can be applied to solve specific problems. However, despite its flexibility and large range of applications, copying has several limitations, for example, when it comes to dealing with high-dimensional data, or with certain problem environments. In the following, we describe different real-life applications where copying a model may be challenging and discuss different approaches to overcoming the identified barriers.

Copying is highly dependent on the synthetic data generation process. The complexity of this process grows with increasing dimensionality. Hence, while copying remains valid in this context, its performance may be affected. Mostly because sampling an unknown decision function is hard. More so, because we have no information about the training data distribution and lack any insight on how the different classes may be distributed throughout the space. In theory, we could overcome this problem by generating infinite query points. Yet, this is not tractable in practice, since we are limited by our computational resources. Assuming all d variables take discrete values among a finite set $\mathcal{C}$, we would need to generate $\mathcal{C}^d$ to cover the complete attribute domain. While this may be possible for low dimensionality spaces, it rapidly becomes unfeasible for increasing number of attributes and is even worst for the case of continuous

variables.

In our experience, when considering large dimensionality data it is worth replacing uniform sampling distributions with normal distributions. The first conduct an arbitrary exploration of the space, whereas the second better characterize the typicality[3] of a standardized dataset. This is because, as the number of dimensions increases, so do the regions of the space where no data are present. By using a normal distribution to guide sampling we focus only on those areas that could potentially contain data.

Not only the amount of data but also their structure can be problematic. In structured environments, such as images or text, data tend to lie on top of a variety, so that finding the optimal synthetic dataset requires sampling the appropriate manifold. While this may be doable, it is not straightforward. In general, copying in such domains requires access to the training data to generate synthetic data with a suitable representation. This could be done, for example, using an autoencoder that preserves image invariance.

An additional limitation is choosing P_Z. As shown above, blindly exploring the input space works well for simple cases. As the complexity grows, however, so does the intricacy of the decision function and more *ad hoc* techniques are needed to appropriately sample the input space. Appendix A shows our results when assaying different methods to guide sampling when generating synthetic datasets in different copying problems.

Lastly, many local minima exist. This is because an infinite number of different synthetic sets can be used to replicate a given decision boundary. In theory, the empirical error is known and equal to zero, so that all sets should converge to the same result. Due to training variability, however, this is not always the case.

[3]The concept of typicality refers to properties holding for the vast majority of cases [250]

Lessons learned

- The copying loss is theoretically defined by the *fidelity error*, which refers to the disagreement between the original and copy model predictions over the input space.
- This theoretical error can be divided into three different contributions. The capacity error refers to loss that arises from a capacity mismatch between the model and the copy hypothesis spaces. The coverage error refers to the representativeness of the generated synthetic dataset. Finally, the interaction error measures a certain coupling between the other two.
- Collectively, these three terms represent all the theoretical sources of error in the copying framework. However, they are not all measurable in practice given our limited computational tools. Instead, we introduce a set of performance metrics to evaluate copies in terms of the available information.
- In the general case, we assume no access to the training data or the original model internals. In this context, the empirical fidelity error is the only metric at our disposal. In cases where additional information is also available, we discuss additional checks, including the empirical fidelity error over the training domain, the copy accuracy or the estimated copy accuracy,
- We validate all these checks in practice in a set of 60 datasets. We show that copying can be successfully performed in a wide diversity of problems and for different model architectures. Finally, we also discuss practical considerations of this approach.
- In the next chapters we put these ideas to practice to present our idea son how inheritance by copying could be exploited to ensure accountability of machine learning systems in company production environments.

Part III

Practice

"I never have, above my signature, announced anything that I did not prove first. That is the reason why no statement of mine was ever contradicted, and I do not think it will be, because whenever I publish something I go through it first by experiment, then from experiment I calculate, and when I have the theory and practice meet I announce the results."

– Nikola Tesla, *Work with Alternating Currents*

In the previous chapters we have introduced our theoretical framework. In the following, we describe our devised applications. We envisage scenarios where machine learning is deployed in company production environments to deliver a certain product or service to the market. A context in which environmental adaptation through differential replication has already been discussed. Here, we continue this discussion and suggest how inheritance by copying can be used to conduct an effective mitigation of the risks derived from machine learning deployment. We begin by motivating the need for machine learning accountability. In *Chapter 5* we introduce our proposed framework for actionable accountability and demonstrate the practical utility of copying in this context. In *Chapter 6* we suggest two different approaches to deliver interpretable machine learning solutions that yield a good prediction performance, while complying with regulatory requirements. In *scenario 1*, we use copies to ensure the attributes of a risk scoring model remain intelligible. In *scenario 2*, we avoid the pre-processing step by exploiting a high capacity model and then copying it with a simpler yet interpretable one. Finally, in *Chapter 7* we discuss how inheritance by copying can be used to mitigate the bias learned by a given classifier in conditions where this model cannot be modified.

Chapter 5

Risk mitigation in machine learning accountability

5.1 Introduction

In recent years, a growing number of voices have publicly denounced the potential negative impact of reliance upon machine learning [5][34][189][212][160]. Deployment of commercial machine learning solutions constitutes a major source of risk for users, who may be affected by discriminatory practices [9][41] or impacted by decisions which they cannot contest [213] or whose data may be inadvertently leaked [224]. In this sense, they also constitute a risk for the society as a whole. On top of that, there exist additional risks for the companies that deploy these models, in terms of unmet legal requirements for interpretability, unsatisfied performance needs, lack of transparency, non-sustainable deployment or general design flaws. The debate around machine learning has therefore increasingly focused on the issue of accountability [8][43][96]. Many governments have shown their determination to regulate machine learning [158][157][186][180]. Plus, both companies and researchers are dedicating increasing efforts to developing tools to identify and measure the shortcomings of machine learning, as well as to mitigate any potential harm that may be derived from them. Even so, many real problems are still open for exploration.

In general, there exists a gap between theory and practice [17][162][248] that results in most theoretical proposals for accountability failing at meeting the requirements of real life scenarios. While the machine learning community has mainly focused on designing tools for accountability in *in-vitro* settings, where both the data and the algorithms are readily accessible, most real-life situations do not conform to this

ideal. Machine learning models operate in complex environments, subject to a large number of constraints which tend to change in time. Hence, the potential shortcomings of a machine learning model may not be apparent from the beginning. On the contrary, ensuring accountability may require a continuous process of auditing and mitigation. While the protocols for auditing are better understood, tools for mitigation are more scarce. Partly due to the fact that mitigation is not necessarily tied to a legal requirement, a situation which can result in a lack of motivation when it comes to removing the risks. As a result, additional tools are needed that provide a flexible, cost effective approach to accountability.

In this chapter we study machine learning accountability in real-life deployment scenarios. In particular, we focus on studying the role of differential replication through copying as a risk mitigation mechanism in company production environments. Here, models are only a part of a larger system [82] that involves different levels of abstraction, responsibility and knowledge [70][49][156][159][151]. Our proposed accountability framework includes an auditing stage to identify the potential shortcomings of a model. These shortcomings are understood as risks for either the users of the model, the company or the society and should be mitigated or removed. We identify situations where copying might be of value to this end and provide a non-exhaustive list of practical examples. In many cases, additional applications may exist that we are yet to fully grasp. Nonetheless, we believe there is value to understanding how these techniques may contribute to a fairer, more sustainable use of machine learning [2].

5.2 Machine learning systems

Accountability refers to a set of protocols to evaluate the conduct of an individual or entity, as well as an obligation to report or justify one's actions, especially in cases where these may result in any wrong doing [69][249]. Accountability is the instrument through which agents can be held accountable of the consequences of their actions or decisions. Being accountable implies accepting responsibility in face of possible sanctions. Machine learning accountability is therefore the instrument through which we ensure criminal or civil liability for any negative impact derived from the use of this technology. While several contributions have aimed to set the basis for how to enforce machine learning accountability in practice [69][96], no single widely accepted standard exists today [212][164]. This is mainly because most legal and computational proposals to enforce machine learning accountability face many challenges when it comes to their practical implementation in real settings [248].

5.2.1 A model's environment

The lack of success in implementing these proposals in practice has been often attributed to carelessness and indifference on the part of data practitioners [14][162]. However, evidence shows that there exist numerous practical restrictions that prevent such implementation. As discussed in *Chapter 1*, deployment

of commercial machine learning in any big or small corporation is subject to a large number of constraints and specificities. These limitations collectively conform a machine learning model's environment, which includes the following elements, among others.

Business alignment Different business areas may contribute their views on how a machine learning model should be designed and deployed. Moreover, their demands over the delivery performance of the system may change as their business needs evolve. Alignment of business objectives with the devised solutions is a non-trivial task that requires a constant monitoring to prevent performance degradation in time.

Data sources Data usually come from different sources and need to be enriched along the process. Some input attributes are inferred or extracted from records of past events and client interactions. When companies lack informative data, they might resort to third parties for data collection or to trusted sources, such as national statistic agencies, for specific information that might be relevant for the considered application.

Ethics and business rules Companies often have a set of client or project admission rules that define the contexts in which they are willing to do business. These may exclude, for example, trading with companies selling military products, admitting clients below a certain age range or avoiding the use of non-sustainable raw materials. As a result, companies usually lack data in certain regions of the space. This generates blind areas in the attribute domain during training of machine learning models.

Globalized markets Nowadays, companies are not restricted to operating in a single place. On the contrary, they may offer their services and products in multiple geographies simultaneously. Take, for example, the case of cloud-based organizations. In order to optimize resources, these organizations may export machine learning models devised for certain geographies to others. However, this generally requires some form of fine-tuning and the maintenance of several versions running in parallel.

Regulatory constraints Several industries, such as banking or insurance, are subject to great scrutiny by national and international regulators. In the case of banking, for example, regulators require, among other things, that internal coefficients of credit scoring models be accessible and in line with human domain knowledge. This largely limits the type of models that can be used. Given that companies aim to maximize revenue through model accuracy, dealing with such limitations is often far from trivial.

Local country legislation Apart from global regulatory constraints, companies are also subject to the local legislation of each of the different countries where they operate. This legislation may introduce additional safeguards, for example, to eliminate disparate treatment by removing sensible data attributes from the training process.

Company disorganization While models are usually trained by data scientists in dedicated departments, different business areas may claim ownership of the products and services derived from their use. Machine learning deployment in any big, or even small, corporation requires the interaction of several departments and the integration of different points of view, goals and strategies.

Technological infrastructure Models are not trained in a single run, but iteratively, and are deployed to specific production infrastructures through continuous integration. These are designed in terms of the available budget and the pre-existing software licenses and contract agreements. However, a company may decide to modify or even redesign its technological infrastructure at any given time.

Deployment barriers In many sectors there exist huge problems with legacy that put strong barriers to machine learning deployment. Overcoming these barriers involves understanding how deployment itself works, accepting the need for a unique deployment pipeline, and learning to navigate a messy ecosystem.

Impact Decisions output by commercial machine learning models have a significant impact on customers. Take, for example, the case of credit risk scoring in the mortgage market. The estimation of mortgage default risk has a significant impact on the pricing and availability of mortgages. This, in turn, puts a lot of pressure on consumers, as it affects their disposable income. When designing these models companies must understand and account for their potential impact.

To overcome these constraints, machine learning models are usually deployed as part of the larger structure entailed by a machine learning system. A machine learning system comprises all the different actors, considerations, and elements that have to be taken into account when delivering a solution to the market. This is, everything from the business understanding to the deployment of a model, including data identification, collection and pre-processing, model training, evaluation and continuous integration of one's own models with third-party black-box components and APIs, and eventually, production deployment or issues regarding legal aspects and specific regulation. Altogether, these elements display complex interactions that are highly dependent on the considered application. Moreover, while some of them may be subject to strict supervision, most usually evolve in ways that are out of our control and can therefore result in unwanted risks for the company or its customers. Identifying and evaluating such risks is vital to ensuring accountability. In what follows we provide an overview of what they entail and who they affect.

5.2.2 Potential risks of machine learning systems

A system that fails to perform according to its specified requirements can pose a risk to its users. The users of a system are the individuals who are impacted by the its decisions. In the case of a credit risk scoring system, for example, the users are loan applications, who are impacted by the system's estimation of their probability of default. When a system is flawed, its users may be affected by its shortcomings.

Such shortcomings can also pose a risk to the company who is responsible of the system. Consider, for example, cases where the system does not deliver according to expectations. This can potentially have an impact on the company's revenue. In addition, there also exist risks to the customers of the company, independently of whether they use the considered system or not. Companies use data of past and present interactions to train predictive models. In the case above, a company may use data about the outcome of previously granted loans to train credit scoring models. These models may pose a risk to past loan borrowers, for example, if their personal data are exposed to safety breaches. Finally, the shortcomings of a system may also pose a risk to the society as a whole. Especially in cases of critical applications where the misbehavior is perpetuated in time.

Depending on their source, risks can be classified as endogenous or exogenous. A risk is considered to be exogenous if it arises from outside the system. Alternatively, it is endogenous if it arises from the system itself. Endogenous risks are related to a system's architecture, and are therefore largely influenced by the design and training processes. In contrast, exogenous risks are mostly determined by a system's environment. While endogenous risks may be more easily controlled for through carefully defined design protocols, exogenous risks are largely subject to contingency. Changes in the regulatory framework, the appearance of new market trends, changes in clients' behavior or simply the reshaping of a company's business model may give rise to exogenous risks.

Independently of who they might affect, managing endogenous and exogenous risks of a machine learning system requires, first, a framework for identification and surveillance and, second, a remedy mechanism, to mitigate any potential harm. In line with these ideas, we propose a framework for actionable accountability. We build on previous contributions [96][70] to make a distinction between the notions of risk identification, which deals with clearly reporting the potential shortcomings of a machine learning system, and risk mitigation, which focuses on addressing such shortcomings to reduce the negative effects that may be derived from them. This second mechanism includes precautionary measures imposed on systems *by design* and reactive tools to correct a system's misconduct, as well as additional safety measures [104]. In the case of reactive tools, they should be such that they minimize disruption to the pipeline of the given machine learning system.

5.3 Actionable accountability

Machine learning accountability is intrinsically related to an obligation to report and justify automated decision making [69]. This implies having knowledge about how a system behaves in face of different scenarios. In particular, it implies understanding what its behavior will be in those cases that may be particularly sensitive. Hence, enforcing accountability implies being able to audit a system. Auditing is the process whereby an external agent objectively examines and evaluates the performance of a system to ensure no harm is derived from its use. For the auditing to be performed, this agent should have full access to the system. She should be able to interact with it through queries of varying nature and test

its response against different assumptions. She should also be able to check that the system complies with the appropriate architectural safety-guards to guarantee its safety and that of the individuals that interact with it.

In addition to reporting, the notion of accountability is also tied to a need to accept responsibility. It is our believe that this need should be understood in a broad sense. This is, beyond discharging duties, corrective measures should also be taken in cases where potential flaws of a system are identified during auditing. Such measures can be reactive or preventive or, preferably, both. Under this perspective, accountability should enable action.

Moreover, accountability cannot be conceived as an outward journey. A machine learning system's environment is prone to change multiple times and in multiple ways throughout its life-cycle. Ensuring accountability in this context therefore requires a continuous process of going back and forth, checking and updating. As additional information is gathered, this process can and should be enriched to ensure a fully in-depth inspection every time, as well as to equip data practitioners with tools to take action in uncertain environments.

With this view in mind, we introduce actionable accountability as the process summarized in Fig. 5.1. This process consists of two different stages. First, machine learning systems are inspected during the risk-based auditing phase to identify any potential shortcomings or flaws. These flaws may refer to different risk dimensions, as described below. Second, each of these shortcomings are addressed individually during the risk mitigation stage. Finally, for this process to be truly actionable, there exist two enabling conditions: *governance* and *auditability*.

5.3.1 Governance

We consider accountability to be ultimately an issue of trust, of the extent to which we can rely on our knowledge of a system, in terms of how it works and why it behaves the way it does. Accountability is therefore related to responsibility in the legal sense. If algorithms themselves cannot be made accountable of their decisions, there needs to be a clear legal subject or entity who will. This is where the notion of *governance* comes in. Governance encompasses all organizational roles and protocols related to outlining a company's data strategy and ensuring the social impact of machine learning systems is well understood and accounted for before deployment. This includes the need to declare the ownership of the different parts of a system, to properly document the developments, to monitor performance and to define auditing protocols. In sum, governance defines the responsibility of the different stakeholders and clearly designates their roles with respect to the considered machine learning solution. Particularly for those obligations that represent a legal liability.

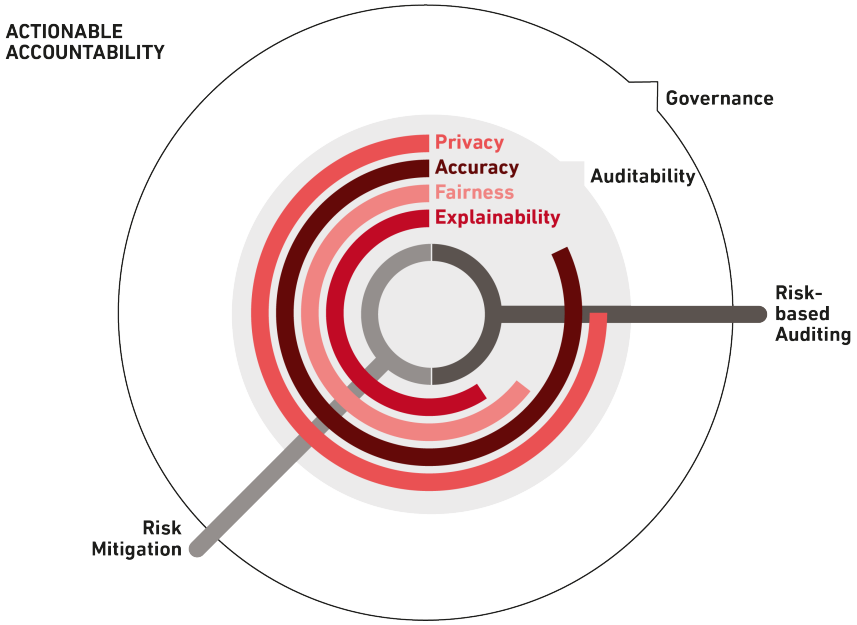


Fig. 5.1 The actionable accountability process. The smaller circles inside correspond to the risk auditing and mitigation stages. The coloured circles surrounding the centre correspond to the different risk categories. The whole process is engrained in two larger structures: governance and auditability.

5.3.2 Auditability

The other enabling condition for accountability is *auditability*. Auditability deals with the capability and possibility of a system to be audited by the general public or by a third party. This may include probing, understanding or reviewing of a system's behavior. Auditability is related to issues of confidentiality and intellectual property. It is an instrumental requirement. In making a system open for auditing, a company must ensure the inviolability of the solution, the confidentiality of the data, the non-disclosure of sensitive information and the maintenance of industrial secrecy in critical or strategic business applications. Hence, we consider auditability to be a necessary condition for the actionable accountability process, *i.e.* a *sine qua non*. Assuming that there exist an appropriate governance framework and that the auditability condition is satisfied, we envisage actionable accountability as a process composed of two different stages: *risk-based auditing* and *risk mitigation*.

5.3.3 Risk-based auditing

The process of algorithmic auditing can be traced back to audit studies in the social sciences [207], where different mechanisms were suggested to probe a system's behavior. In scenarios preceding the data revolution, learning about a system's behavior was no easy task. Some of the proposed mechanisms

included interacting with the system through multiple petitions, posing as potential users or directly confronting a company to obtain relevant information. All these approaches were largely limited by the ability to design the different scenarios of study and collect the necessary data. Today, these mechanisms are mostly automatized and attention has shifted to how and when they should be employed, by whom, to which end and whether they are deemed enough.

Similar to the case of audit studies, auditing in machine learning delves with system inspection. This inspection can be conducted by external third-parties or by the company who owns the system. Or, preferably, by both. In cases where it is the company who performs the auditing, it is usually a dedicated department who is responsible for this process. Or people with specific roles inside the organizational chart. When auditing comes from the outside, it can be performed by external agents hired by the company to this end. Or by institutional representatives. See, for example, the case of the financial regulator, who audits all credit scoring models.

Irrespective of who performs it, it is important that the auditing be conducted in a continuous manner, given the changing nature of a machine learning system's environment. While a system may perform adequately at the time of its conception, the requirements of its environment may evolve very quickly towards an scenario where it is no longer fit to satisfy them anymore. An initial auditing at the time of first deployment may serve identify wide-range shortcomings. However, additional issues may appear throughout a system's life cycle. A continuous surveillance is therefore necessary. Moreover, ideally the process of auditing should encompass the assurance of standards at different levels of abstraction. Hence, the steep learning curve for potential auditors, who are required to demonstrate different levels of technical and tacit knowledge, is one of the most relevant challenges in this field.

The purpose of a risk-based auditing is to ensure that a given machine learning system provides the intended service without unintended consequences or side-effects [49][70]. Much research has been conducted on the topic of algorithmic auditing for several applications [62][174][224][3], including commercial software [9]. Recently, many well-known articles have audited existing machine learning solutions and publicly denounced their shortcomings, for example, when it comes to ensuring a faithful representation of the different phenotypes in face recognition software [41]. Publicly naming a company misconduct may therefore be one outcome of auditing [195]. More generally, this process should provide a clear overview of the scenarios where a given system may under-perform and therefore constitute a risk, in terms of a given dimension of study. In that respect, identifying, measuring, reporting, advising, and acknowledging the different risk sources should be the final outcome of the risk-based auditing stage.

The risk categories

With regards to a machine learning system, we can identify the following sources of risk [119]: (i) lack of alignment with business needs [162], (ii) inconsistency with respect to organizational expectations and requirements, (iii) improper translation of tactical plans from the strategic plans, and (iv) ineffective governance structures that fail to ensure accountability and responsibility. Altogether, these sources may give rise to risks of very different natures. Here, we are interested in those that are most commonly

encountered in commercial machine learning. We group them under the following categories.

Accuracy Accuracy amounts to understanding performance, identifying the sources of error and the limitations of a solution and considering the quality and reliability of the decisions, as well as their direct societal impact. This dimension accounts for the risk of a negative impact because of unreliable or low quality decisions.

Interpretability and explainability The first refers to a measure of the *white-boxiness* of a model. The second seeks the verbalization of algorithmic decisions at different levels of abstraction, corresponding to the different knowledge and needs of stakeholders, regulators and end-users. It accounts for the risks of ensuring that automated decisions can be contested and reasoned upon[1].

Fairness This dimension ensures that automated decisions do not display an unjust or biased behavior with respect to sensitive factors such as gender, ethnicity or religion. It accounts for the ethical and legal risk of discrimination against certain collectives or minority groups.

Privacy The privacy dimension aims at preserving data confidentiality [117]. It encompasses the legal risks of re-identification, which considers the probability of identifying an individual in the training set, data linkage, which concerns the probability of being able of linking/joining records in two different datasets, and sensible attribute inference, which is concerned with the problem of how a machine learning system may be used to infer protected information [19][192][230].

The different risk dimensions above capture some of the potential shortcomings of machine learning systems and may result in harm to different individuals or collectives. For example, they may constitute a risk for the user of a system (risk of bias in prediction), for the company (risk of misalignment between business needs and technical solutions) or even for the average customer, independently of whether he or she is a user of the system (risk of data leakage) and aware of it or not [207].

Auditing in practice

An important characteristic of the risk-based auditing is whether is it *partial* or *total*. A partial auditing focuses on evaluating a system's performance in terms of a single, or multiple, risk dimensions. This auditing is specific to the chosen risk dimension and generally oversees shortcomings related to the others. A regulator requiring that all credit scoring models be interpretable and all the input attributes understood might not be concerned with whether these models have been trained on data that is representative of the company's client portfolio.

[1]It is worth mentioning that the idea of explainability often transcends the machine learning models themselves to include not only the technical but also the human dimension [134]. Nonetheless, in this thesis we approach this notion from a mostly normative perspective.

Alternatively, when conducting a total auditing, all the different risk dimensions are considered simultaneously. In general, conducting such a total auditing is not feasible in practice. This is because highly specialized knowledge is usually required for this purpose. Moreover, reasons for conducting the auditing may diverge from one agent to another. While public institutions may be interested in auditing risks that affect users of a model, they may not be bothered with ensuring that the model is properly aligned with a company's business needs. In contrast, in absence of specific sanctions, a company may lack motivation to report potential biases in its systems. However, it may very well be invested in identifying issues that may affect profit. A more realistic approach that requiring a total auditing is assuming the existence of different auditing stages by different agents. These parallel audits may focus on one risk dimension or the other so that they collectively account for all of them.

Finally, an additional issue related to auditing is the impossibility of proving a system against every possible scenario. Understanding a model's behavior in face of corner cases implies having previously defined such cases. Something which is not always feasible in practice. Certain risks may stay undetected until they become relevant. It is because of this, as well as for the reasons discussed above, that the risk-based auditing phase should accompanied by a risk mitigation stage, during which all the risks identified, either through auditing or through any other method, can be properly managed. The risk mitigation stage refers to the process of providing specific countermeasures to the issues identified during risk-based auditing. This is a key step of the actionable accountability process, since it ensures that there exist tools to avoid potential harms derived from system shortcomings.

5.3.4 Risk mitigation

ISO 31000:2018 [119] provides a set of generic guidelines for the design, implementation and maintenance of risk management processes in organizations. Explicitly, it incorporates the following indications: (i) avoiding risk by deciding not to start or continue with the activity that gives rise to it, (ii) accepting or increasing the risk in order to pursue an opportunity, (iii) removing the risk source, (iv) changing the likelihood of risk, (v) changing the consequences, (vi) sharing the risk with another party or parties (including contracts and risk financing) and (vii) retaining the risk by informed decision. Consider, for example, the risk derived from a misalignment between business needs and a devised machine learning solution. In this case, a company who is aware of the existence of this risk may choose to retain it. Conversely, when faced with a risk of data privacy, a company may be forced to remove this risk. In this document focus exclusively on risk mitigation from the perspective of points (iii), (iv), and (v).

Risk mitigation can be conducted following an *ex-ante* approach, by imposing *by design* principles upon systems. As more knowledge is gathered about potential situations where a system may be found lacking, this knowledge can be incorporated into the design process to refine successive iterations. This is a view that enforces prevention. Alternatively, mitigation can also be conducted *ex-post*, by applying reactive measures that tackle the issues reported during auditing. In order to ensure a fast response, this

measures should be agile. Moreover, in order to motivate companies to take action even in those cases where they are not compelled by legal liability or reputational reasons, these measures should also be cost-effective. Both the *ex-ante* and the *ex-post* approaches are complementary and depending on the considered case it will be one or the other that will be more successful in ensuring risks are handled appropriately. Precautionary measures may and should be in place. Nonetheless, it is often the ability to provide a quick response that makes the difference.

It is our believe that this response should transcend the exclusive legal framework. In other words, potential negative impacts of machine learning systems should be addressed independently of whether they are illegal or not [207]. Very often, however, the absence of a regulatory framework that prohibits certain conducts, discourages companies from taking action. See, for example, some of the cases alluded to before. Even when a risk may be properly identified and reported, the lack of reprobation may result in a company not be willing to mitigate it. Especially if such mitigation is costly. Redesigning a system from scratch is a complex, tiresome process that delays time-to-market delivery and incurs in large costs for a company. There are the costs directly linked to employees' working hours or the use of shared resources. But there are also costs derived from non-earnings when stopping production. Hence, in absence of any external motivation, companies tend to oversee certain design or implementation flaws of systems. In such situations, additional tools need to be devised to provide a cost-effective alternative that guarantees a reliable, yet agile approach to managing deployment risks. Depending on the level of knowledge about the considered system, we foresee different such tools.

Risk mitigation mechanisms

When new data are available and one has access to the internals of a system, mitigating a risk may simply refer to adapting the model and/or the data to be sensitive to the identified risk dimension. This is effectively done, for example, by changing the training loss to accommodate a dimension-sensible term, or by modifying the internals of the model to redefine the hypothesis space so that it is compliant with the level of risk we are willing to assume. Under this category we find most of the literature solutions. As previously mentioned, however, many elements contribute to a machine learning system, so that retraining or fine-tuning may not always be an option.

Alternatively, mitigating a risk may involve adding a new component that wraps the original solution and endows it with a new functionality. Consider once again the example discussed in *Chapter 1*, where we want to measure the confidence of a predictions output by a deterministic black-box system. We could do so by using a *wrapper* to add a layer of uncertainty to this model [170][171]. In some cases, wrappers may require access to the internal states of the model or to more informative prediction outputs. Ideally, however, pure wrappers only have access to inputs and outputs. Hence, as far as these techniques as concerned, systems can be considered to be black boxes. In contrast, knowledge of the training data is always required. Wrappers are useful when data are available, but the solution is very complex or we do not have access to its internals.

Here, we are interested in scenarios where one does not have access to the training data, or the system

is either very complex or not accessible for inspection. In such cases, neither retraining nor wrappers can be used. Instead, we explore how differential replication through copying can be exploited during the risk mitigation stage of the actionable accountability process, when risks identified during the risk-based auditing stage need to be managed and removed from the system.

5.4 The role of copying in risk mitigation

Let us assume a regular commercial machine learning deployment scenario. Without loss of generality, we restrict ourselves to the case of classification. In what follows, we describe possible outcomes of the actionable accountability process, where potential risk are identified during the risk-based auditing phase and discuss how they could be mitigated through copying.

5.4.1 The risk-based auditing stage

The following is a list of possible scenarios where a machine learning system may fail to pass the risk-based auditing stage. Because we are primarily interested in understanding the role of copies as a tool for mitigation, we will not delve into details about how specifically the auditing should be conducted. Instead, we reflect upon different outcomes of this stage. For this purpose, we propose scenarios where the auditor can find the machine learning system lacking in one dimension or the other. Note that these scenarios do not necessarily happen all at once, although we do not exclude the possibility that they do. Mostly, they represent problems that may arise during the system's life cycle. Also note that the list below is non-exhaustive, since we purposely focus on those examples where we believe copying may be of use:

1. A drawback we may come across during auditing are the confidentiality restrictions related to the final product. This is common in companies whose business model relies on industrial secrecy and who require the non-disclosure of the specifics of their data solutions. This is a major issue, since when the confidentiality is not guaranteed and the model is not made accessible to third parties, auditability of the system is not guaranteed and the auditing cannot be conducted.

2. When auditing the in-time viability of a machine learning system we may encounter situations where one of the training attributes is no longer available. Consider, for example, cases where a certain variable is obtained from an external source and added to the dataset. At some point during deployment, this external source may stop facilitating this information any more. In this event, the deployed system would be rendered inoperable therefore posing a risk for the company who relies on its predictions to go on with business as usual.

3. In other circumstances, it is the admission policy of a company itself that may be subject to change. Imagine scenarios where the company wishes to focus on a new portfolio of clients and adapt the existing products or services to this new collective. The new data that might be generated belong to areas of the space that where not accessible to the original solution. This may lead to risk of accuracy due to a decrease in overall performance.

4. Companies may wish to export systems originally devised for certain countries to others. However, differences in local legislation may prevent them from doing so. Consider the case of a bank that wishes to reuse a credit scoring model designed for the Mexican market in Spain. Local legislation differs from one country to the other. Particularly in relation to the use of sensible information in predictive modelling. Indeed, while a given practice may be legal in Mexico, the Spanish law requires that sensitive data not be accessible to the model to avoid discriminative outputs. The use of the gender attribute in credit scoring, for example, is accepted by the Mexican authorities but it is not allowed by the Spain legislation. Trying to deploy the original solution in Spain would therefore constitute a major risk to the company, as well as to the eventual users of the system.

5. As mentioned before, eliminating disparate treatment by removing sensitive attributes may not be enough to avoid that discrimination [96]. Other attributes directly or indirectly related to the protected dimensions could act as *proxies* by leaking the sensitive information into the model. Dealing with this issue is often complex. In applications that exploit personal data, the existence of such proxies may pose a serious risk of bias, even in cases where disparate treatment is not explicitly forbidden by law.

6. Often, when evaluating performance of a model we may wish to incorporate measures other than the error percentage or accuracy itself. Take, for example, the case of customer *churn* prediction, where a predictive system is trained to estimate the probability with which individual clients stop doing business with a company. In this context, measuring the financial impact of each wrongly predicted instance may be more valuable than simply counting the number of errors. When the original system does not incorporate this information, for example in the loss function, an internal auditing might identify a risk of accuracy.

7. For certain applications, the existing regulation imposes explainability requirements on models. In the case of credit scoring models, this regulation requires that the underlying rules and logic of a predictive system be properly described, a demand that focuses on alleviating the potentially negative impact of model inscrutability, as noted by [213].While this requirement does not affect all models in a bank, it usually applies to the credit area. And even if several voices advocate against the use of *black-box* classifier for high-stakes decisions [202], the truth remains that many companies deploy these type of systems to ensure an improved performance. Systems that do not comply with this requirement would therefore incur in a risk of opacity.

8. In this context, logistic regression is a commonly used algorithm. A major drawback of this learner,

however, is that it requires a complex variable preprocessing to obtain a reduced set of highly predictive attributes. These attributes often lack any meaning by themselves, since they represent complex combinations of different dimensions. This may give raise to problems, since it can obfuscate the interrogation of the model [147] and therefore be penalized by auditors.

9. Beyond interpretability, explanaibility is often also a requirement in many disciplines. Recent regulation in the EU, for example, states that companies and institutions should provide meaningful information about the logic involved in automated processing systems [186]. While the implications of this assertion are not yet clear [96][251], the truth remains that companies not complying with this requirement may face legal actions.

5.4.2 The risk mitigation stage

Finally, having identified different scenarios with negative auditing outcomes, we describe how copies could be used to mitigate the identified risks. In all cases, we assume that the original training data are not available to perform a model re-training, nor to add a wrapper to the original structure.

1. When the model internals cannot be fully disclosed for proprietary reasons, it is possible to make a copy available instead. One of the advantages of copies is that they are agnostic not only to the original training data, but also to the model structure itself. Thus, publishing a copy instead of the original model can ensure that no business critical information is disclosed. An additional advantage of this is that it may encourage companies to make their products available for auditing. Note, however, that given that the copy would deliberately omit sensitive aspects of the original model, there might be a trust issue as to whether the copy faithfully represents the most critical aspects of the model in production. In general, disclosing a copy of a confidential model would require transparency on the hypothesis space of the model projection.

2. In cases where one of the original variables is no longer available, it is possible to build a copy that specifically drops this information, while closely replicating the original decision behavior. This can be done by reducing the dimensionality of the synthetic dataset to remove the missing attribute [235].

3. If the performance of a model decreases due to a change in the admission policy rules or in the data themselves, it is possible to move to a copy with online capabilities [234]. This copy can replicate the learned decision behavior, while incorporating new knowledge from previously unseen regions of the space.

4. In general, dropping a variable is not sufficient to avoid bias [14][23]. The information of the protected variable has to be taken into account to actively remove any existing correlation. In cases where this correlation can be measured, we can ensure that there exists no residual leakage of

information after removal of the sensitive attribute. When this is not possible, one cannot avoid an accidental bias, nor the corresponding disparate treatment. A possible solution in this case would be to include this constraint in the cost function when building a copy.

5. When one or more of the attributes convey sensitive information that leads to biased predictions, copies have been found to mitigate the bias learned by models [235]. Same as above, we can remove the sensitive information from the data used to build the copy. This approach is feasible, of course, provided certain checks are in place. In the next chapter we describe this case in greater depth.

6. When training based on cost-sensitive metrics is necessary, it is possible to substitute the original solution with a copy based on an updated loss metric. This would be possible, for example, if using neural nets to build the copies, so that more than one loss function could be simultaneously defined and optimized for.

7. When there exists a regulatory requirement for interpretability, an existing non-interpretable solution can be projected into the set of interpretable models. As a result, we would obtain a regulatory compliant copy, with the same decision behavior. Indeed, previous research has demonstrate the value of surrogate models to ensure *ex-post* interpretability [136][102].

8. The case where unintelligible variables obfuscate the interrogation of the model has been previously studied in detail [233]. Here we can build a model based directly on those attributes in the original set that remain comprehensible. To this end, both the preprocessing module and the model itself can be treated as black-boxes and embedded into the copying process.

9. Finally, when the data system is required to be self-explanatory, it may be useful to move to more flexible model architectures. Copying allows the projection to any desired solution space so that different approaches can be explored.

The above is a is a representative list of scenarios where copying can be used to mitigate specific risks derived from the deployment of a machine learning system. While these examples demonstrate that practical solutions exist to different issues, we note that risk mitigation in certain situations remains largely unsolved. This is mostly due to the difficulty of this task, which involves complex conflicts and trade-offs that are often not achievable in practice.

Lessons learned

- In a world where machine learning models have an increasing presence in high-stakes decisions, we need to devise mechanisms to ensure that they are used safely; and in those cases where they are not, that their shortcomings can be accounted for and properly addressed.
- In absence of such mechanisms, the use of machine learning can pose severe risks to companies, to individuals or to the society as a whole, who might be inadvertently exposed to harm derived from flawed systems.
- Ensuring accountability in this context requires understanding all the complex elements that interact with a machine learning model. These include the data and the models, but also the devised production pipeline, the technological infrastructure for deployment or the different stakeholders that interact with a solution. Machine learning models are highly influenced by a volatile environment that evolves in time and that requires that we design flexible, agile tools for actionable accountability.
- We define this notion of actionable accountability as a process composed of two different stages. The first one, risk-based auditing, is oriented to identifying and reporting any shortcomings of machine learning systems. The second, risk mitigation, addresses these shortcomings by providing effective countermeasures.
- We are primarily interested in the second stage. In particular, we study how differential replication can be used to make existing systems sensible to the identified risk dimensions. Among the different options available, we focus on inheritance by copying and refer to scenarios where neither the training data nor the model internals are known, yet models need to be enhanced to meet specific constraints.
- We describe different scenarios where an external or internal auditing of a machine learning system may identify several flaws. For each of these flaws, we discuss how copying could be exploited to eliminate the risk of a potential negative impact. The following two chapters present a more in depth description of two of these scenarios in the form of industrial use cases.

Chapter 6

Use case: Global interpretability in credit risk scoring

6.1 The context

On 25 May 2018 the new European regulation for data protection entered into force for all European Union Member States [51]. Known as the GDPR, this regulation recognized the data subject's right to be provided with meaningful information about how his or her data is being collected and used by artificial decision making systems. This recognition has been the issue of major debate in the legal community, in relation to whether it effectively creates a data subject's *right to explanation* [95, 251]. Independently of whether this is the case, however, the law's intent on providing the data subject with tools to vindicate his of her rights in face of automated decision making is clear [213]. This intent highlights the pressing importance of human interpretability in machine learning design and deployment.

In recent years, many articles have studied the issue of explaining the outputs of machine learning models. Especially in high impact applications. Some researchers have proposed tools for achieving interpretability by design [202]. However, the truth remains that data practitioners tend to favour the more intricate architectures when looking for performance. Hence, much work has been dedicated to extracting local explanations from black-box architectures [197][198]. Given a sample, these explanations are obtained by building linear surrogates [102][101][204] in its vicinity. Other proposals focus on developing so-called counterfactual explanations. Counterfactual explanations describe the smallest change to the

feature values that needs to be enforced in order to modify the prediction to a predefined output [252][243]. While explanations thus obtained may faithfully represent the functioning of a model locally, they do not account for its global functioning. Moreover, they often fail to meet regulatory requirements in that they may be based on non-interpretable data attributes that obfuscate explanations.

In this use case, we describe different scenarios where a credit scoring model fails to meet regulatory requirements and therefore poses a risk to the company or to its potential clients. For each case, we study how differential replication through copying could be used to mitigate the identified risks. We discuss a two-fold approach to the problem of delivering high performing, regulatory compliant machine learning solutions in the context of non-client mortgage loans at BBVA. On the one hand, we remove the pre-processing step by deploying models that remain intelligible while capturing the non-linear relationships in the data. We copy these models using more flexible structures that comply with the technical requirements, while retaining a good overall performance. On the other hand, we deobfuscate model variables by building copies that learn the decision outputs of pre-trained models directly from the raw data attributes.

6.2 The case

The average ratio of defaults in the Mexican mortgage market for the first, second, third and fourth quarters during the years 2015, 2016 and 2017 was 2.7% [196]. The average ratio of defaults in non-client mortgages granted by BBVA during the same period was around ten times bigger. A result that motivates the need for a refined credit scoring model that better allocates credit resources for non-client mortgage loan applications.

Failure to keep with loan repayment, otherwise known as credit default, has significant cost implications for financial institutions. Residential mortgages, being one of the most common type of lending [71], constitute a major source of risk for any bank. More so when loan applicants are non-clients. This is, when there exists no previous active contract between lender and borrower at the time of loan application. In such cases, the bank keeps no previous record on the loan applicants, so that the data used to estimate the creditworthiness of each claim is not based on objective evidence. Instead, it is generally declared by the applicants themselves, inferred using indirect methods or provided by trusted external data sources. As a result, obtaining accurate estimations of the probability of default is far from trivial.

Under such circumstances, huge amounts of money have been dedicated to increase model sophistication to learn complex problems with a high degree of accuracy. This trend has led to the proliferation of so-called *black-box* systems. These are intricate systems that are trained on huge volumes of data and which generally yield a good performance. A main disadvantage of these models, however, is that they fail to provide a comprehensible account of how they reach their conclusions. A situation that stands in contrast to the growing demand for transparency in automated decision making systems [157][158][186][180]. Especially in the financial industry, where, as mentioned before, loan issuers are often required by law to

explain the mechanisms behind the risk scoring models that inform decisions about whether to approve or decline loan applications [51][95]. Failure to provide such explanations could result in legal liability and therefore constitute a major risk for companies. A condition that largely limits the type of models that can be used in practice. When a financial institution trains a credit risk model, prediction accuracy is of paramount importance, *i.e.* companies aim to maximize revenue through model accuracy. Yet, there exists a complex trade-off between accuracy and interpretability. In general, it is the most complex models, such a deep neural networks, that tend to perform better when compared to the simpler structures, which are more easily understood by humans.

In the case of credit risk scoring, logistic regression remains the most widely established technique. Models based on logistic regression, perform relatively well while offering the additional advantage of a relative ease of interpretation [241]. Moreover, being one of the simplest types of models, logistic regression is known and understood by most data scientists. A main drawback of logistic models, however, is that they are linear. To overcome this limitation, non-linear effects are usually modelled during a pre-processing step, when domain knowledge by experts is exploited to obtain a set of highly predictive artificially generated attributes. Obtaining this set usually requires a tiresome and costly process of trial and error by risk analysts. This process can take up to 6 months and delay time-to-market delivery. Moreover, this practice is against the idea of *intelligibility* as described in [154] and often results in *non-decomposable* [147] machine learning architectures. Conversely, using more complex models, such as deep artificial neural networks, that capture the non-linearities in the data results in machine learning solutions that do not comply with the regulatory requirements, because their internals are not open for inspection. The problem of balancing accuracy and interpretability in credit risk scoring therefore remains unsolved. In the following lines, we propose differential replication through copying as a way-around this compromise.

6.3 The data

We use a private dataset consisting of information about 1.328 non-client loan applications recorded by BBVA during 2015 all over Mexico. At the time of loan application all individuals in this dataset were considered to be creditworthy and granted the loan. However, only 1025 of them paid it off. This corresponds to a ratio of defaulted loans of the 23%. Due to proprietary reasons, this dataset is not publicly available.

The complete dataset consists of the 18 attributes listed in Table 6.1. The data include attributes related to the characteristics of the loans, such as their total amount and duration, together with socio-demographic and financial information about the applicants. Some of the attributes, including the *poverty_index* or the *economy_level* are estimations made by the bank. There are also additional attributes such as the *estimated_mila_income* which corresponds to an estimate of each individual's annual income and which is provided by the Mexican Treasury Ministry.

Attribute	Description
indebtedness	Level of indebtedness
credit_amount	Amount of credit
property_value	Property value
loan_to_value	Loan to value
duration	Duration of the loan
studies	Level of studies
poverty_index	Marginalization/poverty index
age	Age
est_soc_income	Estimated socio-demographic income
value_m2	Value per square meter
est_income	Estimated income
installment	Monthly installment
n_family_unit	Members of the family unit
est_mila_income	Estimated income based on MILA model
p_default	Ratio of contracts defaulted in the last 4 months from those signed during the previous 12 to 24 months
zip_code	ZIP code
municipality	Municipality
economy_level	Level of economy

Table 6.1 Complete set of attributes.

6.4 The scenarios

We describe two different scenarios for this problem. In the first case, we assume a logistic regression model is trained on a reduced set of highly predictive attributes that are considered to be obfuscated for the purpose of providing an explanation. In the second, a higher capacity model is trained directly on the raw data attributes to obtain more accurate default probability estimations, at the cost of understandability.

6.4.1 Scenario 1: Deobfuscation of the attribute preprocessing

In the first scenario we assume a standard risk modelling production pipeline where the original input is pre-processed to obtain a reduced set of highly predictive attributes. We build on previous knowledge on this task to manually craft 6 high predictive variables, based on combinations among the existing features. We use these attributes to learn a logistic regression model.

In a real setting, a qualified risk analyst would have to conduct a tedious process of trial and error to obtain this set of predictive variables. This incurs in a large economical cost and a delayed time-to-market delivery. Even worse, this pre-processing largely reduces the intelligibility of the resulting model.

This is because the pre-processed variables often reflect complex relations among the data attributes that cannot be easily explained. Indeed, while the logistic regression itself may be linear, the relationships encoded by the pre-processed variables are non-linear to ensure that the final model is able to capture complex patterns in the data. This may pose a serious risk both for the company, who may be ignoring demands for intelligibility, and for loan applicants, who might not have the level of knowledge required to understand the provided explanations.

To mitigate these risks, in this scenario we propose to build a copy of the whole predictive system, composed of both the pre-processing module and the logistic regression, using an interpretable model that is nonetheless able to capture the non-linearities in the data. This is, we propose to substitute the original pipeline with a non-linear, yet interpretable model that is directly applied on the raw input features and which replicates the predictions outputs of the pre-processed logistic model. A benefit of this approach is that, because the new model is applied directly over the non-processed variables, the resulting decision path is more easily understandable.

6.4.2 Scenario 2: Regulatory compliant, high-capacity copies

In the second scenario, we assume no attribute pre-processing and train a model with a higher capacity instead. This can significantly reduce time-to-market delivery by speeding up the training stage. Additionally, it is expected to yield better performance results. In being too complex, however, this model may fail to provide an understandable account of its decisions. Hence, when deploying it in highly regulated markets, such as the European, the bank could face the risk of legal actions. In order to mitigate this risk, we replicate the decision behavior of the existing model using a copy that complies with regulatory requirements, while reaching a comparable predictive performance.

On this basis, we also discuss how this approach could be exploited to explore the contours of the accuracy-interpretability trade-off. In credit risk scoring, there exist different agents that interact with a machine learning solution. The data scientist fits and fine-tunes the model, the regulator ensures that the resulting machine learning system complies with the law, the computer engineers deploy the model to production, and the final client is affected by the decisions output by the system. All of them are entitled to an explanation, which may be required during the auditing stage. However, they may have different expectations as to what kind of information an explanation should convey. Additionally, it is reasonable to assume that they have different levels of technical knowledge. In this scenario, we build copies based on models of different complexity to explore how this methodology could be used to adapt models to be understandable to each of the different parties involved.

6.5 The experimental settings

Due to the sensitive nature of bank data, we anonymize and identify all customers using randomly generated IDs. In the first scenario, we artificially generate the 4 data attributes shown in Table 6.2. As mentioned before, these are mostly based in combinations of the rest of the variables. These 4 attributes were selected from a larger list in terms of their higher predictive value. We use these attributes together with *age* and *economy_level*. In the second scenario, we use all 18 attributes listed in Table 6.1. In both cases, we convert all nominal attributes to numerical using label encoding for ordinal attributes and one-hot encoders in the case of cardinal variables. Additionally, we re-scale all attributes to the [0,1] range.

Attribute	**Description**
zip_code_municipality	Bivariate attribute resulting from the concatenation of features zip_code and municipality
est_soc_income/est_mila_income	Univariate attribute resulting from the ration between features est_soc_income and est_mila_income
property_value/installment	Univariate attribute resulting from the ratio between features property_value and installment
indebtedness/loan_to_value	Univariate attribute resulting from the ratio between features indebtedness and loan_to_value

Table 6.2 Reduced set of highly predictive attributes in scenario 1.

We perform a 80/20 split to obtain stratified training and test sets. In the first scenario, we use these data to train a logistic regression model on the pre-processed data attributes. The whole predictive system composed by both the pre-processing and feature engineering step and the logistic model, yields an accuracy of 0.77. In the second scenario, we use the raw training set to train a gradient-boosted decision tree classifier using a double 3-fold cross validation search. In the first iteration, we perform a broad search and then narrow down the search space for the second iteration. We train the final gradient-boosted tree with the parameter values listed in Table 6.3. This model yields an accuracy of 0.79. This value is sensibly higher than that obtained by the pre-processed logistic regression in *scenario 1*. This is because the learned decision function is able to capture non-linear relationships among original data attributes directly.

In both scenarios we assume the training data distribution to be unknown. In the first case, we draw

Parameter	Value
gamma	0.1
learning_rate	0.1
max_depth	4
min_child_weight	5
n_estimators	100

Table 6.3 Parameters of the gradient boosted tree in scenario 2.

samples randomly from a uniform distribution defined over the raw attribute domain. We label these samples by passing them first through the pre-processing module and then through the logistic model. In the second case, we define P_Z to be a standard normal distribution. We draw samples from this distribution and label them according to the predictions of the gradient boosted tree model. In both cases, we build balanced synthetic sets comprised of 10^6 instances. For validation purposes, we also generate an additional test set of the same size.

In the first scenario we copy the pre-processed logistic regression model using a decision tree classifier. In the second scenario, we project the original gradient-boosted tree onto two different model hypothesis spaces. Initially, we select the set of logistic regression classifiers. Then, we assay decision tree classifiers of varying depths. In an initial approach we let trees grow until the end (*tree_none*). We then force more compact representations by decreasing the depth parameter from three layers (*tree_3*), to two (*tree_2*) and finally one single layer (*tree_1*). Following the discussion in *Chapter 3*, we enforce no capacity control when copying, so that we use *misclassification-error* as the splitting criteria for the trees. We report all metrics averaged over 100 independent runs.

6.6 The results

The distribution of results for the copy decision trees in *scenario 1* is shown in Fig. 6.1 for the different runs. The mean copy accuracy is 0.71 ± 0.04. The trees replicate the decision behavior of the pre-processed logistic model without significant loss of performance. Moreover, in substituting the original solution with the tree-based copies it is possible to generate explanations based directly on attributes in the original set, which remain comprehensible in most cases. This ensures that we maintain the decomposability of both the feature crafting process and the machine learning model itself. Where we had to resort to complex data combinations before, we can now directly explain the predictions of our model using the more easily understandable raw attributes.

In Table 6.4 we report the mean values for the empirical fidelity error over the synthetic dataset, the empirical fidelity error over the original dataset and the copy accuracy for the five different copy architectures proposed for the second scenario. The first row of results corresponds to the copies based

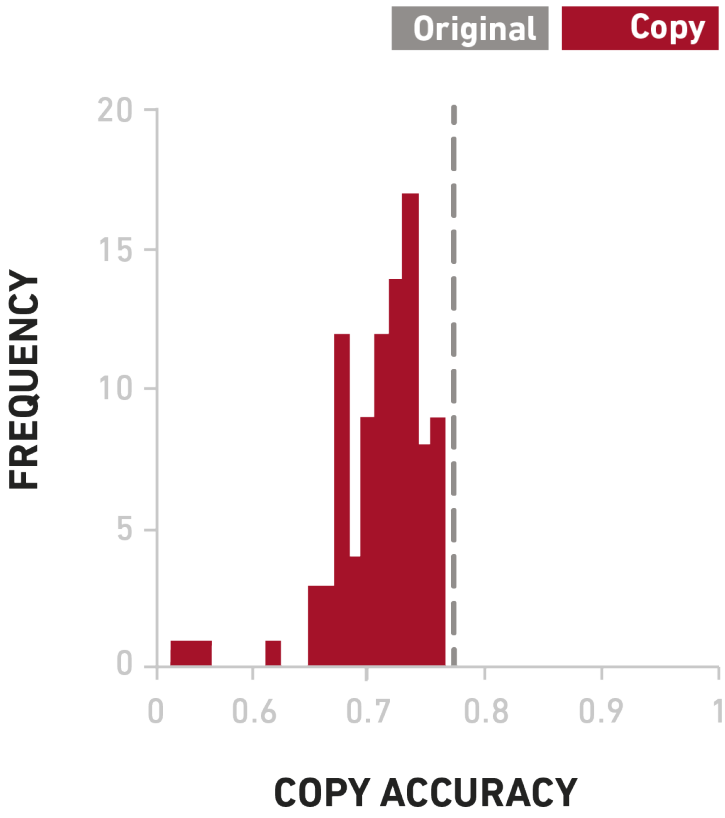


Fig. 6.1 Distribution of copy accuracy for decision tree classifiers that replicate the pre-processed logistic regression model in scenario 1. Results correspond to 100 independent runs.

on logistic regression. The overall loss in accuracy over the original test data is 0.032. For comparative purposes, we also train a logistic regression classifier directly on the training data[1]. This model yields an accuracy of 0.73, which corresponds to a loss of 0.06 points with respect to the original model and of 0.028 points with respect to the measured copy accuracy. We use this model as a baseline to compare our results. This comparison is relevant because it shows that the projection of the gradient-boosted tree onto the logistic family leads to a more optimal solution than direct training on the true data labels.

In addition, the first row of results also shows the values obtained for the empirical fidelity error over both the synthetic and the original datasets for copies based on logistic regression. These values measure how similar the decision boundaries learned by the original gradient boosted-tree and the copy logistic regression are. Results show a reasonably high resemblance. Fig. 6.2 shows the ten attributes with the largest coefficients assigned by the copy logistic regression and their corresponding absolute valued scores.

Model	$\mathcal{R}^{\mathcal{F},\mathscr{Z}}_{\mathbf{emp}}$	$\mathcal{R}^{\mathcal{F},\mathscr{D}}_{\mathbf{emp}}$	$\mathcal{A}_{\mathcal{C}}$
logistic	0.1282 ± 0.0001	0.095 ± 0.002	0.758 ± 0.002
tree_1	0.301 ± 0.002	0.291 ± 0.024	0.619 ± 0.028
tree_2	0.233 ± 0.003	0.141 ± 0.004	0.722 ± 0.001
tree_3	0.212 ± 0.006	0.125 ± 0.003	0.717 ± 0.016
tree_none	0.172 ± 0.083	0.105 ± 0.065	0.731 ± 0.042

Table 6.4 Empirical fidelity error over the original and synthetic datasets and copy accuracy for the 5 different copy architectures.

The next four rows of results in Table 6.4 correspond to projections of the gradient-boosted tree onto the space of decision tree models. We assay varying tree depths to obtain different representations of the given solution. The depth of a tree is directly related to its capacity to fit the data, as well as to its complexity, *i.e.* the number of decision rules it is composed of. Using shallow trees can be useful, for example, to provide clients with succinct explanations that remain informative. Our results, however, show that performance gets better as we let trees grow larger. The average copy accuracy for the smallest trees is 0.591 ± 0.003. Conversely, the larger trees yield a mean copy accuracy of 0.76 ± 0.02. The fidelity error in each case gives us an intuition of the amount of information lost due to the compression: the more complex the copy model, the less information we lose and the lower the values of $\mathcal{R}^{\mathcal{F},\mathscr{Z}}_{emp}$ and $\mathcal{R}^{\mathcal{F},\mathscr{D}}_{emp}$ are. In contrast, the more faithful the copies are to the original solution, the less useful they become for the purpose of extracting understandable explanations.

This idea is better depicted in Fig. 6.3. Here we report the change in accuracy for copies based on decision trees of increasing depths. The shallower trees compact all the information in the original

[1]We exceptionally use the original training data to train the baseline logistic model. Yet, we stress that these data is not used when copying.

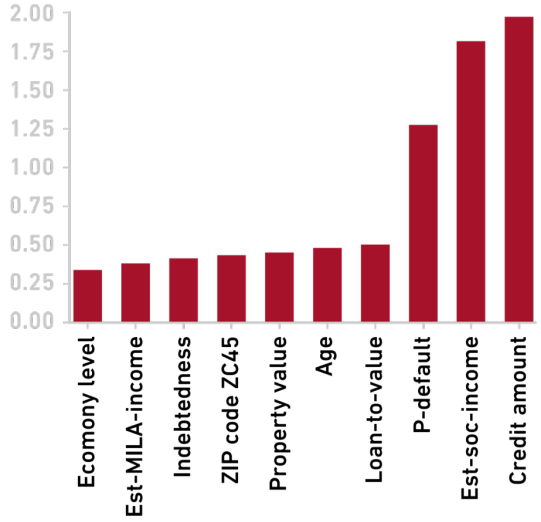


Fig. 6.2 Top 10 largest attribute coefficients in average for the copies based on logistic regression. Bars display absolute valued weights.

solution into a single layer that captures the most variability. As the number of layers increases, so does the amount of information captured by the copies, which grow richer as more layers are added. Indeed, the deeper the trees, the better they perform. Equivalently, the more understandable the trees, the smaller the depth, the higher the loss in accuracy. This plot shows how copies can be used to provide explanations of varying levels of complexity, while at the same time controlling the associated accuracy loss. These could be used to provide clients with explanations in cases where they demand so. But also to provide data scientists or computer engineers with understandable copies to aid in the process of monitoring a given solution.

The decision paths for example copies with varying depths are shown in Fig. 6.4. Because all decision trees are built using the misclassification-error splitting criteria, the initial nodes are shared among the shallower and the deeper trees. As the number of layers increases, copy trees capture additional information through other attributes to enrich the initial splitting. Equivalently, the richer the trees the more intricate they get. The decision paths for trees of unbound depth can reach up to 10 levels for this problem and are therefore not depicted here. When comparing these diagrams with the barplot in Fig. 6.2 we see that the two model families, logistic regression and decision trees, both assign a greater importance to the same set of variables. We take this to be an indication that the projections are consistent across the different hypothesis spaces.

Note that the decision paths in Fig. 6.4 provide an explanation of the copy model's global behavior. The decision rules in the initial nodes provide the more general explanations that capture the most variance. As we traverse the trees towards the inner most nodes, we can refine these explanations by providing more detailed information. Moreover, note also that the outermost node refer to attributes which

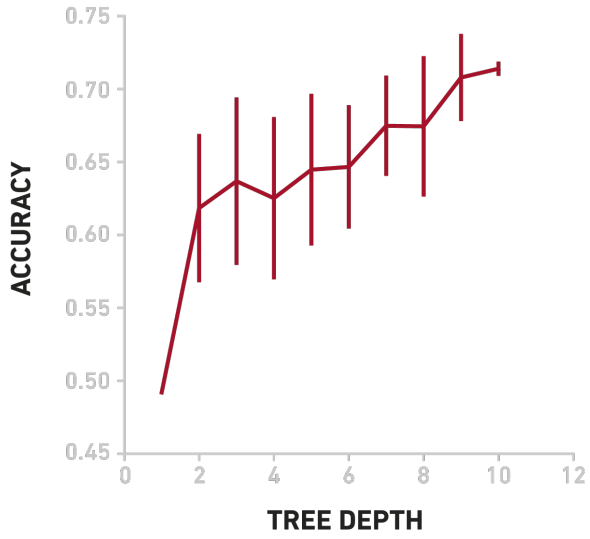


Fig. 6.3 Average copy accuracy for increasing copy tree depths. Error bars correspond to the standard deviation over all runs.

carry a great significant in the context of credit scoring. Variables like *credit_amount* or *est_soc_income* appear on the first levels of the trees, which indicates that the resulting explanations are consistent with the considered problem.

In this second scenario we focus on projecting the original model into a new hypothesis space that encloses model architectures from which explanations are more easily extracted. This allows us to move to a new solution that complies with regulatory requirements and which can be presented to the regulator. Additionally, it also gives us a tool to provide clients with explanations in cases where these are necessary. Further, we could also envisage ways to aid in the process of monitoring a given solution by providing data scientists or computer engineers with understandable copies, the form of which would adapt to the specific needs in each case.

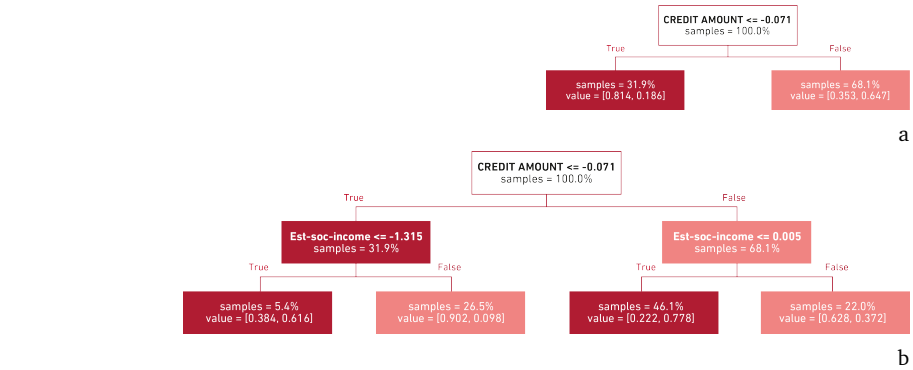
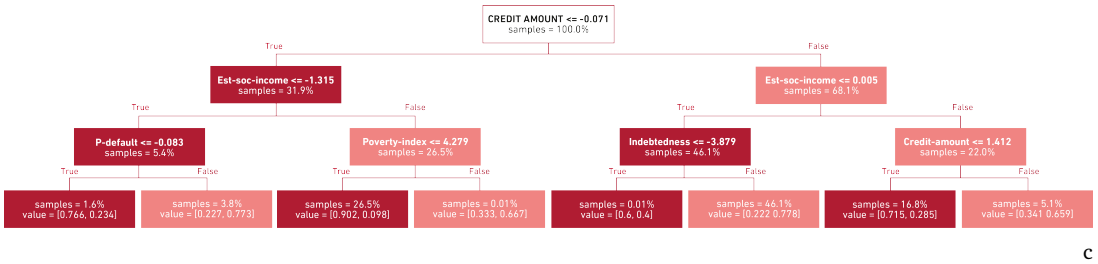


Fig. 6.4 Decision paths for different tree depths. Plots show decision paths for copies based on decision tree classifiers with depths (a) 1, (b) 2 and (c) 3.



Lessons learned

- In highly regulated environments, such as the banking industry, data practitioners often need to deal with a complex trade-off between predictive performance and understandability of deployed solutions.

- In those cases where understandability is imposed by the existing regulation, practitioners often resort to complex pre-processing techniques to ensure a good level of accuracy in models such as logistic regression, which are deemed acceptable by regulators.

- This pre-processing is mostly used to model the non-linearities in the data. Hence, it usually leads to the creation of new variables created as combinations of others. These artificially created variables are often non-intelligible and obfuscate the resulting models.

- In this use case we show that it is possible to de-obfuscate these models by copying the whole predictive system composed of both the pre-processing module and the logistic model using non-linear yet interpretable models, such as decision trees. As a result, we can obtain copies that perform almost as well as the pre-processed logistic regression, but which are based on the original data attributes, which usually remain comprehensible.

- In addition, we also demonstrate the feasibility of another approach that circumvents the need of having to conduct an expensive and time-consuming pre-processing of the variables. We train a high capacity gradient boosted tree directly on the raw variables and copy it with simpler models to ensure interpretability of the final solution.

- In this second scenario we also show how these approach can be exploited to obtain copies of varying depths and complexities. This method allows data practitioners to control the amount of accuracy they are willing to give up for the sake of interpretability and *viceversa*.

Chapter 7

Use case: Mitigating the bias learned by trained classifiers

7.1 The context

The growing use of machine learning is partly explained by the fact that it provides highly accurate predictions [64] with very limited human intervention. This is particularly critical in contexts that involve high-stakes decisions, such as those where the life of someone is at stake, those that involve basic human rights like access to housing or those on which a company's whole business model relies. There is also a generalized belief that machine learning models provide an objective evaluation of social problems. Human decision making follows a clinical approach, which is based on intuition and the subjective processing of information [123]. In contrast, machine learning models generally fall under the category of actuarial decision making techniques. They rely on empirically established relations and statistical analyses of the available data to reach a conclusion. As a result, they are perceived as a reliable alternative to human cognitive biases.

However, while models may escape prejudices, the data with which they are trained do generally not. Models can only be as good as the data they are trained with and data are often imperfect [55], so that even well-intentioned applications might give rise to objectionable results [16]. The data we use to train machine learning models are a collection of past events influenced by previous human decision-making. They therefore often reflect historical prejudices and cultural stereotypes enforced by the people charged with making decisions in the past. Machine learning models that learn from labeled data are susceptible to inheriting these biases. Even when they are not present in the data *per se*. Take, for example, datasets

with a poor representation of the different minority groups. This phenomenon is known as sample size disparity and can lead to higher error rates for certain collectives, as compared to the majority group [16].

In recent years, many studies have publicly denounced machine biases in terms of ethnicity [9][41][128][191], gender [33][42] or sexual orientation [100]. Studies on analogy generation using word embeddings, for example, have demonstrated that the popular Word2Vec space encodes gender biases that are potentially propagated to systems based on this technology [33]. Similarly, models trained to learn word associations from written texts have been shown to display problematic attitudes towards ethnicity and gender [42], including associations between female names and family or male names and career. Moreover, apart from the biases that arise from the data, there are also those that are produced by a poor training of the models. These deficiencies can appear either because of technical limitations or due to the nature of the learning process itself. This process is today mainly based on exploiting correlations among the different data attributes. Very often, however, such correlations are of spurious nature. Models thus built can therefore rely on non-causal relationships that can lead to discriminative outcomes. Examples of this are numerous cases of significant racial disparities in commercial facial recognition software [91][201].

The harmful effects that derive from biased outputs in machine learning systems are generally categorized depending on whether they result in allocative or representative harm [15]. Allocative harm refers to the withholding of resources or opportunities from certain groups of people on the basis of attributes not relevant for the considered task. Take, for example, the case of a financial institution systematically declining to grant loans to non-Caucasian individuals. Conversely, representative harm is related to a reinforcement of the subordination of certain collectives in terms on their ethnicity, social class or gender. A high profile example of this is the Google Photos image classifier, which effectively denigrated dark-skinned people by tagging them as gorillas [110]. Independently of the form of the harm they produce, models that reproduce existing patterns of discrimination work as reinforcement loops of the *status quo* [17][105]. Biased machine learning models promote a system that unfairly undermines the rights of individuals belonging to protected minorities by preventing them from accessing products and services with equality of opportunities. They can, hence, constitute a major risk for users, as well as for the society as a whole, in so far as such practices exist.

In this use case we explore the potential of differential replication through copying to tackle such issues in certain environments. We show how, under certain circumstances, copying can be used to mitigate the bias learned by a machine learning model by removing all the sensitive information. We validate this proposal experimentally and show that we can maintain most of the original predictive performance. Even in cases where the original training data are not available.

7.2 The case

We explore how to reduce the bias inherited by a machine learning classifier which has been already deployed using sensitive information and which cannot be modified. We do so by means of a fictitious

example that nonetheless represents a use case common to many real scenarios. We use the publicly accessible superhero dataset [222], which serves as a good proxy to many real problems where the data contain sensitive information. We choose this public dataset in order to avoid disclosure of private sensitive data. Also because using publicly available data allows us to freely study how the suggested modifications affect variables and instances.

The superheroes dataset contains information about a few hundred superheroes in the literature, including their physical attributes, powers and alignment to good or evil. Among the different attributes there are those that account for protected group features. This is the case, for example, for attributes like *gender* and *race*. In general, usage of this information is not allowed for high-stakes decision making. Without the appropriate control, models trained on these attributes could lead to unfair decision outputs. In this use case, we assume an industrial application where usage of this information is legal. We consider a machine learning classifier that has been trained using sensitive attributes and which outputs biased predictions.

In recent years, many works have studied how to remove bias in prediction [17][76][106] as well as to benchmark discrimination in various contexts. Fairness-aware learning has, as a matter of fact, received considerable attention in the machine learning community of late, with most solutions being aimed at introducing new formal metrics for fairness and ensuring that classifiers satisfy the desired levels of equity under such definitions. Solutions often come in two types. In the first case, an exhaustive data pre-processing removes the ability to distinguish between group membership by getting rid of the sensible information in the training data [80]. This amounts to removing the sensitive attributes themselves, but also to ensuring no residual information is encoded by the remaining data. While simple, this approach often succeeds in repairing the original disparity. In the second case, unfairness is removed by adding corrective terms to the optimization function. A fairness metric [76][106] is defined and incorporated to the training algorithm. Initially biased models are therefore re-trained ensuring that the fairness measure is optimized together with the defined classification loss.

In stringent company environments, however, the deployed machine learning models usually cannot be re-trained once served into production. This is partly because of the way in which machine learning pipelines are conceived. Throughout the stages of deployment, different departments and agents interact with the model. A well established governance framework ensures that data practitioners have access only to certain information at each stage. Hence, going back is often not possible, either because the data are lost or subject to privacy constrains or because the server where the data are hosted is not accessible any more. Whatever the cause, these restrictions make a model re-training ineffective from an economical and practical perspective.

Alternatively, when the bias arises from the intervention of humans in the sample collection process, *i.e.* the dataset is unbalanced or specific minority groups are not equally represented, several papers have advocated for either collecting new data points or using advanced data synthesis techniques [108][131], when this is not possible. Finally, recent proposals suggest moving on from learning based on correlations to being able to draw causal relationships among the data [126]. This would effectively remove any

dependence of the model on features non-relevant for the considered task. However, this kind of techniques are not yet ready for mass adoption.

In what follows, we propose an alternative approach suitable for very stringent environments, where access to the original training data is not supported and re-training is not a cost-effective alternative. We explore how differential replication through copying could be exploited to remove traces of the sensitive information from trained models and show that this can effectively reduce the learned bias in certain circumstances.

7.3 The data

The superheroes dataset [222] describes characteristics such as demographics, powers, physical attributes and studio of origin of every superhero in SuperHeroDb [223]. It contains information about 177 attributes for 660 superheroes. This includes the general information listed in Table 7.1, together with information about whether different superpowers are present in any given hero. We use these data to define a binary classification problem choosing superhero alignment as the target attribute. We label as *good* all superheroes marked as so and as *bad* otherwise. The distribution of target labels is slightly unbalanced, with a third of the dataset set to the positive label, *good*, and the remaining two thirds labelled as *bad*. In terms of *gender*, a 69% of the superheroes are males and a 27% females. The remaining 4% are listed as *other*. The *race* attribute includes 22 different categories. A 23% of the superheroes are human and an additional 1% are humans who have been affected by some form of radiation. Mutants account for 10% of the data, while other races, including *God/Eternal*, *Android* or *Demon*, individually account for less than 1% of the instances.

Attribute	**Description**
name	Name or AKA of the superhero
gender	Gender of the superhero
eyecolor	Color of the eye
race	Race of the superhero
haircolor	Color of the hair
height	Height measured in centimeters
publisher	Publisher of the comic where the superhero appears
skincolor	Color of the skin
alignment	Alignment of the superhero

Table 7.1 Complete set of attributes.

7.4 The proposal

We assume that the original solution to this binary classification problem incorporates knowledge of both the *race* and *gender* attributes and that this gives rise to biased predictions that affect the different groups disproportionately. Since this model cannot be modified and the training data are not available for a re-training, we suggest to mitigate the learned bias copying it with a new model instead.

In the simplest approach, we require the copy to not have access to the sensitive data. As mentioned before, however, this approach may not be enough to remove the bias. Hence, we also require that the sensitive information not be leaked to the copy through the remaining attributes. This is, we require that the sensitive information not be known neither explicitly nor implicitly. In the copying framework this can be accomplished by changing the operational space of the copy, introduced in *Chapter 3* as $\mathcal{H}_{\mathcal{C}}$, to move to a lower dimensional space that doesn't include the two problematic variables.

Effectively, this can be achieved by removing the sensitive attributes during the synthetic sample generation process. To do so, each new sample $\boldsymbol{z}_j$ is first generated to match the dimensionality of the original training instances $\boldsymbol{x}_i$. This is so because the original model used as an oracle only accepts *total* queries. Then, once the sample has already been labelled, the sensitive dimensions are removed. This ensures that the copy has no explicit access to the sensitive information. Further, to ensure no implicit information remains in the system, we also check whether any correlation exists between the removed attributes and those still in the synthetic dataset. Because the copy is built to replicate the original decision behavior, we expect it to re-adjust the learned decision boundary to maximize performance even in the lack of the sensitive data.

7.5 The experimental settings

We begin by removing all entries with an unknown alignment label. We also discard all attributes for which the number of missing values exceeds the 20% of the total size of the dataset. For the remaining columns, we set all missing values to the median for numerical attributes and to *other* for categorical. For the latter, we also group under the general category *other* all values with a count below a defined threshold. We set this threshold to 1% for variable *eye color* and to 10% for *publisher*. In the case of the *race* attribute, we group entries under the more general categories of *human*, *mutant*, *robot* and *extraterrestial*. All those entries that do not fit any of these categories are and grouped under *other*. Additionally, we only retain those superhero powers present in at least a 1% of the entries. Finally, we convert nominal attributes to numerical by means of one-hot encoding and re-scale all variables to zero-mean and unit variance. The resulting dataset contains 135 variables.

We split these data into stratified 80/20 training and test sets. We use the former to train a fully-connected artificial neural network with 4 hidden layers, each consisting of 128, 64, 32 and 16 neurons. We

use *SeLu* activations, a drop-out of 0.6 and a softmax cross entropy loss optimized using *Adam* optimizer for a learning rate equal to $1e-3$. We train the network from a random initialization of weights and without any pretraining. We use balanced batches with a fixed size of 32. We use this model as our baseline.

In order to copy this model, we generate a balanced synthetic dataset consisting of $1e6$ labelled data pairs, from which we extract the two problematic attributes. We generate this set using different sampling strategies for numerical and categorical attributes. For the first, we directly generate synthetic data points in the original attribute domain by sampling a random normal distribution with mean 0 and standard deviation 1. In the case of categorical variables, we sample uniformly at random the original category set. When generating new synthetic values for superhero powers, we ensure that the relationships among the original data attributes is kept. To do so, we sample uniformly at random the *n_powers* variable and then randomly distribute the total count over the individual power attributes. We use the lower-dimensional synthetic dataset to learn a new artificial neural network with the same architecture and training protocol as that of the original model, with a fixed batch size to 512 and no drop-out.

We measure bias in terms of the difference in accuracy between the *gender* and *race* groups. We evaluate copies using the empirical fidelity error over both the synthetic dataset, and the original dataset; and the copy accuracy. In all cases, we run each experiment 10 times and report metrics averaged over all repetitions.

7.6 The results

In many real scenarios, systematic bias results in individuals belonging to privileged and unprivileged groups not having access to the same resources, a reality that could very well be reflected in the remaining data attributes of each group. Hence, before proceeding with our suggested approach, we ascertain the feasibility of our proposal: we verify that the removal of the two sensitive attributes will not result in any residual leakage of information into the copies. This could happen, for example, if the remaining variables encoded information that could be traced back to *gender* or *race*, even in the absence of these data. Hence, we check that no other variable is correlated with these two.

In Fig. 7.1 we report the top ten ranked attributes in terms of their absolute-valued one-to-one correlation score with these two variables. At most, this correlation is equal to 0.18 in the case of *gender* and to 0.35 in the case of *race*. This means that the information coded in these two features is not mirrored elsewhere, *i.e.* the value of these features cannot be estimated from the remaining information. Thus, we can safely conclude that there will be no residual information left in the synthetic dataset after the removal of these attributes.

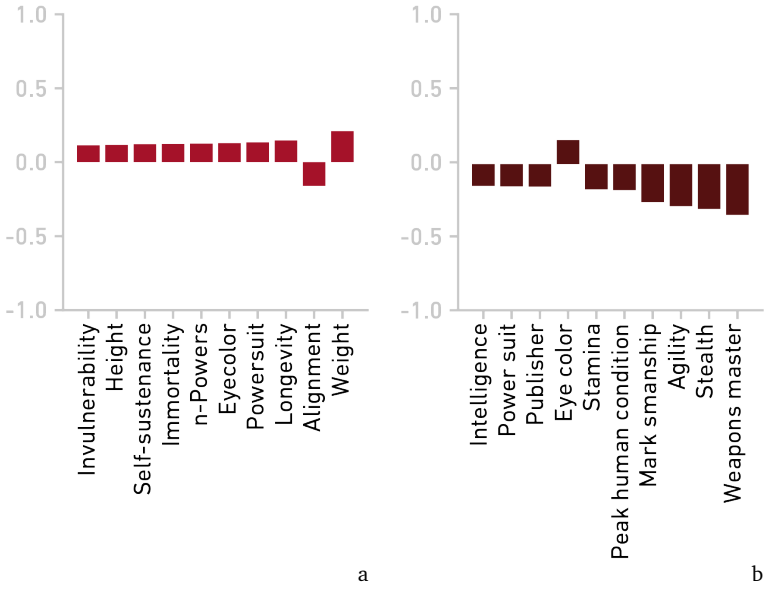


Fig. 7.1 Top ten ranked attributes in terms of their one-to-one correlation coefficient with (a) gender and (b) race. The ranking is computed taking the absolute value.

7.6.1 Evaluating the copy performance

In Table 7.2 we report the averaged results for the empirical fidelity error over both the original and the synthetic datasets and the copy accuracy for the lower dimensionality copies. The original network yields an accuracy of of 0.65. The mean copy accuracy is equal to 0.65 ± 0.01, averaged over all runs. This means that the loss in accuracy we incur when substituting the original with the copy in the original data space is negligible in most cases. Conversely, the empirical fidelity error measured over the synthetic dataset, which corresponds to the residual error of learning an optimal copy model for the synthetic data points, is equal to 0.059 ± 0.003. This means that the second step of the single-pass approach, the parameter tuning, is properly performed. Finally, the mean empirical fidelity error evaluated over the original test data is 0.22 ± 0.01. This value corresponds to the level of agreement between original and copy when generalizing the prediction to new unobserved points in the training data environment. The value of this last error is specially relevant when understanding how the copy is able to replicate the original decision function in the absence of sensitive information. Removal of the protected attributes from the synthetic dataset results in a certain shift in the learned decision function. To better understand how this shift impacts the classification of individual data points, we further study the value of the reported performance metrics over the different population groups.

$\mathcal{A}_{\mathcal{O}}$	$\mathcal{R}_{\text{emp}}^{\mathcal{F},\mathscr{X}}$	$\mathcal{R}_{\text{emp}}^{\mathcal{F},\mathscr{D}}$	$\mathcal{A}_{\mathcal{C}}$
0.65	0.059 ± 0.003	0.22 ± 0.01	0.65 ± 0.01

Table 7.2 Performance metrics averaged over all runs.

7.6.2 Evaluating bias reduction

In Table 7.3 we report the mean accuracy by *gender* for original and copy. We observe that there exist significant differences in the predictive accuracy of the original model across the different gender populations. In particular, *male* superheroes are more usually wrongly classified than *female*. This is a clear sign of the presence of bias in the trained classifier. Independently of whether the decision relies on the *gender* attribute, it does affect the different groups in a disparate form. When compared to the results obtained by the copy, we observe that the disparity among *male* and *female* groups is notably reduced in the latter. In particular, the difference in accuracy among the groups goes from 0.09 for the original to 0.03 for the copy. As a result, the decisions output by the copy have a more balanced impact on individuals in both populations.

	Original	**Copy**
female	0.73	0.69
male	0.64	0.66

Table 7.3 Accuracy by gender groups for original and copy.

	Original	**Copy**
human	0.78	0.76
mutant	0.75	0.75
robot	0.67	0.5
extraterrestial	0.25	0.5
other	0.59	0.64

Table 7.4 Accuracy by race group for original and copy.

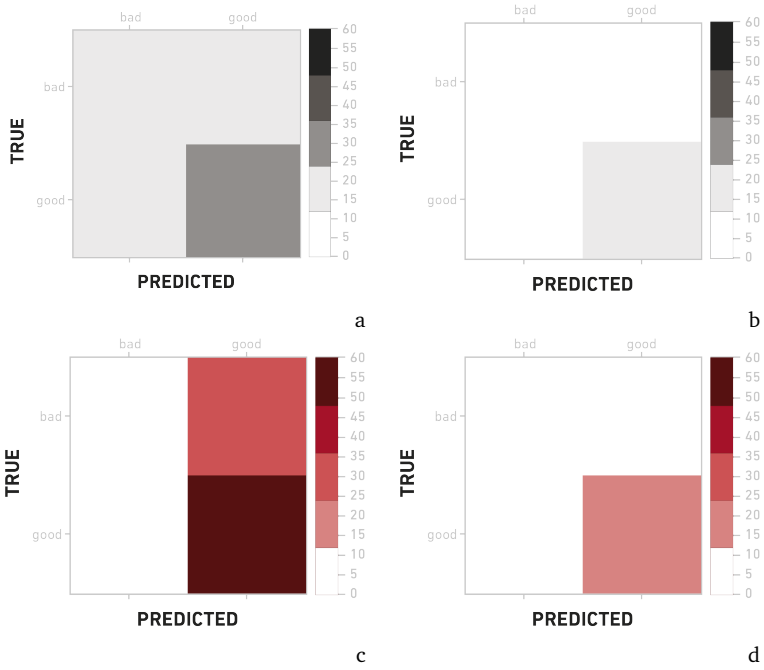


Fig. 7.2 Confusion matrices for male (left) and female (right) gender groups for (a) and (b) the original model and (c) and (d) the copy.

To better characterize these results, we further provide the confusion matrices for the two gender groups. Figures 7.2(a), 7.2(b), 7.2(c) and 7.2(d) show the relation between true and false positives and negatives for data in groups *male* and *female* for original and copy, respectively. In the case of the *male* group, the number of true positives increases for the copy, while the opposite effect is seen for the case of *female*. The net effect of this is the balancing of the predictive accuracy between both groups.

Importantly, these results are also observed for the case of the *race* attribute, although less strongly. As shown in Table 7.4, the mean accuracy by group tends to balance for the copy. This is clearly observed in the two majority classes, namely *humans* and *mutants*. In the minority classes we also see the benefits of our proposed solution for the group *extraterrestial*, which is more often incorrectly classified by the original.

We conclude that this simple approach results in a certain mitigation of the bias for the *gender* attribute. Equivalently, for the *race* attribute we also observe a certain improvement. However, given the higher level of granularity for this case, it is harder to evaluate the actual impact of our proposed approach on the resulting decision boundary.

Lessons learned

- The increasing use of machine learning models in high-stakes decisions is, to a certain extent, explained by the belief that this technology provides an objective mechanism to makes decisions. However, evidence has shown that because models learn from data that represents the outcome of previous human decisions, they replicate and sometimes even amplify the already existing biases.
- In this use case we discuss how differential replication through copying can be used to mitigate the bias learned by a trained classifier by removing the sensitive attributes and forcing a shift in the learned decision boundary.
- We use a public dataset and train a binary classifier to predict superhero alignment. This classifier exploits information about both the *gender* and the *race* of each superhero and outputs predictions that have a disparate impact on the different groups.
- We mitigate this bias by removing the two sensitive variables from the synthetic dataset and building copies that replicate the original decision behavior while operating in a lower dimensionality domain.
- This is possible provided that the remaining attributes do not leak sensitive information into the copies through unwanted correlations. In absence of such correlations, removal of the protected variables ensures that the copy does not have access to the sensible information.
- Our results pave the way for more complex treatments of the fairness problem by means of copies. Following our approach, one could, for example, endow copies with fairness metrics such as equity of learning or equality of odds, so that the resulting classifier retain the original accuracy while at the same time optimizing for this new measures.

Conclusions

This dissertation is the result of an industrial doctoral thesis. As such, it develops an approach to the field of machine learning that lies at the intersection of the academy and the industry. Its aim is to derive new theoretical knowledge that can be transferred to the appropriate stakeholders to provide solutions to specific needs of the industry. This transference requires methods with a good trade-off between quality and practicality. Conditions that are often hard to obtain in practice.

In general, there exists a huge gap between the theoretical postulates of machine learning and their practical implementation. This gap is not exclusive to this field. Yet, it is particularly evident in this case, given the rapid pace with which machine learning evolves. A pace most companies struggle to keep up with. Commercial machine learning models are delivered through stringent production infrastructures and their design and deployment are costly and largely constrained by a rapidly changing environment. The main consequence of this is the lack of adaptability of both the design processes and the resulting products or services. As a result of this deficiency, the industrial deployment of machine learning today is far from being sustainable.

The following are the main contributions of this thesis to tackling this issue from both an academic and an industrial perspective. At the end of this section, we also present the main lines of work that follow from the ideas here presented.

Academic contributions

We have formalized the problem of environmental adaptation in machine learning, which refers to situations where changes in the constraints of a task requires evolving to a new form of knowledge representation or hypothesis space. We have discussed different approaches to this problem under the framework of differential replication. This notion delves with ensuring adaptability of a predictive system as it evolves throughout its lifespan by reusing the knowledge acquired from generation to generation. In

implementing differential replication in practice, we have provided a preliminary categorization of the different inheritance mechanisms available and the scenarios they each refer to. In scenarios where access to the models and their training data are restricted, we have defined inheritance by copying as the process by which the decision behavior of a trained model can be projected into a new hypothesis space that meets the requirements of a stringent environment. The main body of work of this thesis has focused on deriving the mathematical background for copying and discussing the consequences of this formulation in practice. We have introduced an evaluation framework to measure the performance of copying under different assumptions and shown its validity through a series of well-established experiments.

Industrial constributions

We have demonstrated the value of copying under the framework of machine learning accountability. Accountability involves taking the necessary measures to report and justify the shortcomings of machine learning solutions, as well as to establish the legal liability of the people or organizations who are responsible from any negative impact derived from them. We have introduced the notion of actionable accountability, a process composed of two different stages: the risk-based auditing, which refers to an obligation to report all the potential risks that may be derived from a commercial machine learning solution; and risk mitigation, which includes mechanisms to mitigate the identified risks and ultimately remove them from the system. In particular, we have discussed the role of copying as one of such mitigation mechanisms. Finally, we have demonstrated how copying can be used to mitigate the risks related to interpretability and fairness in two real industrial machine learning solutions.

Future work

In ensuring sustainability of machine learning deployment, differential replication is a valuable technique which should be further studied both theoretically and in practice. Differential replication through copying allows us to enhance systems with higher layers of abstraction. A particularly relevant enhancement in this sense is adding causality features to regular models. This would enable a deeper comprehension of the problems at hand to provide more reliable predictions. Another possible enhancement is that of privacy. Copies could be specifically built to be privacy-preserving with respect to the original data attributes and instances, so that commercial solutions could more safely be disclosed to the general public without the risk of data leakage.

When it comes to enhancing the process of copying itself, future work should focus on further developing the dual approach. In moving on from the single-pass, the performance of copies would largely benefit from a more refined synthetic sample generation process. In particular, additional experiments are required to test this strategy in practice against different datasets and prediction tasks. Along these same

lines, future research should also focus on adapting the copying framework to multilabel environments. In its current form, differential replication through copying is feasible in both binary and multiclass scenarios. Yet extension to multilabel problems would ensure its applicability also to relevant fields such a that of medical diagnosis.

Finally, additional mechanisms should be devised with the aim of ensuring sustainability in industrial machine learning deployment. Sustainability implies a cost-effective perspective to develop commercial applications that are profitable. It also entails a responsible use of human, time and material resources. Finally, sustainability also delves with safety. A sustainable deployment of machine learning is one where no harm is derived to companies, users or the society as a whole and which improves previous standards to improve our overall quality of life.

Appendices

Appendix A
Comparison of sampling strategies

A.1 Methods

We explore different methods for generating synthetic datasets that enable copying in different scenarios. These datasets should be such that the original feature space is adequately explored and the learned decision boundary well-represented. We propose two sampling techniques: an exploration-exploitation policy using a Boundary sampling model and a modified fast Bayesian sampling algorithm that uses Gaussian processes to reduce the uncertainty around the decision function. In what follows we describe these two approaches. For comparative purposes, we also use a modified Jacobian sampling strategy based on [184] and random sampling from a uniform distribution defined on original attribute domain.

A.1.1 Boundary sampling

This method combines uniform exploration with a certain amount of exploitation. The main idea is to conduct a targeted exploration of the space until the decision boundary is found. The area around the boundary is then exploited by alternatively sampling at both sides. Often, classifiers do not learn a single decision boundary, but multiple. Hence, different decision regions are to be expected. This process is therefore repeated several times to ensure a proper coverage of the whole decision space. The detailed algorithm for Boundary sampling is shown in Alg. 2.

We begin by generating samples uniformly at random until we find a sample whose predicted class

Algorithm 2 Boundary Sampling(**int** N, **Classifier** $f_{\mathcal{O}}$)

$Z \leftarrow \emptyset$
while $|Z| < N$ **do**
  $z_a \sim \mathit{Uniform}(\mathcal{X}), \quad y_a \leftarrow f_{\mathcal{O}}(z_a)$
  repeat ▷ Search samples with different labels
    $z_b, y_b \leftarrow z_a, y_a$
    $z_a \sim \mathit{Uniform}(\mathcal{X}), \quad y_a \leftarrow f_{\mathcal{O}}(z_a)$
    $Z \leftarrow Z \cup \{z_a, y_a)\}$
  until $y_a \neq y_b$
  while $||z_b - z_a||_2 \geq \varepsilon$ **do** ▷ binary search
    $z_c \leftarrow (z_a + z_b)/2, \quad y_c \leftarrow f_{\mathcal{O}}(z_c)$
    $Z \leftarrow Z \cup \{(z_c, y_c)\}$
    $z_b, y_b \leftarrow z_c, y_c$ **if** $y_c \neq y_a$ **else** $z_a, y_a \leftarrow z_c, y_c$
  end while
  $T \leftarrow \{(z_c, y_c) : \text{repeated } \mathcal{I} \text{ times}\}$
  while $T \neq \emptyset$ and for no more than $\mathcal{T}$ iterations **do**
    $(z, y) \in T, \quad T \leftarrow T \setminus \{(z, y)\}$
    $u \leftarrow u/\|u\|_2, \; u \sim \mathbb{N}(0, I_d)$ ▷ random direction
    for $\mathcal{N}$ times **do**
      for $\alpha \in \{1, 0.9, \cdots, -0.9, -1\}$ and while $f_{\mathcal{O}}(z + \lambda v) = y$ **do**
        $v \leftarrow \alpha \cdot u + w$ with w a random vector s.t. $w \perp u, ||v||_2 = 1$
      end for
      $z \leftarrow x + \lambda v, \quad y \leftarrow f_{\mathcal{O}}(z)$
      $Z \leftarrow Z \cup \{(z, y)\}, \quad u \leftarrow v$
      every Poiss(λ') points do $T \leftarrow T \cup (z, y)$
    end for
  end while
end while
return S

label differs from the others. We then proceed to do a binary search in the line that connects this sample with the one obtained right before. This binary search is stopped when a pair of points $(\boldsymbol{z}_a, \boldsymbol{z}_b)$ is found such that $\|\boldsymbol{z}_a - \boldsymbol{z}_b\|_2 < \varepsilon$ and $f_{\mathcal{O}}(\boldsymbol{z}_a) \neq f_{\mathcal{O}}(\boldsymbol{z}_b)$ for $f_{\mathcal{O}}$ the original classifier and ε a given tolerance. This is, points to which the original classifier assigns different class labels and which are located at a distance from the boundary no larger than ε. We take one of these two points $\boldsymbol{z}$ as a starting point and draw samples at a constant step distance λ in the direction of its unitary random vector. We stop when we obtain a new point $\boldsymbol{z}'$ such that $f_{\mathcal{O}}(\boldsymbol{z}) \neq f_{\mathcal{O}}(\boldsymbol{z}')$, and repeat the process.

The number of samples in the binary search increases with the logarithm of $1/\varepsilon$. The value of ε must be small compared to the boundary exploration step λ, which determines the Euclidean distance between two consecutive samples. The higher the value of λ, the faster the boundary will be covered with less resolution. If λ is small, a large proportion of the boundary may remain unexplored.

The above process results in a set of samples that alternate the two sides of the decision boundary, with distance to the boundary bounded by λ. These samples define a one-dimensional curve: a thread. A thread contains a predefined number of steps $\mathcal{N}$, which at any given time depends on the number of

samples already generated. Threads are stopped when out of range or when no other samples are found in the given direction. To ensure a good coverage of the space, we allow $\mathcal{T}$ threads to be created from each point $\boldsymbol{z}$. When this number is reached, a new binary search starts. For high values of $\mathcal{T}$, the boundary is well sampled in certain areas, but many regions are left uncovered. For lower values of $\mathcal{T}$, we favour an exploration policy: a larger portion of the boundary is explored, albeit with less intensity.

We allow each individual thread to generate other threads with a frequency modelled by a Poisson distribution parameterized by λ'. This parameter controls the overall distribution of the different threads. For higher values of λ', exploration threads are more dispersed and samples in the vicinity of the boundary more spread. We perform $\mathcal{I}$ independent runs, increasing the maximum number of threads from run to run. Given a desired number of synthetic samples N, we generate half of them following the Boundary sampling algorithm and the other half using random sampling. The theoretical computational cost of this method is $\mathcal{O}(Nd)$.

A.1.2 Fast Bayesian sampling

While Boundary sampling ensures a good representation of the boundary neighborhood, it heavily relies on Random sampling to explore the remaining parts of the space. Hence, in a secondary approach, we propose *Bayesian Sampling*. Initially, the rationale behind this mechanism starts assigning a large uncertainty to the whole sampling domain. When a new sample is generated in a certain region, the uncertainty in that part of the space is reduced. The goal of this mechanism is to reduce the global uncertainty by guiding future sampling towards the most uncertain areas.

This method is based on Bayesian optimization, where the function to optimize is assumed to be a random process and samples are generated maximizing an acquisition function. We start by assigning a large uncertainty to the whole input space. Everytime we generate a new sample, we aim to reduce the global uncertainty by guiding future sampling towards the most uncertain areas.

Let us assume a Gaussian Process $g \sim \mathcal{GP}(0, k_{SE})$ $g \sim \mathcal{GP}(0, k_{SE})$ with mean 0 and a squared exponential kernel of the form

$$k_{SE}(\boldsymbol{z}, \boldsymbol{z}') = \sigma^2 e^{-\frac{||\boldsymbol{z}-\boldsymbol{z}'||_2^2}{2l^2}}, \tag{A.1}$$

for length scale l and variance σ^2. Every realization g_i of the stochastic process g is such that $g_i : \mathcal{X} \to \mathbb{R}$. In particular, we treat the original classifier $f_{\mathcal{O}}$ as one of such realizations[1]. Our objective is to find a set of points $\boldsymbol{Z}$ such that the function $\lfloor \mathbb{E}[g^{\boldsymbol{Z}}] \rceil$ [2], where $g^{\boldsymbol{Z}} = (g \,|\, g(\boldsymbol{z}) = f_{\mathcal{O}}(\boldsymbol{z}), \forall\, \boldsymbol{z} \in \boldsymbol{Z})$, is similar enough

[1] The class labels output by $f_{\mathcal{O}}$ take only discrete values, whereas a realization of the Gaussian Process defined as above gives points in the set of real numbers. However, there exist realizations of g as close to $f_{\mathcal{O}}$ as desired. Thus, it is reasonable to consider $f_{\mathcal{O}}$ to be a realization of g.

[2] $\lfloor x \rceil$ rounds x to the nearest integer.

Algorithm 3 Bayesian Sampling(**int** N, **Classifier** $\mathcal{O}$)

while $|Z| < N$ **do**
  $g^Z \leftarrow g \mid g(z) = y \; \forall \, (z, y) \in Z$ ▷ A posteriori Gaussian process
  $x \leftarrow \text{argmax}_{z \in \mathcal{D}} \, f(g^Z, z)$
  $y \leftarrow f_{\mathcal{O}}(z)$
  $Z \leftarrow Z \cup (z, y)$
end while
return Z

to $f_{\mathcal{O}}$. With this aim in mind, we propose an acquisition function $f : \mathcal{GP} \times \mathcal{X} \rightarrow \mathbb{R}$ of the form

$$f(g, \boldsymbol{z}) = \mathbb{V}_{ar}[g(\boldsymbol{z})] \cdot [1 + \tau \cdot \mathit{frac}(\mathbb{E}[g(\boldsymbol{z})])^2 (1 - \mathit{frac}(\mathbb{E}[g(\boldsymbol{z})]))^2], \tag{A.2}$$

where $\mathit{frac}(\boldsymbol{z})$ stands for the decimal part of $\boldsymbol{z}$. The first term involves the variance of the process. When choosing the next sample, it gives a higher priority to points with a bigger variance, *i.e.* located in a region where we have little information about $f_{\mathcal{O}}$. The second term focuses on exploring the boundary. Its maximum is located at 0.5. This encourages refining areas close to a transition between classes. Parameter τ governs the trade-off between the two terms. For high values of τ, more samples are generated in the vicinity of the boundary, when compared with the remaining sampling domain. The algorithm for raw Bayesian sampling is displayed in Alg. 3.

As it is, finding the *a posteriori* distribution of the Gaussian Process has a high computational cost due to the need to compute the mean and the covariance matrix. Moreover, this cost is increased when optimizing for the maximum of the acquisition function. The total cost is roughly $\mathcal{O}(dN^3)$. In its raw form, Bayesian sampling is a very slow process. To overcome this limitation, we propose a faster version, where we find the *a posteriori* Gaussian distribution in a single optimization, by limiting the number of samples used to compute the *a posteriori* process to b. The lower the value of b, the faster the algorithm converges, but also the less accurate it is. Indeed, while this approach is notably faster, it is not warranted to find an optimal solution.

When computing the maximum of the acquisition function, we set the starting point to $z_0 \sim \mathit{Uniform}(\mathcal{X})$ and the total number of iterations to $\mathcal{N}$, the number of random samples used to compute the first *a posteriori* distribution. In general, the acquisition function is non-convex, so that we can identify the next point to sample to be $\boldsymbol{z} = \text{argmax}_{\boldsymbol{z} \in \mathcal{V}(\boldsymbol{z}_0) \subseteq \mathcal{D}} \, f(g^{\boldsymbol{Z}}, \boldsymbol{z})$ for $\mathcal{V}(\boldsymbol{z}_0)$ a neighbourhood of $\boldsymbol{z}_0$ not necessarily open. The number of points generated by this Gaussian process is limited by a slowness factor sf, defined as the inverse of the fraction of samples generated from the gaussian process with respect to those used to compute the *a posteriori*, without recalculation. This factor exploits the fact that the optimization of the acquisition function only finds local minima, i.e. it does not generate the same samples. A low value makes the algorithm faster, but less precise. Its maximum value is set to b.

These modifications allow us to sample the acquisition function without having to constantly recalculate the posterior. This is possible because we find "local" optima. In all cases, the number of points calculated without reoptimizing the Gaussian process is proportional to the number of samples used to

Algorithm 4 Fast Bayesian Sampling(**int** N, **Classifier** $f_{\mathcal{O}}$)

$Z \leftarrow \{(z, f_{\mathcal{O}}(z)) \mid 10 \times z \sim \mathit{Uniform}(\mathcal{D})\}$
while $|Z| < N$ **do**
  if $|Z| \leq b$ **then**
    $Z_r \leftarrow Z$
  else ▷ Limit to b the number of samples to calculate the posterior distribution
    $Z_r \subseteq Z \; s.t. \; |Z_r| = b$
  end if
  $g^{Z_r} \leftarrow g \mid g(z) = y \, \forall (z, y) \in Z_r$ ▷ A posteriori Gaussian process
  $T \leftarrow \emptyset$
  repeat
    $z_0 \sim \mathit{Uniform}(\mathcal{D})$
    $z \leftarrow \text{argmax}_{z \in \mathcal{V}(z_0) \subseteq \mathcal{D}} \, f(g^{Z_r}, z)$
    $y \leftarrow f_{\mathcal{O}}(z)$
    $T \leftarrow T \cup (z, y)$
  until $|T| = \lfloor |Z_r| / \text{sf} \rceil$ ▷ sf: slowness factor
  $Z \leftarrow Z \cup T$
end while
return S

optimize the previous Gaussian process. The full algorithm for Fast Bayesian sampling is depicted in Alg. 4. This method has linear complexity with respect to the number of samples, roughly $\mathcal{O}(Ndb^2)$. Unless otherwise specified, we use the term Bayesian sampling to refer to this faster version.

A.1.3 Adapted Jacobian sampling

Finally, we present an adapted version of the Jacobian-based Dataset Augmentation algorithm proposed in [184]. We refer the reader to this reference for an in-depth description of this method. We here only discuss the main characteristics, as well as our added modifications. Our modified algorithm for Jacobian sampling is shown in Alg. 5.

The original algorithm starts with a set of samples specifically chosen to be similar to the ones in the original training set. These samples are used to train a preliminary substitute model with the same architecture of the original. The heuristics then proceeds by generating new samples in the directions in which the original model's outputs vary. These directions are identified by evaluating the sign of the Jacobian matrix dimension corresponding to the predictions output for different input points. The substitute model is then re-trained by iteratively applying the data augmentation technique on the initial samples. Every time new labelled samples are generated, the substitute model is refined thanks to a better representation of the original decision boundary.

In our case, we allow no access to the training data instances and in turn start with a set of randomly generated samples. At each iteration, we train a general ANN classifier with the available samples. We define the architecture of this classifier to match that proposed by [184]. It consists of 4 hidden layers

Algorithm 5 Jacobian Sampling(**int** N, **Classifier** $f_{\mathcal{O}}$

$Z \leftarrow \{(z, f_{\mathcal{O}}(z)) \mid 10 \times z \sim \mathit{Uniform}(\mathcal{D})\}$ ▷ Initial set of samples
while $|Z| < N$ **do**
  $T \leftarrow$ random subset of Z s.t. $|T| = \mathcal{T}$
  $Z \leftarrow Z \cup$ Bayesian Sampling$(\mathcal{T} + \mathcal{I}, T, f_{\mathcal{O}})$
  $f_{\mathcal{M}} \leftarrow$ trained MLP with Z
  $U \leftarrow$ random subset s.t. $|U| = \max(10, |Z|)/k$
  for $z \in U$ and $g \in \{\nabla p(f_{\mathcal{O}}(z) = i) : i \in |k|$ **do**
    $g \leftarrow \frac{g}{||g||}$ or $\frac{\text{sign}(g)}{||\text{sign}(g)||}$ if $||g|| \approx 0$
    if $p \mod \mathcal{P} = 0$ **then**
      $u \leftarrow z + \lambda(g + \mathcal{N}(0, 0.25))$
      $Z \leftarrow Z \cup \{(u, f_{\mathcal{O}}(u))\}$
    end if
  end for
end while
return Z

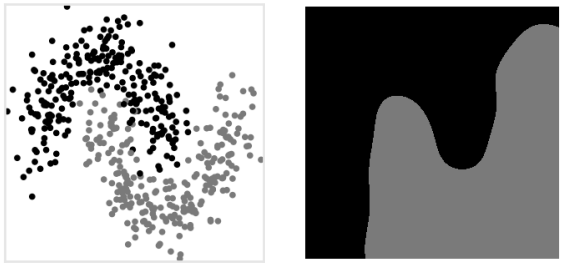


Fig. A.1 (a) Training dataset and (b) decision boundary learned by an SVM with a radial basis function kernel.

with sizes 32, 64, 200 and 200. Once this model is trained, we compute gradients with respect to each label for every sample. New samples are then taken making a step of length λ in the direction of the computed gradients from every available point. Parameter λ determines the distance between one sample and the next. If this value is low, the algorithm is slower in finding the boundary and more samples must be generated. A high value, however, results in a less precise search of the boundary. In cases where the norm of a gradient is close to 0, we avoid running into precision problems by using its sign. We also add Gaussian noise to the steps. The total number of steps is controlled by parameter $\mathcal{P}$, the number of steps performed with the same initial sample subset. For a fixed number of total samples N, a high value of $\mathcal{P}$ ensures we reach the boundary, but we could leave other regions unexplored.

Finally, to ensure we explore the whole feature space, at each iteration we also include a fix number of samples $\mathcal{I}$ obtained by means of Bayesian sampling. For large values of $\mathcal{I}$, samples are better distributed across the domain. However, the boundary might be less explored. We generate these samples for a fixed number of initial samples $\mathcal{T}$ for the *a posteriori* Gaussian process. If the value of $\mathcal{T}$ is high, the algorithm will produce few iterations of a lot of samples, resulting in a poorly covered space. On the contrary, if its value is low, more iterations are performed while less regions near the boundary are sampled. The

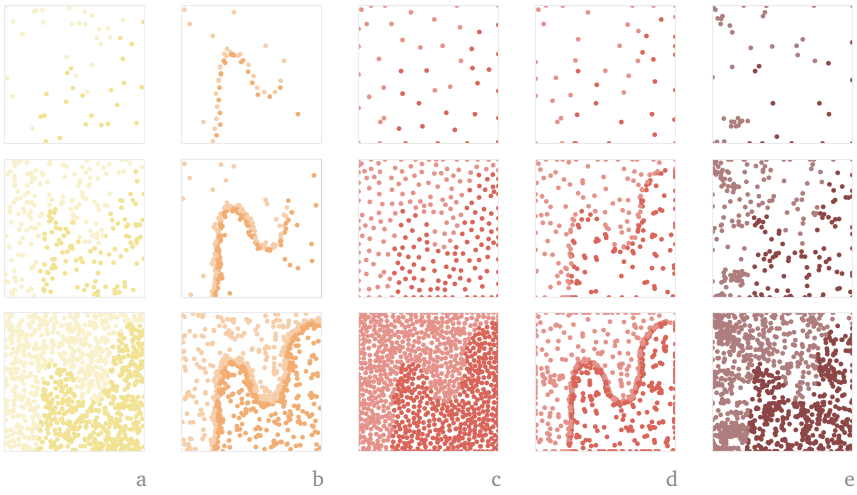


Fig. A.2 From top to bottom, synthetic datasets of sizes 50, 250 and 1000 generated using (a) Random sampling, (b) Boundary sampling, (c) Fast Bayesian sampling, (d) reoptimized Bayesian sampling and (e) adapted Jacobian sampling.

theoretical computational cost of this method is $\mathcal{O}(Ndk)$.

A.1.4 Intuition

In Fig.A.2 we provide an intuition of how the different methods perform on the toy example shown in Fig.A.1, for different number of samples. Fig.A.1(a) shows the original training data and Fig.A.1(b) the decision boundary learned by an ANN with a single hidden layer. Figs. A.2(a), (b), (c), (d) and (e) show the synthetic datasets generated using random sampling, Boundary sampling, fast Bayesian sampling, reoptimized Bayesian sampling and adapted Jacobian sampling, respectively.

When comparing results for Boundary sampling in Fig.A.2(d) to the case of exclusive random sampling in Fig.A.2(c), the boundary is sampled with more emphasis, although a large number of samples is required to properly cover this region. Indeed, the main advantage of random sampling is that it scatters the samples across the space with equal probability. However, this method is oblivious to the structures of interest, *i.e.* it retains no knowledge about the form of the decision boundary for future sampling steps.

Samples generated from Bayesian sampling, in Fig.A.2(c), are well distributed, without forming clusters. The importance given to the boundary on the acquisition function is not manifested in a big scale due to the simplifications. If we let the Gaussian process re-optimize several times, it correctly samples the boundary as well as the rest of the space, as shown in Fig.A.2(d). Finally, synthetic data points generated with the adapted Jacobian sampling, in Fig.A.2(e), form diagonal lines due to the use of the

sign of the gradient.

A.2 Experiments

We compare the different methods above on the 6 UCI datasets described in Table A.1. Overall, these datasets include a heterogeneous sample of binary and multiclass problems with varying dimensionalities. In all cases the data are defined over a real feature space. In order to train the original classifiers for each problem, we assume the data to be normally distributed in all cases and apply a linear transformation such that, when variables are not correlated, 0.99^d samples lay inside the $[0, 1]^d$ hypercube. We split the data into stratified 80/20 training and test sets. We use the training set to learn an artificial neural network with a single hidden layer of 5 neurons for each problem.

Dataset	Features	Classes	Samples
bank	16	2	3616
ilpd-indian-liver	9	2	466
magic	10	2	15216
miniboone	50	2	104051
seeds	7	3	168
synthetic-control	60	6	480

Table A.1 Description of the 6 selected datasets from the UCI machine learning repository.

We use the different methods above to generate synthetic sets of size 10^6 in the restricted input space $[0, 1]^d$. For comparative purposes, we also use random sampling. The choice of parameters for each method is specified in Table A.2. Because all algorithms produce new samples in an accumulative way, we also generate smaller sets by selecting the first j points. Finally, for evaluation purposes, we generate balanced reference sample sets $\mathscr{W} = \{\boldsymbol{w}_i, f_{\mathcal{O}}(\boldsymbol{w}_i)\}_{i=1}^{L}$ for each problem. These sets are comprised of $L = 10^7$ data points sampled uniformly at random in the $[0, 1]^d$ hypercube.

A.2.1 Evaluation metrics

We evaluate the different sampling strategies in terms of the performance of copies built on the resulting synthetic data. We build copies based on different architectures, including an ANN with the

Algorithm	Parameters
Boundary	$\varepsilon = 0.01 \quad \lambda = 0.05 \quad \lambda' = 5\ \mathcal{I} = \text{round}(2 + \log(N))$ $\mathcal{T} = \text{round}(8 + 4\log(N)) \quad \mathcal{N} = 5 + 2.6\log(N)$
Bayesian	$l = 0.5\sqrt{d} \quad \sigma^2 = 0.25k^2 \quad \tau = 10$ $sf = 20 \quad b = 1000 \quad \mathcal{N} = 10$
Jacobian	$\mathcal{I} = \min(100, \text{round}(5 + N/4)$ $\mathcal{T} = 50 \quad \lambda = 0.05 \quad \mathcal{P} = 5$

Table A.2 Parameters settings for the different algorithms.

same architecture as above (ANN), a logistic regression (LR), a decision tree classifier (DT) and a deeper ANN composed of 3 hidden layers with 50 neurons each (ANN2).

To compensate for the potential under-representation of one or more classes in the synthetic datasets, we measure the *balanced empirical fidelity error*, $R^{\mathcal{F}}_{emp,b}$, defined as

$$R^{\mathcal{F}}_{emp,b} = \frac{1}{sk}\sum_{j=1}^{k}\sum_{i=1}^{s}\mathbb{I}[f_{\mathcal{O}}(\boldsymbol{x}_i^j) = f_{\mathcal{C}}(\boldsymbol{x}_i^j)]$$

for k the number of classes and s the number of samples per class, so that $\boldsymbol{x}_i^j$ refers to sample i of class j.

In addition, we also report the original accuracy $\mathcal{A}_{\mathcal{O}}$ of the target classifier in each case, as well as the empirical fidelity error as defined in *Chapter 4*. We report metrics averaged over 10 repetitions, except for Bayesian sampling, for which we use 5 repetitions. We also provide the execution times of the different methods and sample sizes. All experiments are carried out in a single m4.16xlarge Amazon EC2 instance with 64 cores, 256 GB of RAM and 40 GB of SSD storage.

A.3 Results

In what follows we discuss our main experimental results. We first validate the generated reference sample set and then discuss the performance of the different methods, as well as their associated computational cost.

A.3.1 Reference set evaluation

We propose two checks to validate the reference sample sets $\mathscr{W}$. First, we fit the original architecture to the reference data and compute the balanced empirical fidelity error, $R^{\mathcal{F},\mathscr{W}}_{emp,b}$. As a complementary check, we also evaluate the empirical fidelity error over the original set, $R^{\mathcal{F},\mathscr{D}}_{emp,b}$. Results are shown in Table A.3. Most values are close to 0, which we take as an indication that the reference sample sets are a suitable baseline with which to compare our proposed sampling strategies. We note the exception of

	bank	ilpd	magic	miniboone	seeds	synthetic
$R^{\mathcal{F},\mathscr{W}}_{emp,b}$	0.023	0.080	0.001	0.009	0.020	0.010
$R^{\mathcal{F},\mathscr{D}}_{emp,b}$	0.021	0.385	0.001	0.168	0.000	0.000
$\mathcal{A}_{\mathcal{O}}$	0.8829	0.6410	0.8562	0.9119	0.8095	0.6750

Table A.3 Quality checks for the reference sample sets.

the *ilpd-indian-liver* dataset, for which we are not confident enough of our evaluation.

A.3.2 Algorithm evaluation

In Fig.A.3 we report the balanced empirical fidelity error for the different copy architectures, sampling strategies and datasets, measured on the reference sample sets. Plots show the 20, 50 and 80 percentiles of the multiple realizations.

Boundary sampling performs well for copies based on LR, since there is a lot of information to find the optimal decision hyperplane. However, Bayesian sampling performs comparably better with fewer samples *i.e.* displays the fastest growth. This is because it focuses on globally reducing the uncertainty during the first steps. In the case of LR, it learns fast until it reaches its capacity limit. For DTs, random sampling displays the best behavior. This may be because DTs work well when there is a sample in each region of the space in order to create the leaves. In high dimensionality, the coverage of $\mathcal{X}$ with DTs is costly, which seems to be in accordance with their slowly increasing score.

Copies based on ANN and especially on ANN2 achieve the best scores in general. We highlight the cases of *bank* and *ilpd-indian-liver* datasets for which the simpler ANN performs significantly worst, indicating that the use of matching architectures does not guarantee a good performance. This may happen because the characteristics of the problem change when using the synthetic dataset instead of the training data. This, together with the fact that the original architecture has just enough degrees of freedom to replicate the decision boundary, deters the copy from converging to the same solution.

In terms of the aggregated comparison among techniques, the adapted Jacobian sampling seems to perform the worst. This method generates linear structures which contain a large number of samples. As a result, it has a wide uncertainty band. For a large number of samples random sampling gathers the greatest number of victories. Closely behind, Boundary and Bayesian sampling are both reasonably similar in terms of their averaged performance.

A.3.3 Computational cost

Fig. A.4 shows the computational cost of the different algorithms for copies based on ANN2, i.e.

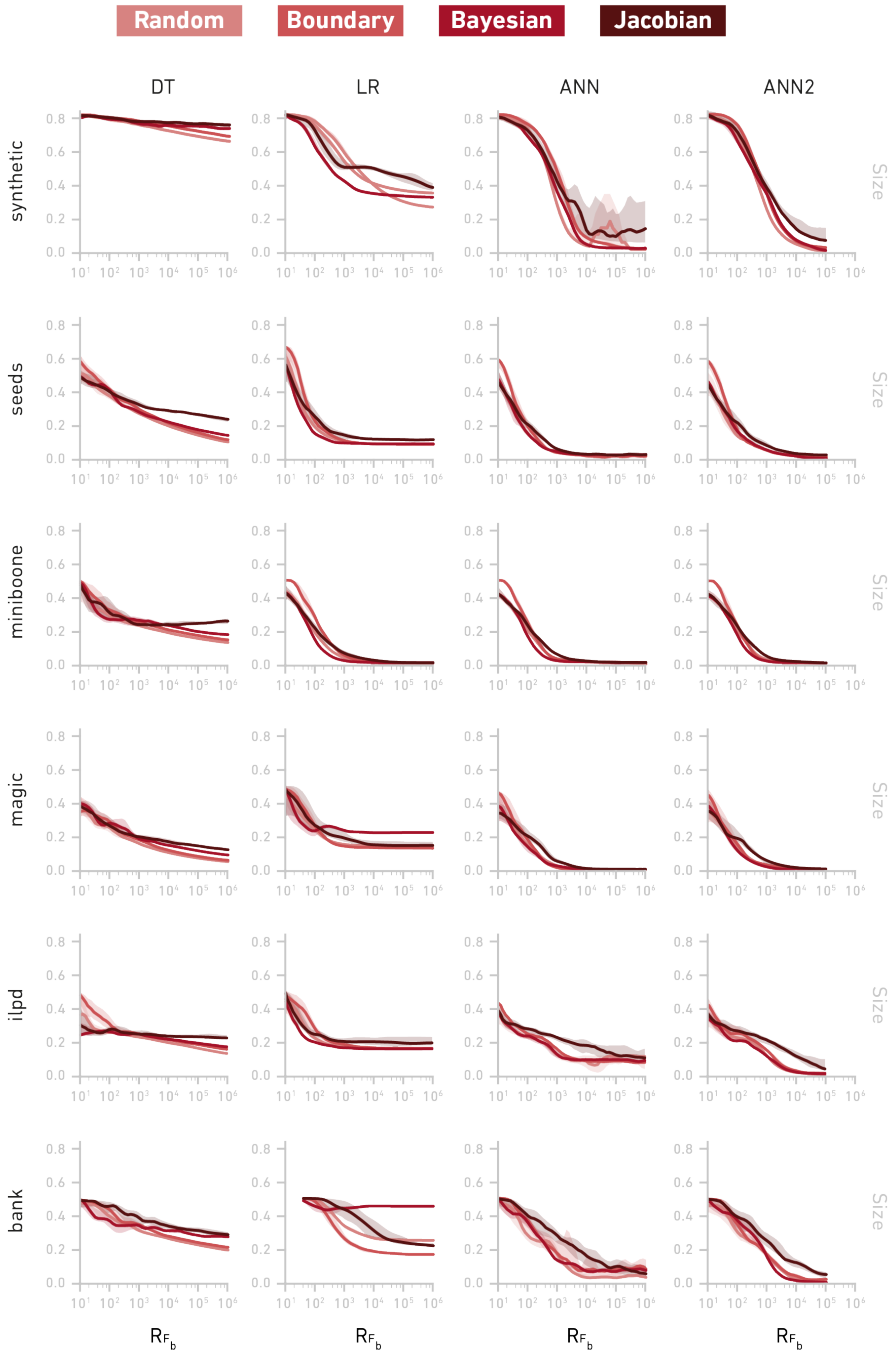


Fig. A.3 Median and 20-80 percentil band. The similarity axis starts from $1/k$ for k the number of classes: the expected score for a classifier that has not learned anything. For ANN2 models, we only show results for 10^5 samples, due to the high training times.

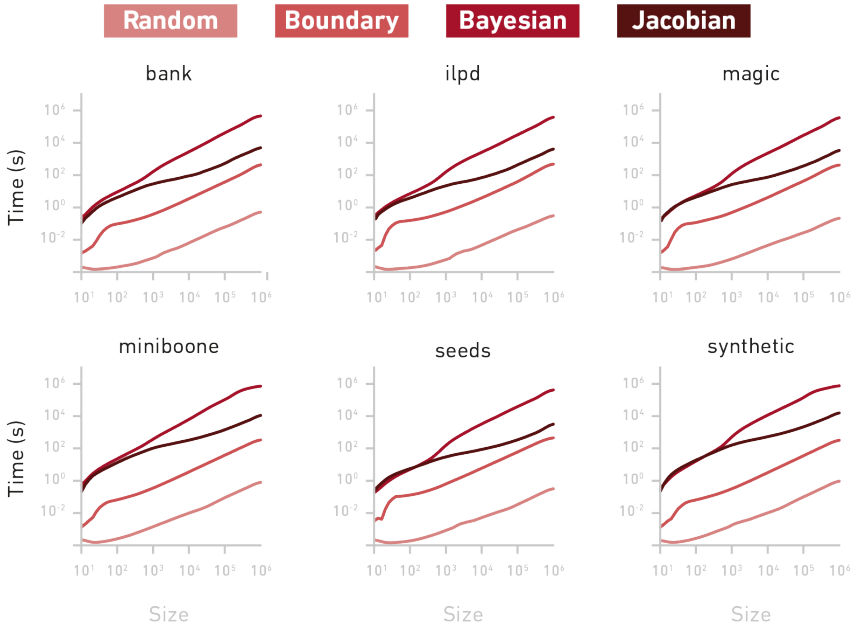


Fig. A.4 Execution time of the different sampling strategies as a function of dataset size.

the worst-case scenario. The execution time is asymptotically linear. Bayesian sampling and Random sampling are the slowest and the fastest, respectively. A great advantage of random sampling is its simplicity, and consequently its low cost. Its main drawback is, however, that it samples points with no regards to the form of the decision function or the resulting class distribution. In high dimensional problems, Boundary sampling may be a good compromise between time and accuracy. In the absence of any time constrain, however, Bayesian sampling ensures a more reliable exploration.

Bibliography

[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation* (Savannah, GA, USA, 2016), USENIX Association, Berkeley, CA, USA, pp. 265–283.

[2] Abebe, R., Barocas, S., Kleinberg, J., Levy, K., Raghavan, M., and Robinson, D. Roles for computing in social change. In *Proceedings of the Conference on Fairness, Accountability, and Transparency* (Barcelona, Spain, 2020), Association for Computing Machinery, New York, NY, USA, pp. 252–260.

[3] Adler, P., Falk, C., Friedler, S., Nix, T., Rybeck, G., Scheidegger, C., Smith, B., and Venkatasubramanian, S. Auditing black-box models for indirect influence. *Knowledge Information Systems 54*, 1 (2018), 95–122. doi: 10.1007/s10115-017-1116-3.

[4] Aguilar, G., Ling, Y., Zhang, Y., Yao, B., Xing, Fan, X., and Guo, C. Knowledge distillation from internal representations. arXiv:1910.03723, 2019. [Online] Available from: https://arxiv.org/abs/1910.03723.

[5] Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. Concrete problems in AI safety. arXiv:1606.06565, 2016. [Online] Available from: https://arxiv.org/abs/1606.06565.

[6] Andrews, R., Diederich, J., and Tickle, A. B. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge Based Systems 8*, 6 (1995), 373–389. doi: 10.1016/0950-7051(96)81920-4.

[7] Angluin, D. Queries and concept learning. *Machine Learning 2*, 4 (1988), 319–342. doi: 10.1023/A:1022821128753.

[8] Angwin, J. Make algorithms accountable. The New York Times, August 2016. [Online] Available from: https://www.nytimes.com/2016/08/01/opinion/make-algorithms-accountable.html.

[9] Angwin, J., Larson, J., Mattu, S., and Kirchner, L. Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks. ProPublica, 2016. [Online] Available from: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.

[10] Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. Stronger generalization bounds for deep nets via a compression approach. In *Proceedings of the 35th International Conference on Machine Learning* (Stockholm, Sweden, 2018), vol. 80, Journal of Machine Learning Research, New York, NY, USA, pp. 254–263.

[11] Atlas, L., Cohn, D., Ladner, R., El-Sharkawi, M. A., and Marks, R. J. Training connectionist networks with queries and selective sampling. In *Proceedings of the 3rd International Conference on Neural Information Processing Systems* (Denver, CO, USA, 1990), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 566–573.

[12] Ba, L. J., and Caruana, R. Do deep nets really need to be deep? In *Proceedings of the 27th International Conference on Neural Information Processing Systems* (Montreal, Canada, 2014), vol. 2, MIT Press, Cambridge, MA, USA, p. 2654–2662. doi: 0.5555/2969033.2969123.

[13] Bagherinezhad, H., Horton, M., Rastegari, M., and Farhadi, A. Label refinery: Improving imagenet classification through label progression. arXiv:1805.02641, 2018. [Online] Available from: https://arxiv.org/abs/1805.02641.

[14] Barocas, S., and Boyd, D. Engaging the ethics of data science in practice. *Communications of the ACM 60*, 11 (2017), 23–25. doi: 10.1145/3144172.

[15] Barocas, S., Crawford, K., Shapiro, A., and Wallach, H. The problem with bias: Allocative versus representational harms in machine learning. In *Proceedings of the 9th Annual Conference of the Special Interest Group for Computing, Information and Society* (Philadelphia, PA, USA, 2017), Society for the History of Technology, Eindhoven, The Netherlands.

[16] Barocas, S., Hardt, M., and Narayanan, A. *Fairness and Machine Learning*. fairmlbook.org, 2019. [Online] Available from: http://www.fairmlbook.org.

[17] Barocas, S., and Selbst, A. D. Big data's disparate impact. *California Law Review 104*, 3 (2016), 671–732. doi: 10.15779/Z38BG31.

[18] Barque, M., Martin, S., Vianin, J., Genoud, D., and Wannier, D. Improving wind power prediction with retraining machine learning algorithms. In *International Workshop on Big Data and Information Security* (Jakarta, Indonesia, 2018), IEEE Computer Society, Washington, DC, USA, pp. 43–48. doi: 10.1109/IWBIS.2018.8471713.

[19] Barreno, M., Nelso, B., Sears, R., Joseph, A. D., and Tygar, J. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security* (Taipei, Taiwan, 2006), Association for Computing Machinery, New York, NY, USA, pp. 16–25. doi: 10.1145/1128817.1128824.

[20] Barreno, M., Nelson, B., Joseph, A. D., and Tygar, J. D. The security of machine learning. *Machine Learning 81*, 2 (2010), 121–148. doi: 10.1007/s10994-010-5188-5.

[21] Bastani, O., Kim, C., and Bastani, H. Interpreting black-box models via model extraction. arXiv:1705.08504, 2018. [Online] Available from: https://arxiv.org/pdf/1705.08504.pdf.

[22] Begoli, E., Bhattacharya, T., and Kusnezov, D. The need for uncertainty quantification in machine-assisted medical decision making. *Nature Machine Intelligence 1* (2019), 20–23. doi: 10.1038/s42256-018-0004-1.

[23] Berk, R., Heidari, H., Jabbari, S., Kearns, M., and Roth, A. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research* (2018). doi: 10.1177/0049124118782533.

[24] Bhattacharya, B., Mukherjee, K., and Toussaint, G. T. Geometric decision rules for instance-based learning algorithms. In *Proceedings of the First International Conference on Pattern Recognition and Machine Intelligence* (Kolkata, India, 2005), Springer, Berlin, Heidelberg, p. 60–69. doi: 10.1007/11590316_9.

[25] Bhattacharya, B., Poulsen, R., and Toussaint, G. Application of proximity graphs to editing nearest neighbor decision rule. In *International Symposium on Information Theory* (Santa Monica, CA, USA, 1981), IEEE Computer Society, Washington, DC, USA.

[26] Bianco-Vega, R., Hernandez-Orallo, J., and Ramirez-Quintana, M. J. Knowledge acquisition through machine learning: Minimising expert's effort. In *Proceedings of the 4th International Conference on Machine Learning and Applications* (Los Angeles, CA, USA, 2005), IEEE Computer Society, Washington, DC, USA, pp. 49–54. doi: 10.1109/ICMLA.2005.45.

[27] Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases* (Prague, Czech Republic, 2013), Springer, Berlin, Heidelberg, pp. 387–402.

[28] Biggio, B., Nelson, B., and Laskov, P. Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning* (Edinburgh, Scotland, 2012), Omnipress, Madison, WI, USA, p. 1467–1474. doi: 10.5555/3042573.3042761.

[29] Biggio, B., and Roli, F. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition 84* (2018), 317–331. doi: 10.1016/j.patcog.2018.07.023.

[30] Binns, R. Algorithmic accountability and public reason. *Philosophy & technology 4*, 31 (2018), 543–556. doi: 10.1007/s13347-017-0263-5.

[31] Blodgett, S., Barocas, S., Daumé III, H., and Wallach, H. Language (technology) is power: A critical survey of" bias" in nlp. arXiv:2005.14050, 2016. [Online] Available from: https://arxiv.org/abs/2005.14050.

[32] Bologna, G., and Hayashi, Y. A comparison study on rule extraction from neural network ensembles, boosted shallow trees, and svms. *Applied Computational Intelligence and Soft Computing* (2018), 1–20. doi: 10.1155/2018/4084850.

[33] Bolukbasi, T., Chang, K. W., Zou, J., Saligrama, V., and Kalai, A. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Proceedings of the 29th International Conference on Neural Information Processing Systems* (Barcelona, Spain, 2016), MIT Press, Cambridge, MA, USA, pp. 4356–4364.

[34] Bostrom, N. Ethical issues in advanced artificial intelligence. In *Science Fiction and Philosophy: From Time Travel to Superintelligence*, S. Schneider, Ed. Wiley-Blackwell, New Jersey, NJ, USA, 2009, pp. 277–284.

[35] Bottou, L., and Le Cun, Y. Large scale online learning. In *Proceedings of the 17th International Conference on Neural Information Processing Systems* (Vancouver, Canada, 2004), MIT Press, Cambridge, MA, USA, pp. 217–234.

[36] Breiman, L. Bagging predictors. *Machine Learning 24*, 2 (1996), 123–140. doi: 10.1023/A:1018054314350.

[37] Breiman, L. Statistical modeling: The two cultures. *Statistical Science 16*, 3 (2001), 199–231. doi: 10.1214/ss/1009213726.

[38] Brendel, W., Rauber, J., and Bethge, M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *Proceedings of the 6th International Conference on Learning Representations* (Vancouver, BC, Canada, 2017).

[39] Brutzkus, A., Globerson, A., Malach, E., and Shalev-Shwartz, S. Sgd learns over-parameterized networks that provably generalize on linearly separable data. In *Proceedings of the 6th International Conference on Learning Representations* (Vancouver, Canada, 2017).

[40] Buciluǎ, C., Caruana, R., and Niculescu-Mizil, A. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Philadelphia, PA, USA, 2006), Association for Computing Machinery, New York, NY, USA, p. 535–541. doi: 10.1145/1150402.1150464.

[41] Buolamwini, J., and Gebru, T. Gender shades: Intersectional accuracy disparities in commercial gender classification *. In *Proceedings of Machine Learning Research; Conference on Fairness, Accountability and Transparency* (New York, NY, USA, 2018), pp. 1–15.

[42] Caliskan, A., Bryson, J. J., and Narayanan, A. Semantics derived automatically from language corpora contain human-like biases. *Science 356*, 6334 (2017), 183–186. doi: 10.1126/science.aal4230.

[43] Carr, B., and Bailey, N. Machine learning paper on explainability in predictive modeling. Tech. rep., Institute of International Finance, 2018. [Online] Available from: https://www.iif.com/Publications/ID/1423/Machine-Learning-Paper-on-Explainability-in-Predictive-Modeling.

[44] Celik, Z., Lopez-Paz, D., and McDaniel, P. Patient-driven privacy control through generalized distillation. In *2017 IEEE Symposium on Privacy-Aware Computing* (Washington, DC, USA, 2017), IEEE Computer Society, Washington, DC, USA, pp. 1–12. doi: 10.1109/PAC.2017.13.

[45] Che, Z., Purushotham, S., Khemani, R., and Liu, Y. Interpretable deep models for icu outcome prediction. In *Proceedings of the AMIA Annual Symposium* (Chicago, IL, USA, 2016), pp. 371–380.

[46] Chen, C., Seff, A., Kornhauser, A., and Xiao, J. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision* (Santiago, Chile, 2015), IEEE Computer Society, Washington, DC, USA, pp. 2722–2730.

[47] Cheng, X., Rao, Z., Chen, Y., and Zhang, Q. Explaining knowledge distillation by quantifying the knowledge. arXiv:2003.03622, 2020. [Online] Available from: https://arxiv.org/pdf/2003.03622.pdf.

[48] Cho, J., and Hariharan, B. On the efficacy of knowledge distillation. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision* (Seoul, Korea, 2019), IEEE Computer Society, Washington, DC, USA, pp. 4793–4801. doi: 10.1109/ICCV.2019.00489.

[49] Clark, A. The machine learning audit - crisp-dm framework. ISACA, 2018. [Online] Available from: https://www.isaca.org/Journal/archives/2018/Volume-1/Pages/the-machine-learning-audit-crisp-dm-framework.aspx?utm_referrer=.

[50] Cohn, D., Atlas, L., and Ladner, R. Improving generalization with active learning. *Machine Learning 15*, 2 (May 1994), 201–221. doi: 10.1023/A:1022673506211.

[51] Commission, E. U. Legislation. OJ, 2016.

[52] Craven, M., and Shavlik, J. W. Using sampling and queries to extract rules from trained neural networks. In *Proceedings of the 11th International Conference on Machine Learning* (New Brunswick, NJ, USA, 1994), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 37–45.

[53] Craven, M. W., and Shavlik, J. W. Learning symbolic rules using artificial neural networks. In *Proceedings of the 10th International Conference on Machine Learning* (Macau, China, 1993), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 73–80. doi: 10.1016/b978-1-55860-307-3.50016-2.

[54] Craven, M. W., and Shavlik, J. W. Extracting tree-structured representations of trained networks. In *Proceedings of the 8th International Conference on Neural Information Processing Systems* (Denver, CO, USA, 1995), MIT Press, Cambridge, MA, USA, p. 24–30.

[55] Crawford, K. The hidden biases in big data. Harvard Business Review, 2013. [Online] Available from: https://store.hbr.org/product/the-hidden-biases-in-big-data/H00ADR.

[56] Csurka, G. Domain adaptation for visual applications: A comprehensive survey. arXiv:1702.05374, 2017. [Online] Available from: https://arxiv.org/pdf/1702.05374.pdf.

[57] Cucker, F., and Smale, S. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society 39*, 1 (2002), 1–49. doi: 10.1090/S0273-0979-01-00923-5.

[58] Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems 2* (1989), 303–314. doi: 10.1007/BF02551274.

[59] Dada, E., Bassi, J., Chiroma, H., Abdulhamid, S., Adetunmbi, A., and Ajibuwa, O. E. Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon 5*, 6 (2019). doi: 10.1016/j.heliyon.2019.e01802.

[60] Dalvi, N., Domingos, P., Sumit, M., and Deepak Verma, S. Adversarial classification. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Seattle, WA, USA, 2004), Association for Computing Machinery, New York, NY, USA, p. 99–108. doi: 10.1145/1014052.1014066.

[61] Darwin, C. *On The Origin of Species by Means of Natural Selection, or Preservation of Favoured Races in the Struggle for Life*. John Murray, London, UK, 1859.

[62] Datta, A., Tschantz, M., and Datta, A. Automated experiments on ad privacy settings. In *Proceedings of the 15th Privacy Enhancing Technologies Symposium* (Philadelphia, PA, USA, 2015), De Gruyter, Warsaw, Poland, pp. 92–112.

[63] Dauphin, Y., and Bengio, Y. Big neural networks waste capacity. arXiv:1301.3583, 2013. [Online] Available from: https://arxiv.org/pdf/1301.3583.pdf.

[64] Dawes, R., Faust, D., and Meehl, P. Clinical versus actuarial judgment. *Science 243*, 4899 (1989), 1668–1674. doi: 10.1126/science.2648573.

[65] Dekel, O., Gentile, C., and Sridharan, K. Selective sampling and active learning from single and multiple teachers. *Journal of Machine Learning Research*, 1 (2012), 2655–2697. doi: 10.5555/2503308.2503327.

[66] Demir, B., Persello, C., and Bruzzone, L. Batch-mode active-learning methods for the interactive classification of remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing 49*, 3 (2011), 1014–1031. doi: 0.1109/TGRS.2010.2072929.

[67] Demontis, A., Russu, P., Biggio, B., Fumera, G., and Roli, F. On security and sparsity of linear classifiers for adversarial settings. In *Structural, Syntactic, and Statistical Pattern Recognition*, A. Robles-Kelly, M. Loog, B. Biggio, F. Escolano, and W. R., Eds. Springer, Berlin, Heidelberg, 2016, pp. 322–332.

[68] Deng, Y., Chen, K., Shen, Y., and Jin, H. Adversarial active learning for sequence labeling and generation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence* (Stockholm, Sweden, 2018), AAAI Press, Palo Alto, CA, USA, p. 4012–4018. doi: 10.5555/3304222.3304328.

[69] Diakopoulos, N., and Friedler, S. How to hold algorithms accountable. MIT Technology Review, November 2016. [Online] Available from: https://www.technologyreview.com/s/602933/how-to-hold-algorithms-accountable/.

[70] Diakopoulos, N., Friedler, S., Arenas, M., Barocas, S., Hay, M., Howe, B., Jagadish, H., Unswort, C., Sahuguet, A., Venkatasubramanian, S., Wilco, C., Yu, C., and Zevenbergen, B. Principles for accountable algorithms and a social impact statement for algorithms. Fairness, Accountability, and Transparency in Machine Learning, 2016. [Online] Available from: https://www.fatml.org/resources/principles-for-accountable-algorithms.

[71] Docherty, A., and Viort, F. *Better Banking: Understanding and Addressing the Failures in Risk Management, Governance and Regulation*. John Wiley & Sons, Ltd, New Jersey, NJ, USA, 2013. doi: 10.1002/9781118651315.

[72] Domingos, P. Knowledge acquisition from examples via multiple models. In *Proceedings of the 14th International Conference on Machine Learning* (Miami, FL, USA, 1997), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 98–106.

[73] Dong, B., Hou, J., LU, Y., and Zhang, Z. Distillation $\approx$ early stopping? harvesting dark knowledge utilizing anisotropic information retrieval for overparameterized neural network. arXiv:1910.01255, 2019. [Online] Available from: https://arxiv.org/pdf/1910.01255.pdf.

[74] Doshi-Velez, F., and Kim, B. Towards a rigorous science of interpretable machine learning. arXiv:1702.08608, 2017. [Online] Available fro: https://arxiv.org/pdf/1702.08608.pdf.

[75] Dua, D., and Graff, C. Uci machine learning repository. Tech. rep., University of California, Irvine, School of Information and Computer Sciences, 2017. [Online] Available from: https://archive.ics.uci.edu/ml/index.php.

[76] Dwork, C., Hardt, M., Pitassi, T., Reingold, O., and Zemel, R. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference* (Cambridge, MA, USA, 2012), Association for Computing Machinery, New York, NY, USA, pp. 214–226. doi: 10.1145/2090236.2090255.

[77] Eigen, D., Rolfe, J., Fergus, R., and LeCun, Y. Understanding deep architectures using a recursive convolutional network. arXiv:1312.1847, 2013. [Online] Available from: https://arxiv.org/pdf/1312.1847.pdf.

[78] Erhan, D., Bengio, Y., Courville, A., Manzagol, P., Vincent, P., and Bengio, S. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research 11* (2010), 625–660. doi: 10.5555/1756006.1756025.

[79] Escalera, S., Masip, D., Puertas, E., Radeva, P., and Pujol, O. Online error-correcting output codes. *Pattern Recognition Letters 32* (2009), 458–467.

[80] Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., and Venkatasubramanian, S. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Sydney, Australia, 2015), Association for Computing Machinery, New York, NY, USA, p. 259–268. doi: 10.1145/2783258.2783311.

[81] Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D., and Amorim Fernández-Delgado, D. Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research 15* (2014), 3133–3181. doi: 10.5555/2627435.2697065.

[82] Ferrari, M., Cremonesi, P., and Jannach, D. Are we really making much progress? a worrying analysis of recent neural recommendation approaches. In *Proceedings of the ACM Conference on Recommender Systems* (Copenhagen, Denmark, 2019), Association for Computing Machinery, New York, NY, USA.

[83] Flaounas, I. Beyond the technical challenges for deploying machine learning solutions in a software company. arXiv:1708.02363, 2017. [Online] Available from: https://arxiv.org/abs/1708.02363.

[84] Fredrikson, M., Jha, S., and Ristenpart, T. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (Denver, CO, USA, 2015), Association for Computing Machinery, New York, NY, USA, p. 1322–1333. doi: 10.1145/2810103.2813677.

[85] Freund, Y., and Schapire, R. E. Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on International Conference on Machine Learning* (Bari, Italy, 1996), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 148–156. doi: 10.5555/3091696.3091715.

[86] Freund, Y., Seung, H. S., Shamir, E., and Tishbyi, N. Selective sampling using the query by committee algorithm. *Machine Learning 28*, 2-3 (1997), 133–168.

[87] Frosst, N., and Hinton, G. Distilling a neural network into a soft decision tree. arXiv:1711.09784, 2017. [Online] Available from: https://arxiv.org/pdf/1711.09784.pdf.

[88] Fu, L. M. Rule Learning by Searching on Adapted Nets. In *Proceedings of the 9th National Conference on Artificial Intelligence* (Anaheim, CA, USA, 1991), MIT Press, Cambridge, MA, USA, pp. 590–595.

[89] Fujii, A., Tokunaga, T., Inui, K., and Tanaka, H. Selective sampling for example based word sense disambiguation. *Computational Linguistics 24*, 4 (1998), 573–597. doi: 10.5555/972764.972766.

[90] Furlanello, T., Lipton, Z., and Anandkumar. Born again neural networks. In *Proceedings of the 35th International Conference on Machine Learning* (2018), vol. 80, Journal of Machine Learning Research, New York, NY, USA, pp. 1607–1616.

[91] Garcia, R., Wandzik, L., Grabner, L., and Krueger, J. The harms of demographic bias in deep face recognition research. In *Proceedings of the 2019 International Conference on Biometrics* (Crete, Greece, 2019), IEEE Computer Society, Washington, DC, USA, pp. 1–6.

[92] Garg, A., Adhikari, N., McDonald, H., Rosas Arellano, M., Devereaux, P., Beyene, J., San, J., and Haynes, R. Effects of computerized clinical decision support systems on practitioner performance and patient outcomes: a systematic review. *JAMA 293*, 10 (2005), 1223–1238.

[93] Goldsteen, A., Ezov, G., Shmelkin, R., Moffie, M., and Farkash, A. arXiv:2008.04113, 2020. [Online] Available at: https://arxiv.org/abs/2008.04113.

[94] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2* (Montreal, Canada, 2014), MIT Press, Cambridge, MA, USA, p. 2672–2680. doi: 10.5555/2969033.2969125.

[95] Goodman, B., and Flaxman, S. European union regulations on algorithmic decision-making and a right to explanation. *AI Magazine 38*, 3 (2017), 50–57. doi: 10.1609/aimag.v38i3.2741.

[96] Goodman, B. W. A step towards accountable algorithms?: Algorithmic discrimination and the european union general data protection. In *Proceedings of the 29th International Conference on*

Neural Information Processing Systems (Barcelona, Spain, 2016), MIT Press, Cambridge, MA, USA.

[97] Google. Deploy tensorflow. [Online] Available from: https://www.tensorflow.org/deploy/.

[98] Gorissen, D., Couckuyt, I., Demeester, P., Dhaene, T., and Crombecq, K. A surrogate modeling and adaptive sampling toolbox for computer based design. *Journal of Machine Learning Research 11* (2010), 2051–2055. doi: 10.5555/1756006.1859919.

[99] Gou, J., Yu, B., Maybank, S. J., and Tao, D. Knowledge distillation: A survey. arXiv:2006.05525, 2020. [Online] Available from: https://arxiv.org/pdf/2006.05525.pdf.

[100] Guha, S., Cheng, B., and Francis, P. Challenges in measuring online advertising systems. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement* (Melbourne, Australia, 2010), Association for Computing Machinery, New York, NY, USA, pp. 81–87. doi: 10.1145/1879141.1879152.

[101] Guidotti, R., Monreale, A., Ruggieri, S., Pedreschi, D., Turini, F., and Giannotti, F. Factual and counterfactual explanations for black box decision making. *IEEE Intelligent Systems 34*, 6 (2019), 14–23. doi = 10.1109/MIS.2019.2957223.

[102] Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., and Pedreschi, D. A survey of methods for explaining black box models. *ACM Computing Surveys 51*, 5 (2018), 1–42. doi: 10.1145/3236009.

[103] Gómez, J. A., Arevalo, J., Paredes, R., and Nin, J. End-to-end neural network architecture for fraud scoring in card payments. *Pattern Recognition Letters 105* (2017), 175–181. doi: 10.1016/j.patrec.2017.08.024.

[104] Hansson, S. Risk. In *Stanford Encyclopedia of Philosophy*, Z. E.N., Ed. Stanford University, Stanford, CA, USA, 2018.

[105] Hardt, M. How big data is unfair. Medium, 2014. [Online] Available from: https://medium.com/@mrtz/how-big-data-is-unfair-9aa544d739de.

[106] Hardt, M., Price, E., and Srebro, N. Equality of opportunity in supervised learning. In *Proceedings of the 29th International Conference on Neural Information Processing Systems* (Barcelona, Spain, 2016), MIT Press, Cambridge, MA, USA, pp. 3323–3331.

[107] Haussler, D. Decision theoretic generalizations of the pac model for neural net and other learning applications. *Information and Computation 100*, 1 (1992), 78–150. doi: 10.1016/0890-5401(92)90010-D.

[108] He, H., Bay, Y., Garcia, E., and Li, S. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* (Hong Kong, China, 2008), IEEE Computer Society, Washington, DC, USA, pp. 1322–1328. doi: 10.1109/IJCNN.2008.4633969.

[109] Heo, B., Lee, M., Yun, S., and Choi, J. Y. Knowledge distillation with adversarial samples supporting decision boundary. In *Proceedings of the 33rd Conference on Artificial Intelligence* (Honolulu, Hawaii, USA, 2019), AAAI Press, Palo Alto, CA, USA, pp. 3771–3778.

[110] Hern, A. Google's solution to accidental algorithmic racism: Ban gorillas. The Guardian, 2018. [Online] Available from: https://www.theguardian.com/technology/2018/jan/12/google-racism-ban-gorilla-black-people.

[111] Hinton, G., and Salakhutdinov, R. Reducing the dimensionality of data with neural networks. *Science 5786*, 313 (2006), 504–507.

[112] Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop* (2015).

[113] Ho, T. B., Kawasaki, S., and Granat, J. *Knowledge Acquisition by Machine Learning and Data Mining*. Springer, Berlin, Heidelberg, 2007, pp. 69–91. doi: 10.1007/978-3-540-71562-7_4.

[114] Hong, S., Xiao, C., Hoang, T. N., Ma, T., Li, H., and Sun, J. Rdpd: Rich data helps poor data via imitation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence* (Macao, China, 2019), International Joint Conferences on Artificial Intelligence, pp. 5895–5901. doi: 10.24963/ijcai.2019/817.

[115] Huang, S., Jin, R., and Zhou, Z. Active learning by querying informative and representative examples. *IEEE Transactions on Pattern Analysis and Machine Intelligence 36*, 10 (2014), 1936–1949. doi: 10.1109/TPAMI.2014.2307881.

[116] Huijser, M. W., and Van Gemert, J. C. Active decision boundary annotation with deep generative models. arXiv:1703.06971, 2017. [Online] Available from: https://arxiv.org/pdf/1703.06971.pdf.

[117] Hundepool, A., Domingo-Ferrer, J., Franconi, L., Giessing, S., Nordholt, E., Spicer, K., and De Wolf, P. *Statistical Disclosure Control*. John Wiley & Sons, Ltd, New Jersey, NJ, USA, 2012. doi: 10.1002/9781118348239.

[118] Idoine, C., Krensky, P., Linden, A., and Brethenoux, E. Magic quadrant for data science and machine learning platforms. Tech. rep., Gartner Research, 2019.

[119] ISO. Iso 31000:2018 risk management - guidelines, 2018. [Online] Available from: https://www.iso.org/iso-31000-risk-management.html.

[120] Jamshidi, P., Velez, M., Kästner, C., and Siegmund, N. Learning to sample: Exploiting similarities across environments to learn performance models for configurable systems. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Lake Buena Vista, FL, USA, 2018), Association for Computing Machinery, New York, NY, USA, p. 71–82. doi: 10.1145/3236024.3236074.

[121] Jordan, T., Ash, R., and Adams, P. On warm-starting neural network training. arXiv:1910.08475, 2019. [Online] Available from: https://arxiv.org/pdf/1910.08475.pdf.

[122] Joshi, A., Porikli, F., and Papanikolopoulos, N. Multi-class active learning for image classification. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition* (Miami, FL, USA, 2009), IEEE Computer Society, Washington, DC, USA, pp. 2372–2379. doi: 10.1109/CVPR.2009.5206627.

[123] Kahneman, D., and Tversky, A. On the reality of cognitive illusions. *Psychological Review 103*, 3 (1996), 582–591. doi: 10.1037/0033-295x.103.3.582.

[124] Kang, M., Mun, J., and Han, B. Towards oracle knowledge distillation with neural architecture search. arXiv:1911.13019, 2019. [Online] Available from: https://arxiv.org/pdf/1911.13019.pdf.

[125] Keane, A., Forrester, A., and Sobester, A. *Engineering Design via Surrogate Modelling: A Practical Guide*. John Wiley & Sons, Ltd, New Jersey, NJ, USA, 2008. doi: 10.1002/9780470770801.

[126] Kilbertus, N., Carulla, M. R., Parascandolo, G., Hardt, M., Janzing, D., and Schölkopf, B. Avoiding discrimination through causal reasoning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, CA, USA, 2017), Curran Associates Inc., Red Hook, NY, USA, pp. 656–666. doi: 10.5555/3294771.3294834.

[127] Kim, J., Kim, S., and Choi, S. Learning to warm-start bayesian hyperparameter optimization. arXiv:1710.06219, 2017. [Online] Available from: https://arxiv.org/pdf/1710.06219.pdf.

[128] Klare, B. F., Burge, M. J., Klontz, J. C., Vorder Bruegge, R. W., and Jain, A. K. Face recognition performance: Role of demographic information. *IEEE Transactions on Information Forensics and Security 7*, 6 (12 2012), 1789–1801. doi: 10.1109/TIFS.2012.2214212.

[129] Kleinberg, J., Ludwig, J., Mullainathan, S., and Sunstein, C. Discrimination in the age of algorithms. *Journal of Legal Analysis 10* (2018), 113–174. doi: 10.1093/jla/laz001.

[130] Kong, E. B., and Dietterich, T. G. Error-correcting output coding corrects bias and variance. In *Proceedings of the 12th International Conference on International Conference on Machine Learning* (Tahoe City, CA, USA, 1995), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 313–321. doi: 10.5555/3091622.3091661.

[131] Kortylewski, A., Egger, B., Schneider, A., Gerig, T., Morel-Forster, A., and Vetter, T. Analyzing and reducing the damage of dataset bias to face recognition with synthetic data. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (Long Beach, CA, USA, 2019), IEEE Computer Society, Washington, DC, USA, pp. 2261–2268. doi: 10.1109/CVPRW.2019.00279.

[132] Kou, Y., Lu, C.-T., Sirwonwattana, S., and Huang, Y.-P. Survey of fraud detection techniques. In *Proceedings of the IEEE International Conference on Networking, Sensing and Control, 2004* (Taipei, Taiwan, 2004), vol. 2, IEEE Computer Society, Washington, DC, USA, pp. 749–754. doi: 10.1109/ICNSC.2004.1297040.

[133] Krasanakis, E., Spyromitros-Xioufis, E., Papadopoulos, S., and Kompatsiaris, Y. Adaptive sensitive reweighting to mitigate bias in fairness-aware classification. In *Proceedings of the 2018 World Wide Web Conference* (2018), International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva,. doi: 10.1145/3178876.3186133.

[134] Kroll, J. The fallacy of inscrutability. *Philosophical Transactions of the Roya Society 376* (2018). doi: 10.1098/rsta.2018.0084.

[135] Kroll, J., Barocas, S., Felten, E., Reidenberg, J., Robinson, D., and Yu, H. Accountable algorithms. *University of Penssylvania Law Review 165*, 165 (2016), 633–705.

[136] Kuttichira, D., Gupta, S., Li, C., Rana, S., and Venkatesh, S. Explaining black-box models using interpretable surrogates. In *Trends in Artificial Intelligence. PRICAI 2019. Lecture Notes in Computer Science* (2019), vol. 11670, Springer, Berlin, Heidelberg, pp. 3–15.

[137] Lakkaraju, H., Bach, S. H., and Leskovec, J. Interpretable decision sets. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, CA, USA, 2016), Association for Computing Machinery, New York, NY, USA, p. 1675–1684. doi: 10.1145/2939672.2939874.

[138] Lang, K., and Baum, E. Query learning can work poorly when a human oracle is used. In *Proceedings of the 9th International Conference on Machine Learning* (Aberdeen, Scotland, 1992), IEEE Computer Society, Washington, DC, USA, pp. 335–340.

[139] Laviolette, F., Marchand, M., and Shanian, S. Selective sampling for classification. In *Proceedings of the 2008 Canadian Conference on Artificial Intelligence* (Ontario, Canada, 2008), vol. 5032, Springer, Berlin, Heidelberg, pp. 191–202.

[140] Lee, H., Hwang, S., and Shin, J. Rethinking data augmentation: Self-supervision and self-distillation. arXiv:1910.05872, 2019. [Online] Available from: https://arxiv.org/pdf/1910.05872.pdf.

[141] Lewis, D. D., and Gale, W. A. A sequential algorithm for training text classifiers. In *PProceedings of the 17th Annual International ACM SIGIR Conference on Research and Development*

in Information Retrieval (Dublin, Ireland, 1994), Springer, Berlin, Heidelberg, pp. 3–12. doi: 0.5555/188490.188495.

[142] Li, D., Yang, Y., Song, Y., and Hospedales, T. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 5542–5550.

[143] Li, M., Zuo, T., Li, R., White, M., and Zheng, W. Accelerating large scale knowledge distillation via dynamic importance sampling. arXiv:1812.00914, 2018. [Online] Available from: https://arxiv.org/pdf/1812.00914.pdf.

[144] Li, Y., Yang, J., Song, Y., Cao, L., Luo, J., and Li, L. Learning from noisy labels with distillation. In *2017 IEEE International Conference on Computer Vision (ICCV)* (Venice, Italy, 2017), IEEE Computer Society, Washington, DC, USA, pp. 1928–1936. doi: 10.1109/ICCV.2017.211.

[145] Lindenbaum, M., Markovitch, S., and Rusakov, D. Selective sampling for nearest neighbor classifiers. *Machine Learning 54*, 2 (2004), 125–152. doi: 10.1023/B:MACH.0000011805.60520.fe.

[146] Ling, C. X., and Du, J. Active learning with direct query construction. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Las Vegas, NV, USA, 2008), Association for Computing Machinery, New York, NY, USA, p. 480–487. doi: 10.1145/1401890.1401950.

[147] Lipton, Z. The mythos of model interpretability. In *Workshop on Human Interpretation in Machine Learning* (New York, NY, USA, 2016).

[148] Lisboa, P., Ifeachor, E., and Szczepaniak, P. *Artificial Neural Networks in Biomedicine*. Springer Science & Business Media, Berlin, Germany, 2000. doi: 10.1007/978-1-4471-0487-2.

[149] Liu, R., Fusi, N., and Mackey, L. Teacher-student compression with generative adversarial networks. arXiv:1812.02271, 2018. [Online] Available from: https://arxiv.org/pdf/1812.02271.pdf.

[150] Liu, Y., Chen, X., Liu, C., and Song, D. Delving into transferable adversarial examples and black-box attacks. In *Proceedings of the 5th International Conference on Learning Representations* (Toulon, France, 2017).

[151] Livingston, J., Norris, J., and Oppenhuis, J. Auditing artificial intelligence. ISACA, 2018. [Online] Available from: http://www.isaca.org/Knowledge-Center/Research/ResearchDeliverables/Pages/Auditing-Artificial-Intelligence.aspx.

[152] Loeffel, P. Adaptive machine learning algorithms for data streams subject to concept drifts. Machine Learning [cs.LG]. Université Pierre et Marie Curie - Paris VI, 2017. English. ffNNT:2017PA066496ff. fftel-01812044v2f, 2017.

[153] Lopez-Paz, D., Bottou, L., Scholkopf, B., and Vapnik, V. Unifying distillation and privileged information. In *Proceedings of the 4th International Conference on Learning Representations* (San Juan, Puerto Rico, 2016).

[154] Lou, Y., Caruana, R., and Gehrke, J. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Beijing, China, 2012), Association for Computing Machinery, New York, NY, USA, p. 150–158. doi: 10.1145/2339530.2339556.

[155] Lowd, D., and Meek, C. Adversarial learning. In *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining* (Chicago, IL, USA, 2005), Association for Computing Machinery, New York, NY, USA, p. 641–647. doi: 10.1145/1081870.1081950.

[156] of Artificial Intelligence (BAAI), B. A. Beijing ai principles. Tech. rep., [Online] Available from: https://www.baai.ac.cn/blog/beijing-ai-principles, 2019.

[157] of the President, E. O. The national artificial intelligence research and development strategic plan. Tech. rep., National Science and Technology Council, 2016. [Online] Available from: https://www.nitrd.gov/PUBS/national_ai_rd_strategic_plan.pdf.

[158] of the President, E. O. Preparing for the future of artificial intelligence. Tech. rep., National Science and Technology Council, 2016. [Online] Available from https://www.whitehouse.gov/sites/default/files/whitehouse_files/microsites/ostp/NSTC/preparing_for_the_future_of_ai.pdf.

[159] on AI, H.-L. E. G. Ethics guidelines for trustworthy ai. Tech. rep., European Commission, 2019. [Online] Available from: https://ec.europa.eu/digital-single-market/en/news/ethics-guidelines-trustworthy-ai.

[160] on Human Rights, G. F. C. How to prevent discriminatory outcomes in machine learning. Tech. rep., World Economic Forum, 2016. [Online] Available from: https://www.weforum.org/whitepapers/how-to-prevent-discriminatory-outcomes-in-machine-learning.

[161] Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., and Zhang, G. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering 31*, 12 (2019), 2346–2363. doi: 10.1109/TKDE.2018.2876857.

[162] Luca, M., Kleinberg, J., and Mullainathan, S. Algorithms need managers, too. Harvard Business Review, 2016. [Online] Available from: https://hbr.org/2016/01/algorithms-need-managers-too.

[163] M., K., and L., S. Uncertainty wrappers for data-driven models. In *Computer Safety, Reliability, and Security. SAFECOMP 2019. Lecture Notes in Computer Science* (2019), vol. 11699, Springer, Berlin, Heidelberg. doi: 10.1007/978-3-030-26250-1_29.

[164] Maestre, R., Rodríguez, J. A., Nin, J., Brando, A., Unceta, I., Hernández, A., Carmona, J., Caggioni, L., and Hosein, S. Delivering advanced artificial intelligence in the banking industry. Tech. rep., BBVA Data & Analytics and Google Cloud, October 2018. [Online] Available from: https://www.bbvadata.com/white_papers/advanced_ai.pdf.

[165] Maurer, M., Gerdes, J., Lenz, B., and Winner, H. *Autonomous Driving: Technical, Legal and Social Aspects*. Springer, Berlin, Heidelberg, 2016. doi: 10.1007/978-3-662-48847-8.

[166] McAllester, D. A. Pac-bayesian model averaging. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory* (Santa Cruz, California, USA, 1999), Association for Computing Machinery, New York, NY, USA, p. 164–170. doi: 10.1145/307400.307435.

[167] McCallum, A., and Nigam, K. Employing em and pool-based active learning for text classification. In *Proceedings of the 15th International Conference on Machine Learning* (Madison, WI, USA, 1998), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 350–358. doi: 10.5555/645527.757765.

[168] Mei, S., and Zhu, X. Using machine teaching to identify optimal training-set attacks on machine learners. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence* (Austin, TX, USA, 2015), AAAI Press, Palo Alto, CA, USA, p. 2871–2877. doi: 0.5555/2886521.2886721.

[169] Melis, M., Demontis, A., Biggio, B., Brown, G., Fumera, G., and Roli, F. Is deep learning safe for robot vision? adversarial examples against the icub humanoid. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)* (Venice, Italy, 2017), IEEE Computer Society, Washington, DC, USA, pp. 751–759.

[170] Mena, J., Brando, A., Pujol, O., and Vitrià, J. Uncertainty estimation for black-box classification models: a use case for sentiment analysis. In *Pattern Recognition and Image Analysis. IbPRIA 2019. Lecture Notes in Computer Science* (Madrid, Spain, 2019), vol. 11867, pp. 29–40. doi: 10.1007/978-3-030-31332-6_3.

[171] Mena, J., Pujol, O., and Vitrià, J. Uncertainty-based rejection wrappers for black-box classifiers. *IEEE Access 8* (2020), 101721–101746. doi: 10.1109/ACCESS.2020.2996495.

[172] Mendelson, S. *A Few Notes on Statistical Learning Theory*. Springer, Berlin, Heidelberg, 2003, p. 1–40. doi: 10.5555/863714.863716.

[173] Menon, A. K., Rawat, A. S., Reddi, S. J., Kim, S., and Kumar, S. Why distillation helps: A statistical perspective. arXiv:2005.10419, 2020. [Online] Available from: https://arxiv.org/pdf/2005.10419.pdf.

[174] Miller, C. When algorithms discriminate. The New York Times, 2015. [Online] Available from: https://www.nytimes.com/2015/07/10/upshot/when-algorithms-discriminate.html.

[175] Mobahi, H., Farajtabar, M., and Bartlett, P. Self-distillation amplifies regularization in hilbert space. arXiv:2002.05715, 2020. [Online] Available from: https://arxiv.org/pdf/2002.05715.pdf.

[176] Mukherjee, K. Application of the gabriel graph to instance based learning. Master's thesis, Simon Fraser University, 2004.

[177] Müller, R., Kornblith, S., and Hinton, G. When does label smoothing help? In *Proceedings of the 33rd Conference on Neural Information Processing Systems* (Vancouver, Canada, 2019), MIT Press, Cambridge, MA, USA.

[178] Narodytska, N., and Kasiviswanathan, S. Simple black-box adversarial attacks on deep neural networks. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (Honolulu, HI, USA, 2017), IEEE Computer Society, Washington, DC, USA, pp. 1310–1318. doi: 10.1109/CVPRW.2017.172.

[179] Nayak, G. K., Mopuri, K. R., Shaj, V., Babu, R. V., and Chakraborty, A. Zero-shot knowledge distillation in deep networks. arXiv:1905.08114, 2019. [Online] Available from: https://arxiv.org/abs/1905.08114.

[180] News, X. China rolls out three-year program for ai growth. [Online] Available from: http://www.china.org.cn/business/2016-05/24/content_38521175.htm, 2016.

[181] Neyshabur, B., Tomioka, R., Salakhutdinov, R., and Srebro, N. Geometry of optimization and implicit regularization in deep. arXiv:1705.03071, 2017. [Online] Available from: https://arxiv.org/pdf/1705.03071.pdf.

[182] Pan, S., and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering 22*, 10 (2010), 1345–1359.

[183] Papernot, N., McDaniel, P., and Goodfellow, I. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv:1605.07277, 2016. [Online] Available from: https://arxiv.org/pdf/1605.07277.pdf.

[184] Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Berkay Celik, Z., and Swami, A. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security* (Abu Dhabi, United Arab Emirates, 2017), Association for Computing Machinery, New York, NY, USA, pp. 506–519. doi: 10.1145/3052973.3053009.

[185] Papernot, N., McDaniel, P., Wu, X., Jha, S., and Swami, A. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE Symposium on Security and Privacy* (San Jose, CA, USA, 2016), pp. 582–597. doi: 10.1109/SP.2016.41.

[186] Parliament, E. Civil law rules on robotics - european parliament resolution of 16 february 2017 with recommendations to the commission on civil law rules on robotics (2015/2103(inl). No.: P8TA-PROV(2017)00 51 [Online] Available from: http://www.europarl.europa.eu/doceo/document/TA-8-2017-0051_EN.html, 2017.

[187] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. In *Proceedings of the 32th International Conference on Neural Information Processing Systems* (Barcelona, Spain, 2017), Curran Associates, Inc., Red Hook, NY, USA, p. 8024–8035.

[188] Phuong, M., and Lampert, C. Towards understanding knowledge distillation. In *Proceedings of the 36th International Conference on Machine Learning* (Long Beach, CA, USA, 2019), vol. 97, Journal of Machine Learning Research, New York, NY, USA, pp. 5142–5151.

[189] Podesta, J, ., Pritzker, P., Moniz, E., Holdren, J., and Zients, J. Big data: Seizing opportunities, preserving values. Tech. rep., Executive Office of the President. The White House, 2014. [Online] Available from: https://obamawhitehouse.archives.gov/sites/default/files/docs/big_data_privacy_report_may_1_2014.pdf.

[190] Polino, A., Pascanu, R., and Alistarh, D. Model compression via distillation and quantization. In *Proceedings of the 6th International Conference on Learning Representations* (Vancouver, Canada, 2018).

[191] Popejoy, A. B., and Fullerton, S. M. Genomics is failing on diversity. *Nature 538* (2016), 161–164. doi: 10.1038/538161a.

[192] Qi, J., Linke, G., Zhanpeng, J., and Yuguang, F. Preserving model privacy for machine learning in distributed systems. *IEEE Transactions on Parallel and Distributed Systems 8*, 19 (2018), 1808–1822. doi: 10.1109/TPDS.2018.2809624.

[193] Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., and Kevin Tucker, P. Surrogate-based analysis and optimization. *Progress in Aerospace Sciences 41*, 1 (2005), 1–28. doi: 10.1016/j.paerosci.2005.02.001.

[194] Radosavovic, I., Dollár, P., Grishick, R. B., Gkioxari, G., and He, K. Data distillation: Towards omni-supervised learning. In *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), IEEE Computer Society, Washington, DC, USA, pp. 4119–4128. doi: 10.1109/CVPR.2018.00433.

[195] Raji, I., and Buolamwini, J. Actionable auditing: Investigating the impact of publicly naming biased performance results of commercial ai products. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society* (Honolulu, HI, USA, 2019), Association for Computing Machinery, New York, NY, USA, pp. 429–435.

[196] Research, B. Mexico real estate outlook. first half 2018. [Online] Available from: https://www.bbvaresearch.com/en/publicaciones/mexico-real-estate-outlook-first-half-2018/, 2018.

[197] Ribeiro, M. T., Singh, S., and Guestrin, C. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, CA, USA, 2016), Association for Computing Machinery, New York, NY, USA, pp. 1135–1144.

[198] Ribeiro, M. T., Singh, S., and Guestrin, C. Anchors: High-precision model-agnostic explanations. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18)* (New Orleans, LA, USA, 2018), AAAI Press, Palo Alto, CA, USA, pp. 1527–1535.

[199] Riccardi, G., and Hakkani-Tur, D. Active learning: theory and applications to automatic speech recognition. *IEEE Transactions on Speech and Audio Processing 13*, 4 (2005), 504–511. doi: 10.1109/TSA.2005.848882.

[200] Romero, A., Ballas, N., Kahou, S., Chassang, A., Gatta, C., and Bengio, Y. Fitnets: Hints for thin deep nets. In *Proceedings of the 3th International Conference on Learning Representations* (San Diego, CA, USA, 2015).

[201] Roth, L. Looking at shirley, the ultimate norm: Colour balance, image technologies and cognitive equity. *Canadian Journal of Communication 34*, 1 (2009), 111–136. doi: 10.22230/cjc.2009v34n1a2196.

[202] Rudin, C. Please stop explaining black box models for high-stakes decisions. In *Workshop on Critiquing and Correcting Trends in Machine Learning* (Montreal, Canada, 2018).

[203] Ruffy, F., and Chahal, K. The state of knowledge distillation for classification tasks. arXiv:1912.10850, 2019. [Online] Available from: https://pdfs.semanticscholar.org/dda8/66de319320a172a9dc7d790eeb32586346c8.pdf.

[204] Rüping, S. *Learning Interpretable Models*. PhD thesis, University of Dortmund, Dortmund, Germany, 2006.

[205] Samek, W., Montavon, G., Lapuschkin, S., Anders, C., and Müller, K. Toward interpretable machine learning: Transparent deep neural networks and beyond. arXiv:2003.07631, 2020. [Online] Available from: https://arxiv.org/pdf/2003.07631.pdf.

[206] Samek, W., and Müller, K. *Towards explainable artificial intelligence*. Springer, Berlin, Heidelberg, 2019, pp. 5–22.

[207] Sandvig, C., Hamilton, K., Karahalios, K., and Langbort, C. Auditing algorithms: Research methods for detecting discrimination on internet platforms. Data and Discrimination: Converting Critical Concerns into Productive Inquiry, 2014. [Online] Available from: http://www-personal.umich.edu/~csandvig/research/Auditing%20Algorithms%20--%20Sandvig%20--%20ICA%202014%20Data%20and%20Discrimination%20Preconference.pdf.

[208] Sattigeri, P., Hoffman, S., Chenthamarakshan, V., and Varshney, K. Fairness gan: Generating datasets with fairness properties using a generative adversarial network. *IBM Journal of Research and Development 63*, 4/5 (2019), 3:1–3:9. doi: 10.1147/JRD.2019.2945519.

[209] Schapire, R. E., Freund, Y., Barlett, P., and Lee, W. S. Boosting the margin: A new explanation for the effectiveness of voting methods. In *Proceedings of the 14th International Conference on Machine Learning* (Nashville, TN, USA, 1997), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, p. 322–330.

[210] Scheffer, T., Decomain, C., and Wrobel, S. Active hidden markov models for information extraction. In *Proceedings of the 4th International Conference on Advances in Intelligent Data Analysis* (Cascais, Portugal, 2001), Springer, Berlin, Heidelberg, pp. 309–318. doi: 10.5555/647967.741626.

[211] Schumann, R., and Rehbein, I. Active learning via membership query synthesis for semi-supervised sentence classification. In *Proceedings of the 23rd Conference on Computational Natural Language Learning* (Hong Kong, China, 2019), Association for Computational Linguistics, Stroudsburg, PN, USA, p. 472–481. doi: 10.18653/v1/K19-1044.

[212] Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., and Young, M. Machine learning: The high interest credit card of technical debt. In *SE4ML: Software Engineering for Machine Learning (NIPS 2014 Workshop)* (Montreal, Canada, 2014).

[213] Selbst, A. D., and Powles, J. Meaningful Information and the Right to Explanation. *International Data Privacy Law 7*, 4 (2017), 233–242. doi: 10.1093/idpl/ipx022.

[214] Settles, B. Active learning literature survey. Computer Science Technical Report 1648, University of Wisconsin–Madison, January 2009. [Online] Available from: http://burrsettles.com/pub/settles.activelearning.pdf.

[215] Shokri, R., Tech, C., Stronati, M., and Shmatikov, V. Membership Inference Attacks Against Machine Learning Models. In *2017 IEEE Symposium on Security and Privacy (SP)* (San Jose, CA, USA, 2017), IEEE Computer Society, Washington, DC, USA, pp. 3–18. doi: 10.1109/SP.2017.41.

[216] Simpson, T., Poplinski, J., Koch, P. N., and Allen, J. Metamodels for computer-based engineering design: Survey and recommendations. *Engineering with Computers 17*, 2 (2001), 129–150. doi: 10.1007/PL00007198.

[217] Simpson, T., Toropov, V., Balabanov, V., and Viana, F. Design and analysis of computer experiments in multidisciplinary design optimization: A review of how far we have come - or not. In *Proceedings of the 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference* (2008), American Institute of Aeronautics and Astronautics, Reston, VA, USA. doi: 10.2514/6.2008-5802.

[218] Song, C., Ristenpart, T., and Shmatikov, V. Machine Learning Models that Remember Too Much. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (Dallas, TX, USA, 2017), Association for Computing Machinery, New York, NY, USA, pp. 587–601. doi: 10.1145/3133956.3134077.

[219] Song, X., Fan, G., and Rao, M. Svm-based data editing for enhanced one-class classification of remotely sensed imagery. *IEEE Geoscience and Remote Sensing Letters 5*, 2 (2008), 189–193. doi: 10.1109/LGRS.2008.916832.

[220] Spector, A., Norvig, P., and Petrov, S. Google's hybrid approach to research. *Communications of the ACM 55*, 7 (2012), 34–37. doi: 10.1145/2209249.2209262.

[221] Srivastava, A., Kundu, A., Sural, S., and Majumdar, A. Credit card fraud detection using hidden markov model. *IEEE Transactions on Dependable and Secure Computing 5*, 1 (2008), 37–48. doi: 10.1109/TDSC.2007.70228.

[222] Super heroes dataset, kaggle. [Online] Available from: https://www.kaggle.com/claudiodavi/superhero-set.

[223] Superhero database. [Online] Available from: https://www.superherodb.com/.

[224] Sweeney, L. Discrimination in online ad delivery. *ACM Queue 11*, 3 (2013), 10–29. doi: 10.1145/2460276.2460278.

[225] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition* (Las Vegas, NV, USA, 2016), IEEE Computer Society, Washington, DC, USA, pp. 2818–2826. doi: 10.1109/CVPR.2016.308.

[226] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *Proceedings of the 2nd International Conference on Learning Representations* (Banff, Canada, 2014).

[227] Tang, J., Shivanna, R., Zhao, Z., Lin, D., Singh, A., Chi, E. H., and Jain, S. Understanding and improving knowledge distillation. arXiv:2002.03532, 2020. [Online] Available from: https://arxiv.org/pdf/2002.03532.pdf.

[228] Thrun, S. Extracting rules from artificial neural networks with distributed representations. In *Proceedings of the 7th International Conference on Neural Information Processing Systems* (Denver, CO, USA, 1994), MIT Press, Cambridge, MA, USA, p. 505–512. doi: 10.5555/2998687.2998750.

[229] Tong, S., and Koller, D. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research 2* (2002), 999–1006. doi: 10.1162/153244302760185243.

[230] Torra, V. *Data Privacy: Foundations, New Developments and the Big Data Challenge*. Springer, Berlin, Heidelberg, 2017. doi: 10.1007/978-3-319-57358-8.

[231] Torrey, L., and Shavlik, J. Transfer learning. In *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, E. Soria Olivas, J. Martín Guerrero, M. Martinez-Sober, J. Magdalena-Benedito, and A. Serrano López, Eds. IGI Global, PA, USA, 2010, pp. 242–264. doi: 10.4018/978-1-60566-766-9.ch011.

[232] Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., and Ristenpart, T. Stealing machine learning models via prediction apis. In *Proceedings of the 25th USENIX Conference on Security Symposium* (Austin, TX, USA, 2016), USENIX Association, Berkeley, CA, USA, pp. 601–618. doi: 10.5555/3241094.3241142.

[233] Unceta, I., Nin, J., and Pujol, O. Towards global explanations for credit risk scoring. In *Workshop on Challenges and Opportunities for AI in Financial Services: the Impact of Fairness, Explainability, Accuracy, and Privacy (FEAP-AI4Fin)* (Montreal, Canada, 2018).

[234] Unceta, I., Nin, J., and Pujol, O. From batch to online learning using copies. In *Artificial Intelligence Research and Development. Frontiers in Artificial Intelligence and Applications* (2019), vol. 319, IOS press, Amsterdam, The Netherlands, pp. 125–134. doi: 10.3233/FAIA190115 [Online] Available from: http://ebooks.iospress.nl/volumearticle/52828.

[235] Unceta, I., Nin, J., and Pujol, O. Using copies to remove sensitive data: A case study on fair superhero alignment prediction. In *Pattern Recognition and Image Analysis. IbPRIA 2019. Lecture Notes in Computer Science* (Madrid, Spain, 2019), vol. 11867, Springer, Berlin, Heidelberg, pp. 182–193. doi: 10.1007/978-3-030-31332-6_16.

[236] Unceta, I., Nin, J., and Pujol, O. Copying machine learning classifiers. *IEEE Access 8*, 11 (2020), 160268–160284. doi = 10.1109/ACCESS.2020.3020638.

[237] Unceta, I., Nin, J., and Pujol, O. Environmental adaptation and differential replication in machine learning. *Entropy 22*, 10 (2020). doi = 10.3390/e22101122.

[238] Unceta, I., Nin, J., and Pujol, O. Risk mitigation in algorithmic accountability: The role of machine learning copies. *PlosONE* (2020). doi = 10.1371/journal.pone.0241286.

[239] Unceta, I., Nin, J., and Pujol, O. Transactional compatible representations for high value client identification: A financial case study. In *Complex Networks XI: Proceedings of the 11th Conference on Complex Networks CompleNet* (Exeter, UK, 2020), Springer, Berlin, Heidelberg, pp. 334–345.

[240] Unceta, I., Palacios, D., Nin, J., and Pujol, O. Sampling unknown decision functions to build classifier copies. In *Proceedings of 17th International Conference on Modeling Decisions for Artificial Intelligence* (Sant Cugat, Spain, 2020), pp. 192–204. doi: 10.1007/978-3-030-57524-3_16.

[241] (FICO), F. I. C. Introduction to scorecard for FICO model builder. [Online] Available from: https://www.fico.com/en/latest-thinking/white-paper/introduction-model-builder-scorecard, 2011.

[242] Valiant, G. A theory of the learnable. *Communications of the ACM 27*, 11 (1984), 1134–1142. doi: 10.1145/1968.1972.

[243] Van Looveren, A., and Klaise, J. Interpretable counterfactual explanations guided by prototypes. arXiv:1907.02584, 2019. https://arxiv.org/pdf/1907.02584.

[244] Vapnik, V., and Izmailov, R. Learning using privileged information: Similarity control and knowledge transfer. *Journal of Machine Learning Research 16*, 61 (2015), 2023–2049. doi: 10.5555/2789272.2886814.

[245] Vapnik, V. N. Principles of risk minimization for learning theory. In *Proceedings of the 4th International Conference on Neural Information Processing Systems* (Denver, Colorado, 1991), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 831–838. doi: 10.5555/2986916.2987018.

[246] Vapnik, V. N. *The Nature of Statistical Learning Theory*. Springer, Berlin, Heidelberg, 2000. doi: 10.1007/978-1-4757-3264-1_1.

[247] Vapnik, V. N., and Chervonenkis, A. *Theory of Pattern Recognition (in Russian)*. Nauka, Moscow, Russia, 1974.

[248] Veale, M., and Binns, R. Fairer machine learning in the real world: Mitigating discrimination without collecting sensitive data. *Bid Data & Society 4*, 2 (2017), 1–17. doi: 10.1177/2053951717743530.

[249] Vedder, A., and Naudts, L. Accountability for the use of algorithms in a big data environment. *International Review of Law, Computers & Technology 31*, 4 (2017), 1–19. doi: 10.1080/13600869.2017.1298547.

[250] Volchan, S. B. Probability as typycality. arXiv:physics/0611172, 2007. [Online] Available from: https://arxiv.org/pdf/physics/0611172.pdf.

[251] Wachter, S., Mittelstadt, B., and Floridi, L. Why a right to explanation of automated decision-making does not exist in the general data protection regulation. *International Data Privacy Law 7*, 2 (2017), 76–99. doi: 10.1093/idpl/ipx005.

[252] Wachter, S., Mittelstadt, B., and Russell, C. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. arXiv:1711.00399. [Online] Available from: https://arxiv.org/pdf/1711.00399.

[253] Wang, D., Li, Y., Wang, L., and Gong, B. Neural networks are more productive teachers than human raters:active mixup for data-efficient knowledge distillation from a blackbox model. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (Seattle, WV, USA, 2020), IEEE Computer Society, Washington, DC, USA, pp. 1498–1507.

[254] Wang, J., Bao, W., Sun, L., Zhu, X., Cao, B., and Yu, P. S. Private model compression via knowledge distillation. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence* (Honolulu, HI, USA, 2019), AAAI Press, Palo Alto, CA, USA, pp. 1190–1198.

[255] Weiss, K., Khoshgoftaar, T., and Wang, D. A survey of transfer learning. *Journal of Big Data 3*, 9 (2016). doi: 10.1186/s40537-016-0043-6.

[256] Wexler, J., Pushkarna, M., Bolukbasi, T., Wattenberg, M., Viégas, F., and Wilson, J. The what-if tool: Interactive probing of machine learning models. *IEEE Transactions on Visualization and Computer Graphics 26*, 1 (2020), 56–65.

[257] Wolpert, D. H. Original contribution: Stacked generalization. *Neural Networks 5*, 2 (1992), 241–259. doi: 10.1016/S0893-6080(05)80023-1.

[258] Yang, C., Xie, L., Qiao, S., and Yuille, A. L. Knowledge distillation in generations: More tolerant teachers educate better students. arXiv:1805.05551, 2018. [Online] Available from: https://arxiv.org/pdf/1805.05551.pdf.

[259] Yang, Y., Ma, Z., Nie, F., Chang, X., and Hauptmann, A. Learning by uncertainty sampling with diversity maximization. *International Journal of Computer Vision 113* (2015), 113–127. doi: 10.1007/s11263-014-0781-x.

[260] Yau, W. C. How zendesk serves tensorflow models in production. Medium, 2017.

[261] Yuan, L., Tay, F. E., Li, G., Wang, T., and Feng, J. Revisiting knowledge distillation via label smoothing regularization. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (2020), IEEE Computer Society, Washington, DC, USA.

[262] Zeng, X., and Martinez, T. Using a neural network to approximate an ensemble of classifiers. *Neural Processing Letters 12*, 3 (2000), 225–237. doi: 10.1023/A:1026530200837.

[263] Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires re-thinking generalization. In *Proceedings of the 5th International Conference on Learning Representations* (Toulon, France, 2017).

[264] Zhang, Y., Xiang, T., Hospedales, T., and Lu, H. Deep mutual learning. In *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (Salt Lake City, UT, USA, 2018), IEEE Computer Society, Washington, DC, USA, pp. 4320–4328. doi: 10.1109/CVPR.2018.00454.

[265] Zheng, A., and Raman, S. The challenges of bringing machine learning to the masses. In *Workshop on Software Engineering for Machine Learning* (Montreal, Canada, 2014).

[266] Zhu, J.-J., and Bento, J. Generative adversarial active learning. arXiv:1702.07956, 2017. [Online] Available from: https://arxiv.org/pdf/1702.07956.pdf.

[267] Zhu, X. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence* (Austin, TX, USA, 2015), AAAI Press, Palo Alto, CA, USA, p. 4083–4087. doi: 10.5555/2888116.2888288.