Selection Bias

Identification and Mitigation With No Ground Truth Information

Katharina Dost

A thesis submitted in fulfilment of the requirements for the degree of
Doctor of Philosophy in Computer Science,
The University of Auckland, 2022.

Abstract

Machine Learning *should* be able to support decision-making by focusing on purely logical conclusions based on historical data. If this data is biased, however, that bias will be transferred to the model and remains undetected as the performance is validated on a test set drawn from the same biased distribution. Existing strategies for bias identification and mitigation generally rely on some sort of knowledge of the bias or the ground truth. This reliance is problematic, particularly if the user is not aware of the bias, no ground truth knowledge is available, or no concrete target task is defined yet, e.g., during data gathering.

We argue that some indication of future problems is present in the historical dataset itself. Extracting it as early as during data gathering can help correct the flaws on-the-fly or create awareness in researchers working with the dataset.

In this thesis, we aim to identify selection biases on the historical data alone when no ground-truth information is available. Selection biases stem from a non-uniform sampling process. To mitigate them, we generate additional data points that bridge the gap between sample and ground-truth distribution. Pioneering this research topic, we suggest three algorithms built on the assumption that the distribution of sufficiently large and unbiased datasets should be smooth, without any sudden drops in density.

Extensive experiments and discussions highlight the need for such data analysis tools and illustrate that each of our methods has its own merits. Overall, we contribute to a better understanding of the data we use and trust and challenge existing procedures in machine learning that accept flawed data as given and treat symptoms rather than causes.

Acknowledgements

First and foremost, I would like to thank my advisors Dr. Jörg Wicker and Dr. Patricia Riddle. You found a myriad of ways to let me experience much more than “just” a PhD and helped me grow not only as a researcher but also as a person. I could not have found any better guides, cheerleaders, or role models. My gratitude also goes to the School of Computer Science at the University of Auckland for granting me the opportunity to pursue this degree and to my research group members and peers for their warm friendship, collaboration, and swarm intelligence, particularly Luke Chang, Jonathan Kim, Zac Pullar-Strecker, Johnny Zhu, Liam Brydon, Dr. Ioannis Ziogas, Olivier Graffeuille, and Mitchell Rogers.

On a more personal note, I would like to thank my three parents for, apart from the genes I received, their constant support and love during those challenging times: My father Thomas for enabling this PhD and for reminding me countless times of the infinite number of gray nuances in between black and white, my mother Ursula for her education paradigm that everything is debatable if only I could bring a strong argument, and my father Matthias for his unbreakable belief in me and the feeling that there is always a safe harbor I can sail home to. I am also deeply grateful for my partner Behzad, my camel. You would have carried me all the way through the desert and still offered me your last sip of water. Special thanks to my friends Steffi, Frauke, and Sabine for reserving my place in their lives despite the distance and my long absence. I could never replace you.

Last but not least, I would like to thank the examiners of this thesis for sacrificing their time to provide valuable feedback and guidance.

Contents

1 Introduction

Throughout the years, machine learning and data mining have gained influence in various applications. To overcome the limitations of our own knowledge and experience, these disciplines learn concepts and patterns from historical data and thereby discover latent knowledge. In contrast to a human decision-maker, machine learning *should* be able to overcome conscious and unconscious human emotions, prejudices, and biases and discover patterns that are well supported by a large body of evidence. As such, it has been applied to domains with large amounts of data that are no longer humanly processible and require us to rely, to a certain degree, on the models trained in automated settings, e.g., credit scoring [63], medical diagnoses [107], or crime risk assessment [56].

After passing thorough tests on historical data (*training data*), the trained concepts are transferred to fresh, previously unseen data (*target data*) under the expectation that they hold equally well and provide us with new insights. Chiang and Yin [29] have found in a study that people, when presented with new data similar to the historical data they (and the model) experienced during training, tend to trust their own intuition rather than the model. However, when presented with dissimilar new data, people tend to rely on the model predictions when making decisions.

This reliance is risky since, by default, machine learning follows the *i.i.d. (independent and identically distributed)* paradigm: If a training dataset is distributed as the target data and independent of it, a model can train on the historical data and be expected to generalize well. However, if the i.i.d. assumption is violated, there is no guarantee that the predictions output by the model will be even remotely correct.

We can observe a similar effect in humans. For example, one might be familiar with housing prices in one's hometown and could estimate a property's value given simple statistics there, but could only make an educated guess for houses in other parts of the world [29]. Although it should therefore be intuitive that a model cannot be expected to generalize to unseen domains, in machine learning, these effects are largely overlooked since the model testing is carried out on the historical data only [117].

While overlooked performance drops of machine learning models due to distributional changes cause inconveniences or monetary setbacks in some applications, they have severe ethical implications in others. In recent years, researchers found multiple widely established machine learning algorithms to behave unfairly due to training data issues [97]. One of the most prominent cases is the *COMPAS (Correctional Offender Management Profiling for Alternative Sanctions)* algorithm that predicts the likelihood of recidivism of defendants in U.S. courts. Here, an imbalance in data availability caused the model to develop racist behavior against black defendants [4, 40]. Another prominent case is Amazon's hiring algorithm that, trained on historical data in the male-dominated tech field, learned to discriminate against women for technical jobs [34].

The examples above highlight that flaws in the training data cannot be considered to be isolated problems – they can cause ripple effects reaching substantially further than a suboptimal performance of a single model. In fact, when used for decision-making, the trained models influence and shape the data to be gathered in the future. For example, if the hiring algorithm continued to choose men for technical jobs, the gender imbalance in the dataset would deteriorate, and the algorithm, when retrained, would discriminate even worse. Additionally, more subtle effects, such as discouragement of women due to the rejection and simultaneous confirmation of men, can influence, even if only slightly, gender roles on a global scale, which can then impact further parts of our lives. To avoid these unexpected effects, it

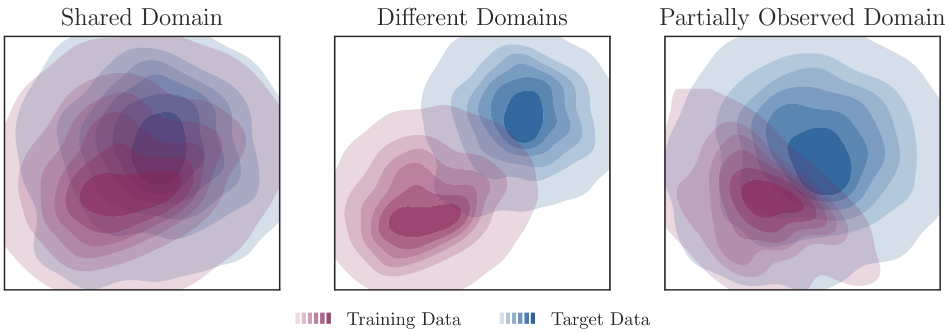


Figure 1.1: Distinction of different types of biases between training and target data. Darker hues indicate higher densities.

is crucial to understand the historical datasets together with their flaws in order to create responsible and reliable models.

Whenever the training distribution does not match the target distribution, we speak of a *bias*. A bias is always relative to a reference distribution. There is a plethora of different types of and reasons for biases in datasets [97]. We distinguish three major issues that arise frequently (see Figure 1.1 for a visualization):

1. Although training and target data might stem from the same domain, their distributions might differ, and the model will focus on the wrong parts of the domain (Figure 1.1 left). An example of this would be political polls where the availability of participants dictates the distribution [62].

2. The model might be applied to at least a partially different domain (Figure 1.1 center), as discussed in the property price scenario. In this case, the model predictions might be entirely incorrect. Another example would be clinical trials [107] where the data is collected from local volunteers that might only represent part of the population. However, the resulting model will be used to predict reactions to treatments or drugs for all future patients.

3. A special case of the previous one arises if the observed domain is a subset of the target domain (Figure 1.1 right). This type of bias occurs frequently, for example, if the volunteers in clinical studies are sourced

from university students or if groups of people are excluded due to health concerns.

In all three cases, biases in the training data are induced into the trained model and can harm its performance on the target data [24]. Knowledge of the bias early in the development process can help improve the data quality and mitigate its effect on the learned model.

Existing bias detection and mitigation strategies require the user to have a certain knowledge of the target domain, such as a representative sample. The bias can then be identified by comparing both domains. By up weighting underrepresented data points in the historical dataset and down weighting overrepresented ones [14], the model training can be calibrated accordingly. Similarly, the data points can be re-sampled to match the target distribution [158]. This strategy is suitable to tackle the first issue mentioned above (Figure 1.1 left). However, since this approach is based on weighting, it requires the historical data to cover the entire target domain. Suppose, for example, a study on cardiovascular disease was restricted to participants between the age 40 and 65 (see Section 4.4 for an extended discussion of this example). However, a trained model assessing the risk of an individual falling ill should be applied to all patients. In this case, weighting strategies will not be sufficient to generalize to other age groups. Generally, if the second or third issue mentioned above arises, existing bias mitigation strategies will not be sufficient.

Comparing the historical data with the target data can help identify distribution mismatches. However, a sample to compare with might not be available as a clear target is not always explicitly defined. For example, consider a fisherperson that wishes to learn about a lake's population. No information on the target distribution is available, which is why the fisherperson attempts the data gathering in the first place. She throws her fishing net day by day and lists the fish she catches. Depending on the fishing net she uses, small fish might be able to escape since it is too coarse, whereas large fish might be strong enough to set themselves free. This induces a bias that can neither be quantified nor mitigated using existing techniques since no target information is available. Nonetheless, creating awareness early on is crucial since the fisherperson could have replaced her net with a larger or smaller one if she had recognized the bias she was creating.

While the existing literature attempts to correct for biases in the model during training, we argue that some information on problems in data collection is present in the biased dataset itself. Extracting it early on can help support the data-gathering process by understanding flaws and shortcomings in the dataset and allowing for correction with subsequent measurements. That would grant the researcher the opportunity to improve her data quality on-the-fly and avoid costly re-measurements as well as fragile bias mitigation techniques later on. Hence, we aim to identify and mitigate biases when no target information is available.

1.1 Research Problem

Given only a potentially biased dataset and no further information, we assume that there exists a "correct" distribution, a *ground truth*. Connecting to the previously introduced example of the fisherperson, we consider the ground truth to be the true distribution of fish in a lake. This distribution could be represented by a random sample, such as all the fish inhabiting the lake. Both the ground-truth distribution and sample are unknown. The observed dataset is a (biased) subset of this ground-truth sample, such as those fish exhibiting a certain size that were caught by the fisherperson. Our task is to identify potential biases, i.e., to tell the fisherperson that some fish are missing and which ones they are. This matches the right situation in Figure 1.1 and leads to the following (informal) research problem:

> Given only a potentially biased dataset and no information on the ground truth, decide whether a bias with respect to the ground truth is present. If it is, locate it and suggest a way to mitigate it.

Note that this problem formulation matches that of *Sample Selection Bias* [137] with the difference that, in our case, no ground truth information is available. To the best of our knowledge, we are the first to state and attempt to solve this problem under the assumption that no ground truth or target

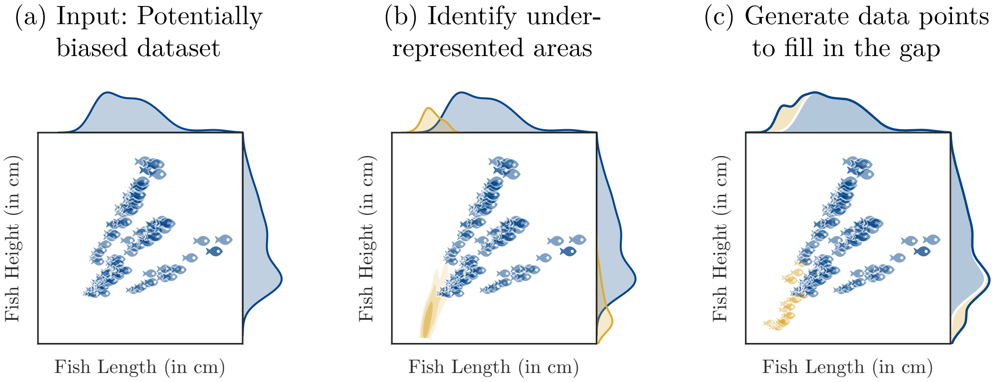


Figure 1.2: We simulate the scenario that a fisherperson is using a coarse net small fish can escape by removing all fish with a maximum diameter below 3cm from the Fish Market dataset [115]. Given a potentially biased set of fish measurements (a, blue), our goal is to identify the fish that did not get caught (b, gold) and generate additional data points that mitigate the bias (c, gold).

information is available. Chapter 3 provides an overview of related problems and highlights their different assumptions.

1.2 Proposed Solution

Aiming to identify a bias without any information on the ground truth, as stated in our research problem, we essentially need to "guess" the true distribution. This is a challenging problem that will likely be infeasible to achieve for all datasets. However, there are cases in which it is possible. To improve our chances, we make one fundamental assumption: We expect an unbiased dataset, i.e., the ground truth sample, to be smoothly distributed and attribute sudden drops in density to biases, such as the age thresholds in the cardiovascular disease example. Particularly for large datasets, this is well justified: The central limit theorem states that the deviation of the measurements from the true mean converges to a Gaussian with the dataset size. Hence we can expect it to be reasonably smooth unless some factors prevent this normal distribution, i.e., biases. We provide a more formal description of the central limit theorem and our expectations for data samples in Section 2.1.

Based on the distribution of the observed dataset, our central idea is to fit a smooth distribution. Data points can then be generated to fill in the gap between observed and fitted distribution. If the artificial points focus on certain areas, this could indicate a bias where these areas are underrepresented in the sample.

While we cannot expect these areas always to signal a true bias, they can create awareness of potential weaknesses of the dataset. A researcher can verify these areas by using additional data from other sources or domain knowledge. If a bias is identified, the researcher can either extend her data-gathering process or add the generated data to the model training procedure.

Figure 1.2 shows an example that reflects the real-world application: Given a potentially biased dataset, we would like to know where the bias is and then find a way to correct it. Note that in order to solve this problem in practice, we swap the last two steps, as described above. This swap brings computational advantages we exploit in our models but ultimately results in the same information.

1.3 Contributions

Using the previously introduced proposed approach, we make the following contributions to the research community:

1. We establish the novel problem of selection bias identification and mitigation under the assumption that no ground truth information is available. Designing our problem in a completely uninformed way is advantageous and allows us to challenge existing machine-learning procedures that accept flawed data as given and treat symptoms rather than causes: A solution can be applied at a very early stage of the data mining process. In particular, it can be applied during the data-gathering phase when it is still possible to improve the data quality instead of accepting it as an immutable fact. Furthermore, it is applicable later as a preprocessing step that helps prevent an induced bias in a model.

2. As a first attempt to solve the stated problem, we assume that the ground truth can be modeled as one multivariate Gaussian per class. Filling in the gap to the ground truth, we improve a model trained on the dataset without interfering with the algorithm or the loss function to be optimized. This constitutes a universally applicable preprocessing method that can be integrated into every machine learning pipeline. We implement this idea as the Imitate (*Identify and MITigATE Selection Bias*) algorithm that aims to “imitate” the ground-truth data in order to reveal a potential bias and mitigate it to enable the training of an unbiased model.

3. To expand Imitate’s scope, we model the ground truth as a mixture of potentially overlapping multivariate Gaussians per class. This assumption drastically increases the range of datasets and distributions that can be modeled, including multi-cluster settings. As an implementation, we propose Mimic (*Multi-IMItate Bias Correction*), which repeatedly uses Imitate to first find clusters and then identify potential biases.

4. We showcase the usefulness of our proposed methods and the interpretability of results in the context of chemical compound datasets. To deal with the unique challenges of these datasets, we propose Cancels (*CounterActiNg Compound spEciaLization biaS*), a specialized version of the Imitate algorithm. Using Cancels, we demonstrate that when adding bias-mitigating compounds from a pool of candidates to a dataset, the predictive performance of a trained model exceeds that of a model trained either on the original dataset or under the addition of the entire pool. This highlights the importance and strength of our bias mitigation techniques.

5. In extensive sets of experiments, we show that each of the proposed methods has its own merits and leads to interpretable results that can help a researcher understand her data better. Easy-to-use Python+sklearn [110] implementations of all methods are provided in the PyPI package imitatebias. Cancels will additionally be inte-

grated into the enviPath website[1], where users can trial their datasets freely via a web interface.

The outcomes of this research have been presented in prestigious peer-reviewed international conferences and journals. In particular, IMITATE was presented at the International Conference on Data Mining (ICDM) 2020 [39], MIMIC was presented at the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD) 2022 [37]. CANCELS has been submitted to the Journal of Cheminformatics [38] and is awaiting its review.

1.4 Thesis Overview

The remainder of this work is organized in the following chapters.

Chapter 2 provides preliminary explanations on concepts relevant to the methods discussed in the following chapters and subsequently considered known.

Chapter 3 reviews related research on distribution shifts between historical and target data dealing with different kinds of transfer between learning tasks and/or domains. We highlight the differences and similarities to our problem formulation.

Chapter 4 formalizes the problem statement, motivates and introduces the IMITATE algorithm and thoroughly investigates its performance and limitations. IMITATE aims to "imitate" the ground-truth data in order to reveal a potential bias and mitigate it to enable the training of an unbiased model. It assumes a multivariate Gaussian ground truth.

Chapter 5 extends the IMITATE algorithm for a more general setting and proposes the MIMIC algorithm. MIMIC repeatedly uses IMITATE to model the ground truth as a mixture of potentially overlapping multivariate Gaussians per class. A corresponding set of experiments highlights strengths and limitations.

Chapter 6 provides the CANCELS algorithm, a specialization of IMITATE to the chemical compound space. This chapter ties the thesis together

[1] enviPath: https://envipath.org/

as it demonstrates our methods' usefulness in the wild and works out a real-world use case and in-depth analysis of the insights that can be gained using Cancels.

Chapter 7 concludes the thesis. Here, we discuss the remaining limitations and avenues to be explored in future research.

2 Preliminaries

This chapter ensures that the reader may find all necessary definitions and explanations relevant to the thesis. Since it does not contain novel ideas, it may be skipped and used as a look-up. However, for the interested reader, we provide an introduction to probability theory in Section 2.1. The probability theory section lays the foundation, covers all aspects of Gaussian distributions that are essential here, and discusses the fate of growing sample sizes using convergence theorems. Section 2.2 provides tools to estimate the density of datasets. Section 2.3 introduces data transformation techniques that transform a dataset into a new space with beneficial properties. Lastly, Section 2.4 offers ways to express relationships within a dataset by fitting suitable curves.

2.1 Probability Theory

We aim to investigate a potentially biased dataset's distribution and thereby identify biases and ways to mitigate them. A basic understanding of probability theory and distributions is required as a foundation, and we provide all the necessary concepts here. See Zwillinger [165, pp. 509–511] for details on these concepts. Particularly Gaussian distributions are essential to our work and are discussed subsequently. Lastly, we present three fundamental theorems in probability theory that will be required: the law of total probability, the law of large numbers, and the central limit theorem.

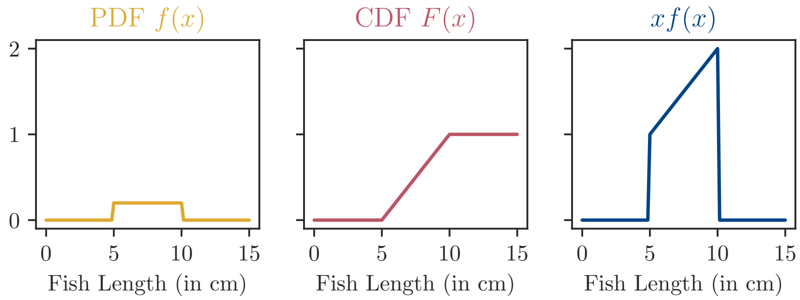


Figure 2.1: Visualizations for the running example in Section 2.1 – 'Foundations': Suppose a lake contains no fish below 5 cm or above 10 cm length, but all lengths within the interval are equally likely. Then the probability density function (PDF) f is given on the left, the cumulative distribution function (CDF) in the center, and $xf(x)$, the integral of which yields the expectation, on the right.

Foundations

For an experiment, each possible outcome corresponds to a unique element of a set, the *sample space* Ω. For example, the experiment could be catching fish in a lake and measuring their length, as discussed in the introduction. The sample space is then the set of fish swimming in the lake. A *probability space* consists of three parts: a sample space, a set of events $\mathcal{A}$, and a *probability measure* $\mathbb{P}$ able to assign probabilities to each of the events within the sample space. A *random variable* formally is a function $X\colon \Omega \to \mathbb{R}$ that maps any element of the sample space to a real number, such as a caught fish to its length. Since the sample space is a part of a probability space, we can measure probabilities that the random variable takes on certain values.

If existent, a *probability density function (PDF)* for X is a non-negative, integrable function f such that

$$\mathbb{P}[X \in A] = \int_A f(x)\,dx \qquad \text{and} \qquad \int f(x)\,dx = 1$$

for any event $A \in \mathcal{A}$. Suppose, for the sake of simplicity, there cannot be any fish of length below 5 cm or above 10 cm in a lake, but any length within this interval is equally likely. Then, the corresponding probability density function is $f(x) = 1/(10-5) = 1/5$ for $x \in [5\,\text{cm}, 10\,\text{cm}]$, and 0 otherwise. This way, f integrates to 1 and assigns equal likelihoods to all fish lengths within the interval. See Figure 2.1 (left) for a visualization.

The *cumulative distribution function (CDF)* essentially measures the area under the PDF below a threshold and is formally defined as

$$F(x) = \mathbb{P}[X \leq x] = \int_{-\infty}^{x} f(t)\,dt.$$

Since the probability density function integrates to 1, we can conclude that $F(-\infty) = 0$ and $F(\infty) = 1$. In the fish example, we have $F(x) = 0$ for $x < 5\,\mathrm{cm}$, $F(x) = (x-5)/(10-5)$ for $5\,\mathrm{cm} \leq x \leq 10\,\mathrm{cm}$, and $F(x) = 1$ for $x > 10\,\mathrm{cm}$. See Figure 2.1 (center) for a visualization. The probability that the output of the random variable lies between a and b is then

$$\mathbb{P}[a < X \leq b] = \mathbb{P}[X \leq b] - \mathbb{P}[X \leq a] = F(b) - F(a).$$

Hence, the probability that a fish is 7 to 8 cm long is $\mathbb{P}[7\,\mathrm{cm} < X \leq 8\,\mathrm{cm}] = F(8) - F(7) = 3/5 - 2/5 = 1/5$. Since $[7\,\mathrm{cm}, 8\,\mathrm{cm}]$ is exactly 1/5 of the full interval $[5\,\mathrm{cm}, 10\,\mathrm{cm}]$ and all lengths are equally likely, this is what we would expect intuitively.

The *expectation* quantifies the “best guess” for the outcome of a random variable X and is formally defined as

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} x f(x)\,dx.$$

Suppose we were to catch much fish from the lake and average their sizes. In that case, we would obtain the expectation (see the law of large numbers below for the mathematical foundation of this statement). Since we have an explicit PDF, we can alternatively calculate the expectation in the fish example, i.e., $\mathbb{E}[X] = 7.5\,\mathrm{cm}$. See Figure 2.1 (right) for a visualization of the integrand.

Of course, when catching fish, most fish will not be exactly 7.5 cm long. The *standard deviation* σ quantifies the dispersion of the caught fish’s lengths around the expectation. In other words, it measures the expected deviation from the expected value and is defined as

$$\sigma = \sqrt{\mathbb{E}[(X - \mathbb{E}[X])^2]} = \sqrt{\mathbb{E}[X^2] - (\mathbb{E}[X])^2}.$$

To avoid the square root, researchers often use the *variance* instead, which is exactly σ^2.

Two random variables X and Y are *independent* if knowing about one of their realizations does not inform about the other, i.e., if

$$\mathbb{P}[X \leq x \cap Y \leq y] = \mathbb{P}[X \leq x] \cdot \mathbb{P}[Y \leq y] \qquad \text{for all } x, y.$$

This assumption is violated, for example, in time series data where observations depend on previous observations. One essential assumption in machine learning is *independent and identically distributed (i.i.d.)* random variables. That is, X and Y are independent as defined previously and share the same PDF and CDF.

If X and Y were *dependent*, the above equation would not hold. Instead, we would express the joint probability via *conditional probabilities*:

$$\begin{aligned} \mathbb{P}[X \leq x \cap Y \leq y] &= \mathbb{P}[X \leq x \mid Y \leq y] \cdot \mathbb{P}[Y \leq y] \\ &= \mathbb{P}[Y \leq y \mid X \leq x] \cdot \mathbb{P}[X \leq x] \qquad \text{for all } x, y \end{aligned}$$

because the intersection of events is symmetric. Here, $\mathbb{P}[X \leq x \mid Y \leq y]$ is the probability that $X \leq x$ if we already know that $Y \leq y$. Since the events are dependent, the information about Y influences the probability for X.

A direct consequence of the symmetry of the intersection (as in the above equation) is *Bayes' theorem*:

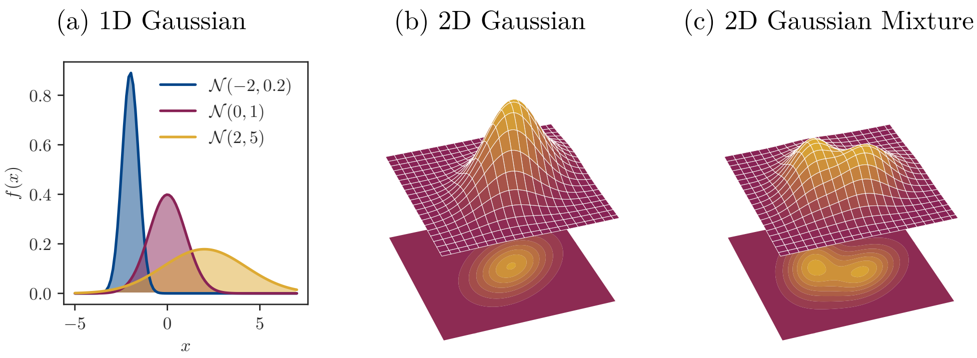
$$\mathbb{P}[A \mid B] = \frac{\mathbb{P}[B \mid A]\,\mathbb{P}[A]}{\mathbb{P}[B]} \qquad \text{for all events } A, B.$$

Normal/Gaussian Distribution

The *normal distribution* or *Gaussian distribution* is a continuous probability distribution for a 1-dimensional real-valued random variable X. Its probability density function is given by

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right] \tag{2.1}$$

for $x \in \mathbb{R}$, where $\mu = \mathbb{E}[X] \in \mathbb{R}$ denotes the mean of the distribution (or the expectation of X), and $\sigma \in \mathbb{R}$ its standard deviation [113]. In short, we write $X \sim \mathcal{N}(\mu, \sigma^2)$ to indicate that X is normally distributed with these


(a) 1D Gaussian

(b) 2D Gaussian

(c) 2D Gaussian Mixture

Figure 2.2: Examples of Gaussian probability density functions. **Left:** different 1D distributions; **Center:** the 2D distribution $\mathcal{N}\left(\begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}1 & -0.5\\-0.5 & 1.5\end{bmatrix}\right)$; **Right:** 2D Mixture of the previous distribution and $\mathcal{N}\left(\begin{bmatrix}-1\\-1\end{bmatrix}, \begin{bmatrix}1.5 & -0.5\\-0.5 & 1\end{bmatrix}\right)$.

parameters. See Figure 2.2a for an overview of how the parameters can shape the density function.

The univariate Gaussian can be generalized to a *multivariate Gaussian distribution* for a multivariate random variable. For d dimensions, the mean $\mu \in \mathbb{R}^d$ is a vector and the variance σ^2 is replaced by the covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$. The covariance matrix for a d-dimensional random variable $X = [X_1, \ldots, X_d]$ can be calculated as $\Sigma_{i,j} = \mathbb{E}[(X_i - \mu_i)(X_j - \mu_j)] = \mathrm{Cov}[X_i, X_j]$, which makes it symmetric by definition. If the covariance matrix is *positive-definite* (that is, symmetric and all eigenvalues are real and positive), the multivariate Gaussian distribution has density

$$f(x_1, \ldots, x_d) = \frac{1}{\sqrt{\det \Sigma \cdot (2\pi)^d}} \exp\left[-\frac{1}{2}(x - \mu)^{\mathrm{T}} \Sigma^{-1} (x - \mu)\right] \qquad (2.2)$$

for $x_i \in \mathbb{R}$. Here, $\det \Sigma$ is the determinant of Σ [165, p. 527]. Figure 2.2b shows an example of a 2-dimensional Gaussian probability density.

The condition of a positive-definite covariance matrix is particularly helpful when generating synthetic data drawn from random multivariate Gaussian distributions because of the following theorem [165, p. 93].

> **Theorem** (Cholesky Decomposition)**.** *Let A be a symmetric positive-definite matrix. Then, there exists a factorization*
>
> $$A = LL^T,$$

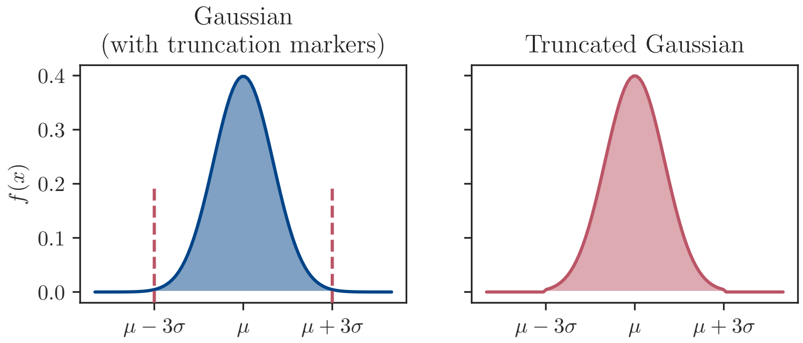


Figure 2.3: A Gaussian (left) is truncated at $\pm 3\sigma$ (right).

where L is a real lower triangular matrix with positive entries on its diagonal.

Hence, instead of generating covariance matrices and subsequently checking if the positive-definite condition holds, we generate L and calculate the corresponding covariance matrix. We use this method in Section 5.3 to generate multivariate synthetic Gaussians.

To expand the scope of what shapes multivariate Gaussians can model, one can consider *Gaussian mixture models (GMMs)* [113]. Here, multiple Gaussians are linearly combined to model new shapes. See Figure 2.2c for an example of a mixture with 2 Gaussians. The density of the mixture can be expressed as

$$f_{\text{mix}}(x) = \sum_{i=0}^{k} c_i f_i(x) \qquad \text{for parameters } c_i \in \mathbb{R} \text{ with } \sum_{i=0}^{k} c_i = 1.$$

Here, k is the number of individual Gaussians with density f_i. This definition translates directly to the multivariate case, where x and the f_i are multivariate.

For normal distributions, $f(x) > 0$ for all $x \in \mathbb{R}$. This can be inconvenient when dealing with multiple Gaussian clusters and assigning data points to clusters based on their distribution. As a compromise, we consider *truncated Gaussians* for a cleaner cut between clusters. To obtain a truncated PDF, the PDF of a regular Gaussian is chopped from below, above, or both. Expressing the general truncated PDF explicitly is substantially more complex than the regular Gaussian (see Burkardt [22]); however, we only remove ≈ 0.2% of the area under the PDF by cutting at $\mu \pm 3\sigma$. Therefore, our truncated PDF of $\mathcal{N}(\mu, \sigma^2)$ with PDF f can be approximated by

$$f_{\text{trunc}}(x) \approx \begin{cases} 0, & \text{if } x < \mu - 3\sigma. \\ \frac{1}{0.998} f(x), & \text{if } x \in [\mu - 3\sigma, \mu + 3\sigma]. \\ 0, & \text{if } x > \mu + 3\sigma. \end{cases}$$

Figure 2.3 illustrates this procedure. As we can see, the overall PDF changes very little; however, truncating the Gaussians brings an advantage: We can now meaningfully distinguish between areas in the space that are "covered" by the distribution and others that are "untouched". The *support* of a function is defined as the subset of the space that is mapped to non-zero values. In the case of the original Gaussian PDF, the support would be the entire space $\mathbb{R}$. For the truncated Gaussian, however, it is the interval $[\mu - 3\sigma, \mu + 3\sigma]$, which grants us meaningful information on which parts of the space the density focuses. Note that although truncated Gaussians can be advantageous, we actively create false distributions here and, in the cluster example, misassignments of points to clusters.

Law of Total Probability

The *law of total probability* is an elemental result in probability theory, according to which the probability of an event can be calculated via its partial probabilities conditioned on a partition of the sample space [113].

> **Theorem** (Law of Total Probability)**.** *Let $B_1, \ldots, B_n$ be a partition of a sample space, that is, the B_i are pairwise disjoint, and their union equals the entire space. Then, for any event A in the same space holds*
>
> $$\mathbb{P}[A] = \sum_{i=1}^{n} \mathbb{P}[A \mid B_i]\mathbb{P}[B_i] = \sum_{i=1}^{n} \mathbb{P}[A \cap B_i].$$

This theorem is particularly useful in cases where conditional probabilities are more readily accessible; for example, the probability for a specific feature is often easier to approximate if its class is known.

Law of Large Numbers

Various formulations for a *law of large numbers* have been proposed using different assumptions. However, the central idea is similar. Here, we choose to include Borel's version due to its simplicity as it does not require a long chain of preliminary definitions [149].

> **Theorem** (Borel's Law of Large Numbers)**.** *Let $S_n(E)$ be the number of occurrences of an event E with probability p in the first n trials of a repeated random experiment. Then, with probability one,*
>
> $$\frac{S_n(E)}{n} \xrightarrow{n\to\infty} p.$$

See Wen [149] for a proof. This theorem states that if an experiment is repeated (independently and under the same conditions) a large number of times, the empirical probability of an event is similar to its true probability. The larger the number of repetitions, the closer these two probabilities become. Therefore, if a dataset is sufficiently large, for every feature, the empirical distribution of its realizations will be close to the true distribution. Note that convergence is not always guaranteed, and it depends on the variability and complexity of the underlying population distribution, the sampling method used, and the presence of any biases or confounding factors in the data.

The Central Limit Theorem

The *central limit theorem (CLT)* states that, for a sufficiently large number of independent random variables, the distribution of their sum is approximately Gaussian [113]. We provide the theorem formally before discussing its implications.

> **Theorem** (Central Limit Theorem after Lindeberg-Lévy)**.** *Let $X_1, \ldots, X_n$ be a sequence of i.i.d. random variables, each with mean μ and finite variance $\sigma^2 < \infty$, and let $\bar{X}_n = n^{-1}(X_1 + \cdots + X_n)$ be the average over the first n samples. Then, the random*

variables $\sqrt{n}(\bar{X}_n - \mu)$ *converge in distribution to a Gaussian* $\mathcal{N}(0, \sigma^2)$ *as* $n \to \infty$*:*

$$\sqrt{n}(\bar{X}_n - \mu) \rightsquigarrow \mathcal{N}(0, \sigma^2),$$

where $\rightsquigarrow$ *denotes convergence in distribution.*

The presented version of the CLT is the traditional one. Since then, different versions have been proposed that drop the condition of identical distributions (CLT after Lyapunov) and independence of the random variables [66]. See Pfeiffer and Schum [113] for proof of the classical version.

The central limit theorem explains why normal distributions, or at least approximately normal distributions, are so frequently observed in nature. We can assume that real-world measurements do not measure raw signals but rather the effect of many individual causes. These causes in themselves are effects of a set of causes, et cetera. The further we unravel this thought, the clearer it becomes that measurements frequently combine many individual signals. The more signals contribute to the measured effects, the more likely it is to resemble a Gaussian following the central limit theorem. Although most of the measured effects will not be perfectly normally distributed, a Gaussian often yields a reasonable approximation [94].

2.2 Density Estimation

Machine learning and data mining aim to learn concepts and deduct models from data. This data, the *training data* or *historical data* X over a feature space $\mathcal{X}$, consists of real-valued d-dimensional vectors $x = [x_1, \ldots, x_d] \in \mathbb{R}^d$ called *examples* or *data points*. These can be accompanied by labels y, but they are irrelevant in this section and hence omitted. Note that it is not a coincidence that we use the same letter X for the dataset as for a random variable. As is common in the literature (for an example, see Pan and Yang [106]), we use a dataset in the role of a random variable. Although $\mathbb{P}[X]$ is typically estimated via the data, due to the law of large numbers, this is a close approximation of the true 'theoretical' probability (that would be denoted using the random variable) for sufficiently large datasets. Where

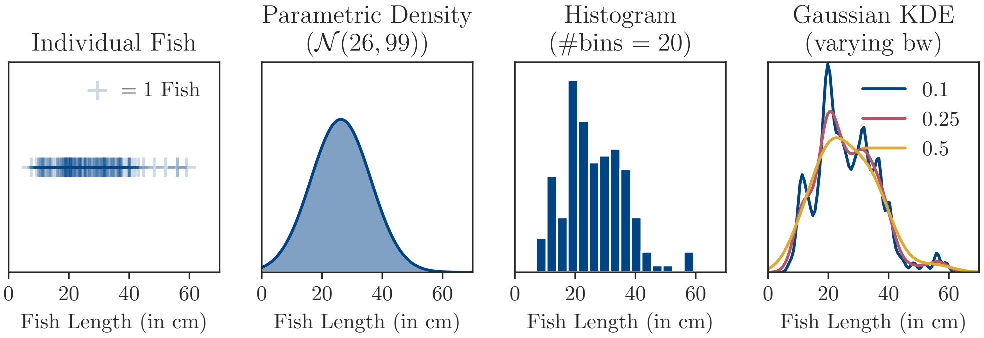


Figure 2.4: Comparison of different density estimators for the length distribution of fish from the Fish Market dataset [115], including kernel density estimation (KDE) with varying bandwidths (bw).

the distinction is necessary, we point it out throughout the thesis. In this section, we briefly review approaches to model the probability density of datasets, how they can be estimated from the data, and how to use them to identify outliers.

Parametric Density Estimation

Given a sample of data points from an unknown distribution, the *parametric density estimation* approach assumes a specific distribution, such as the normal distribution, and then finds the Gaussian that fits the data best [130].

In particular, we choose a specific family of density functions, such as the Gaussian family in Equation 2.1 (or Equation 2.2, depending on the data dimensionality). Then, we search for the concrete parameters μ and σ^2 such that $\mathcal{N}(\mu, \sigma^2)$ represents the dataset the best. In the case of one Gaussian, this is a simple task: calculating the mean and standard deviation (or covariance matrix in the multivariate case) of the dataset already yields a suitable estimate for the parameters. See Figure 2.4 for an example.

However, when fitting a Gaussian mixture model, the task becomes substantially more complex as we need to estimate the parameters of all individual Gaussians. This optimization problem is typically solved using the *expectation-maximization (EM)* algorithm [35]. The assumption behind EM is that each data point in our dataset has been generated by one of k individual Gaussians, but the concrete assignment is unknown. EM uses this assumption to break the overall optimization into two steps, i.e., assigning the points given the parameters and finding the best parameters given the assigned points. Given an (often random) initial set of parameters, EM alternates between these two steps until some convergence criterion is met. In the *E-Step*, EM calculates for each point the probabilities that it was drawn from each of the Gaussians. We call them membership probabilities. In the *M-Step*, EM optimizes each Gaussian individually for its members, weighted by these membership probabilities. See Dempster, Laird, and Rubin [35] for a detailed outline of the algorithm.

An improvement in the convergence speed of the EM algorithm can be made using a more informed initialization than random parameters. Frequently used is an initial clustering using the *k-means* algorithm with k clusters. The initial Gaussians are then centered at the identified cluster centers. k-means operates similarly to EM. It initially selects k points in the dataset as the initial cluster centroids. Alternatingly, it assigns all points to the nearest centroid and then refines the centroid as the mean of the assigned points. k-means is fast, and the found centroids are often a good starting point for EM, reducing the number of required iterations until convergence by a large portion.

Although Gaussian mixture models can already model substantially more general distributions than a single Gaussian, they are still limited by the number of individual Gaussians to be included. Using too few Gaussians leads to underfitting (the corresponding error of the model is often referred to as the "bias"). Using too many Gaussians leads to modeling the random noise in the data and hence to overfitting (the corresponding error of the model is often referred to as the "variance"). Note that the terms "bias" and "variance" are *not* used in the above sense in this work. Unfortunately, finding the right number of Gaussians that leads to the best model is not straightforward since we cannot distinguish between a true signal and one observed due to noisy behavior.

A compromise is to use *information criteria* to trade off model complexity against how well the model fits the data [57]. Two examples we use later on are the *Bayesian information criterion (BIC)*

$$\mathrm{BIC} = o \ln n - 2 \ln \hat{L}$$

and the *corrected Akaike information criterion (AICc)*

$$\mathrm{AICc} = 2o = 2\ln \hat{L} + \frac{2o^2 + 2o}{n - o - 1}.$$

In both cases, n denotes the sample size, o the number of parameters (for a GMM, that is twice the number k of individual Gaussians plus $k-1$ weights for the mixture), and $\hat{L} = \mathbb{P}[X \mid M]$ is the likelihood of the observed data X given the best model M with o parameters. Both criteria are heuristics and do not guarantee to find the optimal model. BIC typically penalizes model complexity more strongly than AICc.

Finding the right number of individual Gaussians requires trying a large set of numbers, calculating the preferred information criterion for each, and selecting the optimum. While this is computationally expensive, a *probabilistic clustering* of the input dataset is obtained as a by-product.

An alternative criterion typically used to find the number of clusters in a dataset is the *silhouette coefficient* [57]. For a data point x, let $a(x)$ be its mean distance to all points within the same cluster, and $b(x)$ be the mean distance to its next closest cluster. The silhouette of a data point is defined as

$$s(x) = \begin{cases} 1 - a(x)/b(x), & \text{if } a(x) < b(x). \\ 0, & \text{if } a(x) = b(x). \\ b(x)/a(x) - 1, & \text{if } a(x) > b(x). \end{cases}$$

The silhouette coefficient expressing the cohesiveness of a clustering is then the average silhouette and a larger coefficient corresponds to better clustering.

Nonparametric Density Estimation

Parametric density estimation is rather rigid in that one has to choose the family of density functions beforehand, which limits the expressivity of the model. In contrast, *nonparametric density estimation* is more flexible as it does not constrain the density to a particular parametric family and hence can capture the flaws and uniqueness of datasets [130].

Due to its simplicity and computational benefits, a widely used nonparametric density estimator is the *histogram* [130]. First, a range needs to be identified over which the density will be evaluated. This can be easily inferred from the dataset using the range from minimum to maximum, potentially with a padding range around it. Second, the range is split into equidistant grid cells or bins. We denote the number of bins with #bins. Third, for each cell, we count the number of measurements that fall into the cell and assign the count to the cell. To obtain a probability-like result, the cell counts can subsequently be normalized so they add up to one. Figure 2.4 shows an example. The main limitation of histograms is that their result strongly depends on the choice of the grid. Extensions have been proposed that use non-equidistant grids [130]; however, they sacrifice the model's simplicity.

More sophisticated and widely popular due to its smoothness is *kernel density estimation (KDE)* [130]. Here, we describe *Gaussian KDE* as this version of KDE is particularly popular and used subsequently. For a more general formulation of kernel density estimation, we refer the reader to Silverman [130].

The histogram can be seen as stacking small boxes of height one and width according to the grid's width. For each observation, we add one of these boxes on top of the stack of the corresponding grid cell. If instead of on the grid cell, we were to stack the boxes right where their measurement lies, we would overcome the dependency on the concrete grid location and obtain a more robust result. Gaussian KDE spins this thought further and, instead of boxes, stacks small Gaussians to obtain smoother results.

Formally, for any point $x \in \mathbb{R}$, the Gaussian kernel density estimator $\hat{f}$ with f being the PDF for $\mathcal{N}(0,1)$ can be evaluated to

$$\hat{f}(x) = \frac{1}{n \cdot \mathrm{bw}} \sum_{i=1}^{n} f\left(\frac{x - x_i}{\mathrm{bw}}\right),$$

where the x_i are the n observations in our dataset, and bw is the bandwidth [130]. The bandwidth controls the smoothness of the overall result. A small bandwidth corresponds to narrow, little Gaussians and will emphasize noise in the dataset. A large bandwidth suppresses the noise and focuses on the general trend of the distribution. Usually, an in-between solution is favorable in practice. See Figure 2.4 (right) for a comparison of different bandwidths.

Silverman [130] suggests a rule-of-thumb to choose the bandwidth for a given dataset with n data points in d dimensions, i.e.,

$$\mathrm{bw} = (n(d+2)/4)^{-\frac{1}{d+4}}.$$

Unless specified explicitly, we use this method to determine the bandwidth for the Gaussian KDE.

Outlier Detection

Using density estimators, for any example $x \in \mathbb{R}$, we can estimate the *likelihood* $\hat{L} = \mathbb{P}[x \mid M] = \hat{f}_M(x)$ of x given the model M capturing the density. Here, the likelihood is the corresponding estimated density $\hat{f}_M$ evaluated in x. If the likelihood of a point is low, it is likely an *outlier*. However, deciding whether a point is an outlier requires choosing a threshold for the likelihood, separating outliers from inliers.

Local outlier factor (LOF) [21] is a parameter-free alternative. Let $d_{k\mathrm{NN}}(x)$ be the distance from x to its kth-nearest neighbor, and $N_k(x)$ the set of x's k nearest neighbors, including ties in the case of equally distant neighbors. The reachability distance $\mathrm{rd}_k(x, x') = \max\{d_{k\mathrm{NN}}(x'), d(x, x')\}$ between two points x and x' is the distance d between these points, but at least $d_{k\mathrm{NN}}(x')$ to ensure that all nearest neighbors are assigned the same distance for statistical stability. The local reachability density $\mathrm{lrd}_k(x) = |N_k(x)| \,/\, \sum_{x' \in N_k(x)} \mathrm{rd}_k(x, x')$ is the inverse of the average reachability distance of x from its neighbors. Finally, the local outlier factor is

$$\mathrm{LOF}_k(x) = \frac{\sum_{x' \in N_k(x)} \mathrm{lrd}_k(x')}{|N_k(x)| \cdot \mathrm{lrd}_k(x)},$$

which is the quotient of the neighbors' local reachability density and that of x. If $\mathrm{LOF}_k(x) \leq 1$, x has a density that is at least as high as that of its neighbors, and we consider x an inlier. If $\mathrm{LOF}_k(x) > 1$, it is considered an outlier. See Breunig *et al.* [21] for details. The authors demonstrate that LOF's performance is insensitive to the choice of k as long as k is sufficiently large, such as $k = 20$.

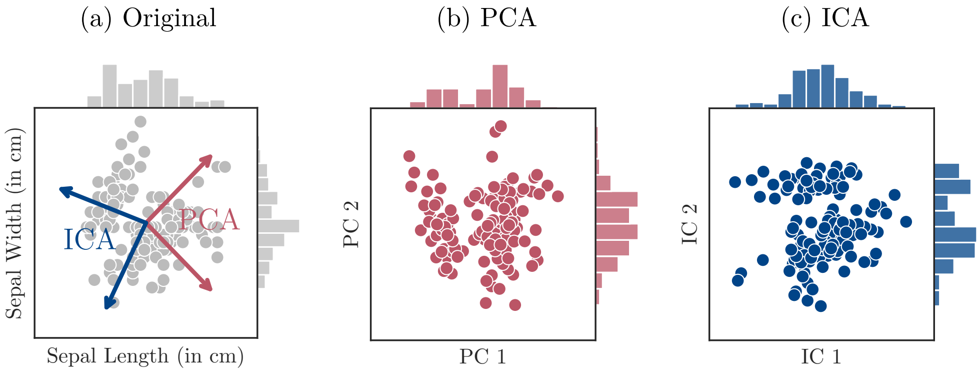


Figure 2.5: Visualization of individual data points and their marginal distributions after transformation of the Iris dataset [41] using PCA and ICA.

2.3 Data Transformation

Linear *data transformation* techniques transform a dataset X into a new space, ideally with some beneficial properties, where the axes are linear combinations of the original axes. This type of transformation is inexpensive to compute as it can be expressed as a matrix multiplication

$$x \mapsto Ix =: x',$$

where each data point $x \in \mathbb{R}^d$ in the original space is mapped to $x' \in \mathbb{R}^{d'}$ in the new space by multiplication with the transformation matrix $I \in \mathbb{R}^{d' \times d}$. If both spaces share the same dimensionality, i.e., if $d = d'$, the transformation retains all information contained in the dataset if it is a bijection. A bijective function maps each input element to exactly one output element and ensures that no two input elements are mapped to the same output.

However, data transformation techniques are also used to reduce the dimensionality of the space for $d' < d$. This can be beneficial, among other reasons, when the data is sparse in the input space. It is important to note that while reducing the dimensionality of the space can be largely helpful for subsequent modeling tasks, it is a one-way transformation and cannot be inverted.

This section reviews two popular approaches used subsequently: principal component analysis and independent component analysis.

Principal Component Analysis

The idea behind *principal component analysis (PCA)* is to find those orthogonal axes (called *principal components*) that explain the greatest amount of variance in the input data [74]. This way, using only the first few components and omitting the later ones will reduce the dimensionality of the data but preserve a large portion of the information contained in the data. Additionally, the resulting components are uncorrelated.

The principal components can be found as the eigenvectors of the dataset's covariance matrix and ordered in descending order based on the corresponding eigenvalues [74]. *Eigenvalues* can be found by solving the equation

$$\begin{aligned} \operatorname{Cov}(X)v = \lambda v \quad &\Leftrightarrow \quad \operatorname{Cov}(X) = \lambda \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{bmatrix} \\ &\Leftrightarrow \quad \det\left(\operatorname{Cov}(X) - \lambda \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{bmatrix}\right) = 0. \end{aligned}$$

for λ. The *eigenvectors* can then be obtained via $\operatorname{Cov}(X)v = \lambda v$, and the PCA transformation matrix A can be filled column-wise with the eigenvectors in the correct order [74].

As a pre-processing step, the dataset needs to be zero-centered. In order to ensure fair comparability of feature variation, the dataset can additionally be scaled such that each feature shows the same range. See Figures 2.5a and 2.5b for the identified principal components in the Iris dataset [41] and the PCA-transformed dataset, respectively.

Independent Component Analysis

Independent component analysis (ICA), in contrast, does not aim for orthogonal axes, but (approximately) statistically independent axes [70]. Independence implies uncorrelatedness, but the opposite direction does not necessarily hold.

Assume you find yourself in the *cocktail party problem*: you are in a room where two people speak simultaneously. Two microphones are placed in different positions in the room and record the signals x_1 and x_2, respectively. We can expect each of the recorded signals to be a combination of both individual speeches x'_1 and x'_2, i.e., $x_1 = a_{11}x'_1 + a_{12}x'_2$ and $x_2 = a_{21}x'_1 + a_{22}x'_2$ for parameters a_{ij} that depend on the location of the microphones with respect to the speakers. In matrix notation, this can be summarized as $x = Ax'$, where $X \in \mathbb{R}^d$ contains the recorded signals, $A \in \mathbb{R}^{d \times d}$ the parameters, and $x' \in \mathbb{R}^d$ the original signals. The goal of ICA is to recover the original speeches; however, neither A nor x' is known [70].

The first assumption to be made is that the x'_j are statistically independent. The second assumption is that the input data is normalized to zero mean and variance one. This is not a restrictive assumption since the input data can easily be normalized in a pre-processing step. Inspired by the central limit theorem, the third assumption is that the original components have non-Gaussian distributions, although the exact distributions are unknown. Non-Gaussianity of the x'_j is crucial since the joint density of the observed signal would not provide any information as to how they could have been mixed otherwise. Under these three assumptions, Hyvärinen and Oja [70] present multiple approaches to find A using numerical methods, including *FastICA* that iteratively searches for the least Gaussian axes. Finally, the transformation matrix is the inverse of the mixing matrix, i.e., $I = A^{-1}$. See Figures 2.5a and 2.5c for an example of identified independent components and the ICA-transformed dataset, respectively.

Whereas PCA orders the components by importance, in ICA, all components are equally important and not ordered. Hence, it is not advisable to reduce the dimensionality with ICA. Instead, the dimensionality can be reduced using PCA, and ICA can be applied subsequently. Since ICA searches for independent non-Gaussian components, it is of particular importance for our research.

2.4 Curve Fitting

For a dataset consisting of tuples $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, it can be helpful to find a curve (from a family of curves) that describes the behavior of y with respect to x. This scenario is similar to parametric density estimation (see Section 2.2), where a family of distributions was pre-selected, and the concrete parameters were optimized to represent the dataset. Here, instead

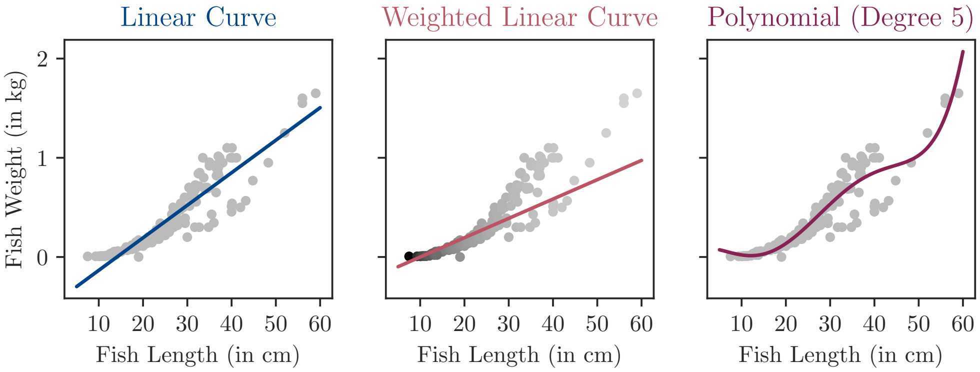


Figure 2.6: Fitted curves to express the weight of fish from the Fish Market dataset [115] with respect to their length. The center displays a weighted fit where each fish (x_i, y_i) was assigned the weight $w_i = 1/x_i^2$ during the optimization, as indicated by the shades of grey.

of optimizing the likelihood of the data given the model, we minimize the distance between the curve and the data.

In *least squares optimization* [99], this distance is squared (hence the name) to tackle different signs and obtain a differentiable loss function. Let $r_i \coloneqq y_i - f(x_i; \beta)$, where f is the family of curves to be fitted, and β is the corresponding set of parameters. The optimization problem translates to

$$\min_{\beta} \sum_{i=1}^{n} (y_i - f(x_i; \beta))^2 = \min_{\beta} \sum_{i=1}^{n} r_i^2 =: \min_{\beta} S.$$

The minimum can be found by setting the gradient to zero and solving the arising system of equations, i.e.,

$$\frac{\partial S}{\partial \beta_j} = 0 \quad \Leftrightarrow \quad 2 \sum_i r_i \frac{\partial r_i}{\partial \beta_j} = 0 \quad \Leftrightarrow \quad -2 \sum_i r_i \frac{\partial f(x_i; \beta)}{\partial \beta_j} = 0$$

for each partial derivative. If we were to fit a line as shown in Figure 2.6 (left), the family would be given by $f(x_i; \beta_1, \beta_2) = \beta_1 x + \beta_2$. Hence, the equations that need to be solved are $-2 \sum_i r_i x = 0$ and $-2 \sum_i r_i = 0$.

In some cases, some of the data points might be more important than others. To account for this effect during the optimization, we can define S as the weighted sum of squares $S = \sum_i w_i r_i^2$ with weights $w_i \in \mathbb{R}$ and solve the system of equations via the gradient as before. Figure 2.6 (center) shows an example where the left points are assigned a higher weight than the right ones leading to a different line.

Similarly to Gaussian mixture models, the complexity of the line needs to be considered carefully to avoid overfitting. Figure 2.6 (right) shows a polynomial of degree 5 that fits the data more closely than the degree 1 lines. However, using more complex models will overfit the noise in the data rather than represent a true trend. To overcome this, information criteria mark a suitable course of action, as well as penalization terms added to S that penalize model complexity [99].

In our work, we do not use least squares optimization to fit the data directly. Instead, we represent the data density as a histogram and fit a Gaussian to the histogram bin heights, given their position in the grid. Although unusual, this approach grants us more freedom to manipulate the weights than parametric density estimation would and, in contrast to nonparametric density estimation, provides us with a Gaussian density. We refer the curious reader to Chapter 4 to see this idea in action.

3 Related Research

Training on a biased dataset with respect to the target data means that, in order to perform well, the concepts and patterns found in the historical data need to be transferred to the target task. As humans, we are used to lifelong learning and transferring knowledge fluently between tasks and domains [112]. For example, having seen paintings of hedgehogs in a book, we will be able to recognize a hedgehog on the side of the road. Speaking French may help the process of learning to speak Portuguese. Metaphors and analogies in our everyday language are explicit connections between two different domains. Reading our work may spark ideas or draw connections in very different contexts. For machine learning models, this concept of transferring learned knowledge that is crucial when learning from biased data is not natively granted [106].

Many disciplines have been established under the machine learning umbrella that attempt to solve different kinds of shifts between datasets, distributions, and tasks. "Transfer learning" aims to transfer a trained model to a different domain or task [106, 108, 164]. As such, it is particularly popular in applications such as image recognition [109], where training a model from scratch would be extremely costly or infeasible due to the lack of data. "Domain adaptation" is closely related and often used interchangeably but is formally defined as a special case where only the marginal distributions between source and target domains change [106]. However, domain adaptation also has a different focus than transfer learning: Rather than transferring a model, it is concerned with leveraging labeled data from a different domain to avoid the expenses associated with labeling the target data [148]. “Dataset shift” has been introduced to summarize a set of problems driven by the statistics community rather than machine learning researchers [116]. Dataset shift problems generally aim to train a fresh model on a source dataset that can be expected to perform well on a target dataset under the assumptions that feature and label spaces remain consistent during the transfer and at least an unlabeled sample of the target data is available [100]. Driven by a diversity of application scenarios with corresponding induced assumptions, all three disciplines, i.e., transfer learning, domain adaptation, and dataset shift, face problems of overcoming biases and shifting knowledge from their unique perspectives. However, there is a non-negligible overlap between the disciplines’ core problems. Figure 3.1 visualizes the shared problem settings when approached from different perspectives and connects them to the research presented here.

Assuming equal feature and label spaces in source and target domains, “prior probability shift” refers to a shift in class distributions. “Concept drift” refers to a change in the relationship between features and labels. “Covariate shift” describes a change in the feature distribution. “Sample selection bias” occurs when a sample is drawn non-uniformly from a ground-truth (target) distribution and can be considered a cause for a covariate shift rather than a separate category of dataset shift [100]. However, there is a body of research concerned with specific solutions under the sample selection bias problem formulation, and it is the closest to our research. Hence, we consider it to be a separate category of dataset shift.

In this chapter, we demonstrate that dataset biases affect a wide range of real-life applications. Multiple research areas have been dedicated to mitigating the effects of biases in many different problem scenarios. However, they all share their acceptance of the data as it may be and treat the symptoms of the bias rather than the bias itself. While introducing these different problem scenarios, we highlight the assumptions regarding ground-truth knowledge these disciplines make and manifest our research gap: bias mitigation with no ground-truth information.

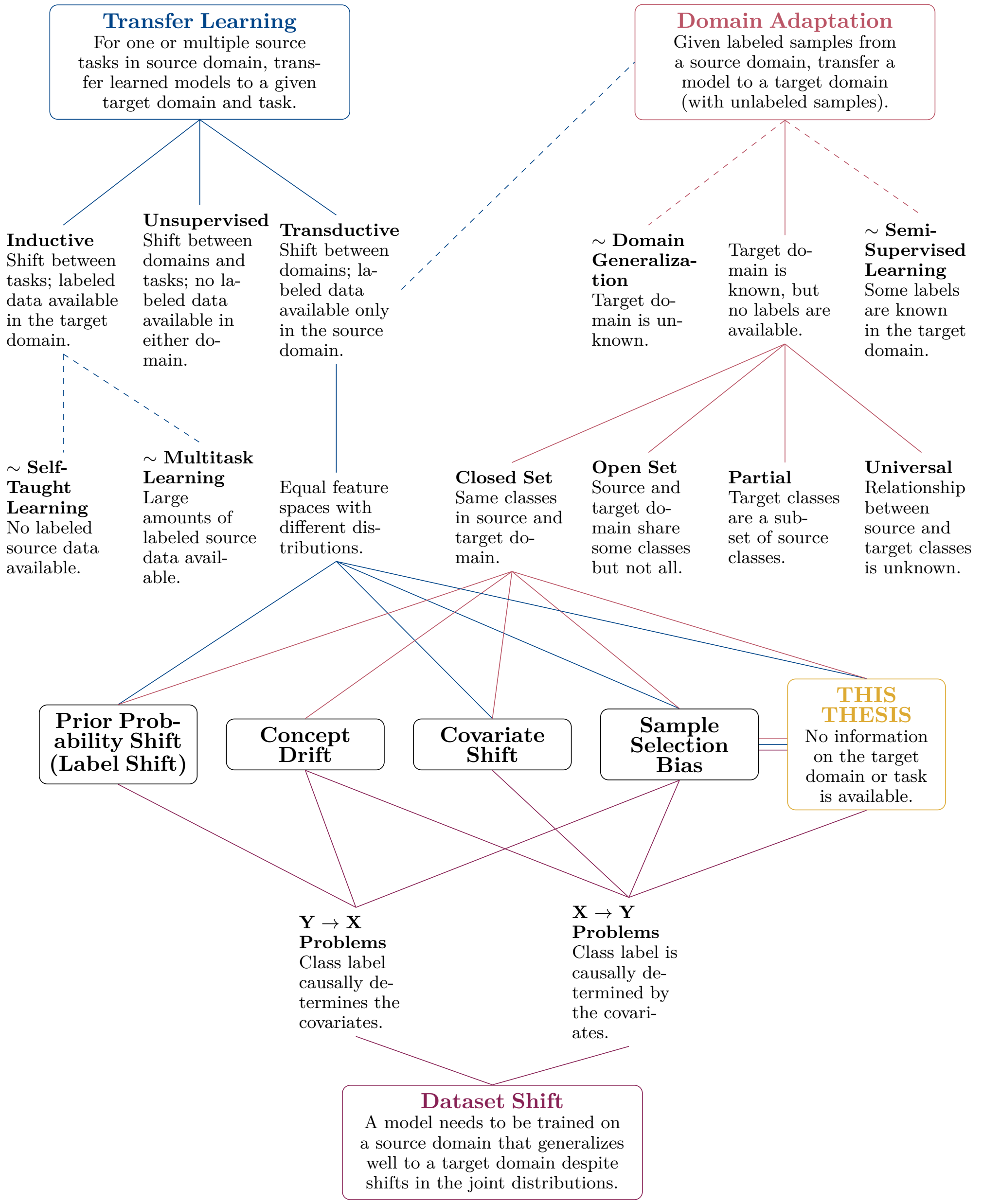

Transfer Learning
For one or multiple source tasks in source domain, transfer learned models to a given target domain and task.
Domain Adaptation
Given labeled samples from a source domain, transfer a model to a target domain (with unlabeled samples).
Inductive
Shift between tasks; labeled data available in the target domain.
Unsupervised
Shift between domains and tasks; no labeled data available in either domain.
Transductive
Shift between domains; labeled data available only in the source domain.
∼ Domain Generalization
Target domain is unknown.
Target domain is known, but no labels are available.
∼ Semi-Supervised Learning
Some labels are known in the target domain.
∼ Self-Taught Learning
No labeled source data available.
∼ Multitask Learning
Large amounts of labeled source data available.
Equal feature spaces with different distributions.
Closed Set
Same classes in source and target domain.
Open Set
Source and target domain share some classes but not all.
Partial
Target classes are a subset of source classes.
Universal
Relationship between source and target classes is unknown.
Prior Probability Shift (Label Shift)
Concept Drift
Covariate Shift
Sample Selection Bias
THIS THESIS
No information on the target domain or task is available.
Y → X Problems
Class label causally determines the covariates.
X → Y Problems
Class label is causally determined by the covariates.
Dataset Shift
A model needs to be trained on a source domain that generalizes well to a target domain despite shifts in the joint distributions.

Figure 3.1: Overview of related research areas when approaching from different perspectives. Dashed lines indicate similarities to other research areas, whereas solid lines stand for categorization of the color-coded umbrella terms “transfer learning”, “domain adaptation”, and “dataset shift”.

The remainder of this chapter discusses all problems integrated into Figure 3.1 in-depth and is organized as follows: Sections 3.1 and 3.2 discuss the research areas “transfer learning” and “domain adaptation”, respectively, that are currently in high demand. Section 3.3 discusses “dataset shift”, a third and more theoretical perspective on distribution shifts between datasets. These three research areas constitute the three perspectives from which we approach the core problems “prior probability shift”, “concept drift”, “covariate shift” and “sample selection bias”. The two latter ones are most closely related to our research and are discussed separately in Sections 3.4 and 3.5, respectively. Once the foundation is laid, we draw connections to related research areas. Section 3.6 discusses “imbalanced data” where there is an imbalance in the class distribution of the dataset, but a model should be trained that performs, for example, equally well on all classes. “Domain generalization” is a special case of domain adaptation where no target domain is specified. Here, a trained model is expected to generalize well to all similar potential target domains. We discuss domain generalization in Section 3.7. Section 3.8 provides an overview of “fairness in machine learning”, a research area concerned with avoiding discrimination against individuals or groups of individuals such as people of certain races, backgrounds, or ages due to dataset biases. Finally, we conclude the chapter with a discussion of existing detection methods for dataset shifts in Section 3.9.

3.1 Transfer Learning

Transfer learning [108] covers several kinds of transfer problems and aims to mimic our human learning using specialized strategies to train machine learning models under different transfer settings.

For a given target learning task $\mathcal{T}_T$ in a target domain $\mathcal{D}_T$, the idea of transfer learning is that there might be one or multiple source domains $\mathcal{D}_S$ with source tasks $\mathcal{T}_S$ that can be exploited to support and improve the training of the target task.

While a distribution shift can occur naturally (and even remain unnoticed), it can also be a strategic decision to reduce resource requirements for a model’s training process. For example, an already trained model might be transferred to the target dataset to reduce the computational burden imposed by the training process or because the target data would be too small to train the model there [104]. Either way, we are dealing with a bias between the source and target domain that needs to be accounted for during training.

Formally, a *domain* $\mathcal{D} = (\mathcal{X}, \mathbb{P}[X])$ corresponds to a feature space $\mathcal{X}$ together with a marginal probability distribution $\mathbb{P}[X]$ for $X \in \mathcal{X}$ [106]. Given such a domain and a label space $\mathcal{Y}$, a *task* is to learn a predictor $f\colon \mathcal{X} \to \mathcal{Y}$ from a training set that estimates the posterior probability distribution $\mathbb{P}[Y \mid X]$ for $Y \in \mathcal{Y}$ and $X \in \mathcal{X}$. Hence, a task can be written as the tuple $\mathcal{T} = (\mathcal{Y}, \mathbb{P}[Y \mid X])$. Following this notation, the transfer learning literature speaks of domains being different, i.e., $\mathcal{D}_S \neq \mathcal{D}_T$, if either $\mathcal{X}_S \neq \mathcal{X}_T$ or $\mathbb{P}_S[X] \neq \mathbb{P}_T[X]$ or both. Similarly, tasks are different if either the label spaces or the posterior distributions or both differ from one another. Note that traditional machine learning assumes equal source and target domains as well as tasks. While samples of both source and target domains are expected, depending on the concrete setting, they may not need to be labeled.

Prime examples of transfer learning are problems involving text data where large amounts of labeled data might be available from a different domain with a pre-trained model and can be transferred to the target domain, or task [33, 51, 103, 124].

Inherently, transfer learning deals with biases between source and target domains of different natures [108]: (i) *Inductive transfer learning* assumes equal source and target domains, but a shift in the tasks, and some available labeled data in the target domain, (ii) *transductive transfer learning* allows for different domains but expects the same task and a lot of labeled data in the source domain, and (iii) *unsupervised transfer learning* assumes both tasks and domains are different but related. The latter typically expects no labeled data and focuses on unsupervised methods.

Depending on the availability of labeled data in the source domain, inductive transfer learning can be split further [108]. If large amounts of labeled source data are available, the problem is similar to *multitask learning* [26], where multiple tasks are to be learned simultaneously. While multitask learning aims to learn all tasks equally, inductive transfer learning prioritizes one target task [106]. If no labeled source data is available, the problem setting resembles that of *self-taught learning* [118] where additional unlabeled data is used to train distinctive features that improve a target learner. Common strategies for inductive transfers evolve around careful selection, or iterative reweighting of training instances with respect to the new task [33, 146], the identification and transfer of shared features to bridge the gap between tasks [156], or the learning of shared parameters [93]. Regardless of the source data availability, inductive transfer learning expects complete visibility of the target domain and task, i.e., a labeled sample of the target is expected.

In transductive transfer learning, instead, no labeled target sample is required, but full visibility of the source domain needs to be granted. Researchers further distinguish between different feature spaces, which yields a problem setting similar to that of *domain adaptation* [6] (see Section 3.2), and identical feature spaces with different marginal probability distributions of the features in source and target domains. The latter is related to the problem settings of *covariate shift* [128], and *sample selection bias* [158]. Since these problem settings are closely related to our target problem, we discuss each of them individually in more depth in Sections 3.4 and 3.5, respectively. A general assumption in transductive transfer learning is that some unlabeled data from the target domain is available during training [106].

3.2 Domain Adaptation

Domain adaptation [6] is concerned with aligning the disparity between the source and target domain in order to train a machine learning model on the source domain that generalizes well to the target domain [82]. As such, it can be seen as a special case of transductive transfer learning.

Given labeled samples from a source domain and unlabeled samples of a target domain, domain adaptation aims to train a generalizable model. If the purpose of domain adaptation is to predict the labels of the provided target sample, it is called *transductive*. If, instead, the trained model is supposed to predict the labels of new samples in the target domain, we speak of *inductive domain adaptation* [82].

Depending on the relations between source and target label spaces, domain adaptation can be further split into the following categories [46]: (i) In *closed set domain adaptation*, both domains share the same classes, although their distributions may differ. Prior probability shifts, covariate shifts, concept drifts, and our research are considered to fall under this category [46]. (ii) *Open set domain adaptation* expects the source and target domain to share some but not all classes. For example, datasets containing images of wild animals in two countries' wildlife habitats might contain shared species and ones unique to the respective countries. (iii) If the target classes are a subset of the source classes, we speak of *partial domain adaptation*. An example would be a worldwide animal species database as the source domain when species modeling in a particular country is targeted. (iv) The most general case is *universal domain adaptation* where no prior knowledge about the label spaces is available. In this setting, researchers first need to identify common classes and place their problem into one of the previous three settings [46]. Ben-David *et al.* [12] investigate under what conditions a classifier trained on the source domain can generalize well to the target domain and provide bounds for the expected errors.

With the assumption of an unlabeled sample of the target domain, domain adaptation is enclosed by *semi-supervised learning* [45] and *domain generalization* [163]. In semi-supervised learning, a large pool of unlabeled data can be leveraged in tandem with a small labeled dataset, both from the target domain, to improve the quality of a trained model. In domain generalization, the target domain is entirely unknown. See Section 3.7 for details.

3.3 Dataset Shift

In 2009, Quionero-Candela *et al.* [116] coined the term *dataset shift* for problems in machine learning where the joint distribution of the source data differs from that of the target data, i.e., $\mathbb{P}_S[X, Y] \neq \mathbb{P}_T[X, Y]$. Three years later, Moreno-Torres *et al.* [100] made an additional effort to unify existing definitions and terminology regarding different types of dataset shifts. Although the umbrella term "dataset shift" is rarely used throughout the literature, the terms for the specific problems have become the standard.

Depending on the causal relationship between covariates and the class label, we can split learning problems into the following [47]: (i) $X \rightarrow Y$ problems, in which the values of the covariates causally determine the class label, and (ii) $Y \rightarrow X$ problems, in which the class label causally determines the values of the covariates. An $X \rightarrow Y$ example would be fraud detection, where the user's behavior causes the label. In contrast, medical diagnosis typically is a $Y \rightarrow X$ problem since the disease causes the symptoms [100].

Following the categorization by Moreno-Torres *et al.* [100], there are three types of dataset shift that can occur: *Prior probability shift*, *concept drift*, and *covariate shift*. *Sample selection bias* is often considered to be either a synonym for covariate shift [67], a special case or a cause thereof [100], or a separate type of dataset shift [137]. Since it is the closest to our research, we present it as a separate category.

Prior probability shift, also called *label shift*, refers to a change in class distribution, i.e., $\mathbb{P}_S[Y] \neq \mathbb{P}_T[Y]$ but $\mathbb{P}_S[X \mid Y] = \mathbb{P}_T[X \mid Y]$, and appears only in $Y \rightarrow X$ problems.

Concept drift refers to situations where the relationship between features and labels changes between source and target data. For $X \rightarrow Y$ problems, this means that $\mathbb{P}_S[Y \mid X] \neq \mathbb{P}_T[Y \mid X]$ while $\mathbb{P}_S[X] = \mathbb{P}_T[X]$. For $Y \rightarrow X$ problems, the roles of X and Y are swapped.

Covariate shift is the analog of prior probability shift for $X \rightarrow Y$ problems and is discussed in-depth in Section 3.4. Similarly to concept drift, sample selection bias is defined for both types of problems. We present the $X \rightarrow Y$ version in Section 3.5, however, the definitions for the $Y \rightarrow X$ problem can be obtained by swapping X for Y and vice-versa.

3.4 Covariate Shift Correction

Covariate shift describes the scenario that training and test set are "shifted" in terms of features, i.e., $\mathbb{P}_S[X] \neq \mathbb{P}_T[X]$, while the label distributions are invariant, i.e., $\mathbb{P}_S[Y \mid X] = \mathbb{P}_T[Y \mid X]$ [59, 100]. The goal is then to use the available source data to train a model that overcomes the bias and performs well on the target data. An often implicit assumption of covariate shift correction is that an unlabeled sample of the target domain is available to enable the shift.

Another critical assumption for a successful shift is that the *support* of $\mathbb{P}_T$, that is, the subset of the feature space with non-zero probability, is contained in that of $\mathbb{P}_S$ as the training set cannot be shifted towards the test set otherwise. This is an assumption we do not make. In fact, we assume that parts of the test space are not represented by the training dataset.

A recent application of covariate shift correction techniques is image re-identification, where a person needs to be spotted in images taken with different cameras in different environments and angles [133]. Another example of covariate shift occurs in the drug discovery process [96], where predictive models are trained on known drugs but expected to generalize to unexplored compounds.

Shimodaira [128] shows that minimizing a loss function l on an appropriately weighted training set is equivalent to minimizing the loss on the test set as

$$\mathbb{E}_{(X,Y)\sim\mathbb{P}_T}[l(X,Y,\theta)] = \mathbb{E}_{(X,Y)\sim\mathbb{P}_S}\left[\frac{\mathbb{P}_T[X,Y]}{\mathbb{P}_S[X,Y]} l(X,Y,\theta)\right].$$

We denote the weights as $\beta(X,Y) \coloneqq \frac{\mathbb{P}_T[X,Y]}{\mathbb{P}_S[X,Y]}$. As a result, assuming that there exists a way to estimate $\beta(X,Y)$, a classifier can be trained on the weighted training set and will be able to perform well on the test set.

The traditional approach to obtain β is *importance reweighting*. After estimating both probability distributions (either directly [158], via the selection probabilities [89], or via the class distributions if available [139]), their quotient evaluated for every training sample yields the weights. In all direct approaches, the quality of the obtained weights depends strongly on the estimates of the required measures. As density estimation typically struggles in high-dimensional applications [136], the weights will be compromised [59]. Smith and Elkan [131] suggest a subsequent gradual refinement of the weights using the expectation-maximization algorithm.

A strategy to avoid the density estimation and hence to improve the obtained weights is *kernel mean matching (KMM)* [67]. After mapping both sets into a *reproducing kernel Hilbert space (RKHS)*, the weights minimize the maximum mean discrepancy between the weighted training data and the test data. Under the assumptions made above regarding the label distribution invariance and the support, the weights obtained in the RKHS are proven theoretically to converge to β [157]. The KMM minimization problem can be formulated as a quadratic program. More efficient extensions involve repeated sampling of the test set [98] or the training set [28] before aggregating the obtained weights.

Sugiyama *et al.* [141] propose a direct approach, *Kullback-Leibler importance estimation procedure (KLIEP)*, that estimates β by minimizing the Kullback-Leibler divergence between the true test density and the weighted training density. This approach overcomes the need for intermediate density estimation and the implied performance drop in high dimensions. Similarly, Bickel, Brückner, and Scheffer [14] suggest learning a discriminative model that estimates the weights directly. In the domain of natural language processing, importance reweighting has been reported to perform poorly in many scenarios. Xia, Pan, and Xu [154] attribute this to the increased risk of overfitting to a few examples that have been weighted highly. To overcome this problem, the authors suggest introducing limits and penalization terms into the loss function when learning the weights directly.

A large body of research has been dedicated to quantifying the expected error in classification caused by the covariate shift. Tripuraneni, Adlam, and Pennington [145] provide a recent extension of existing estimates and an overview thereof. The authors prove that stronger shifts cause larger error gaps, and a linear relation exists between training and test error in the presence of a covariate shift.

3.5 Sample Selection Bias

Sample selection bias occurs when a sample is drawn non-uniformly from a ground-truth distribution, i.e., it forms a biased sample, and hence is not representative of this distribution [137].

Prime examples of this scenario are surveys or political polls via telephone or on the street. The accessibility and the participant's momentary mood determine the sample more than the desire to capture the entire population accurately [137]. Similarly, sample selection biases are found when browsing the news feed on social networks since people are more apt to post their successes than failures or standard events. The choice of news to be covered and broadcast is biased by limited resources, ideological affinities, information availability, and others [20]. Credit scoring models can only be developed based on applicants accepted in the past. Assuming that a bank does not hand out credits randomly but selects only candidates most likely to repay their loan, this selection makes for a biased subset [100]. Another example is species habitat modeling, where the data is typically biased towards the more accessible sampling sites [42].

To formally define sample selection bias, the literature typically introduces a selection variable $s\colon \mathcal{X} \to \{0,1\}$ that outputs for each data point if it is contained in the observed sample ($s = 1$) or not ($s = 0$). With the selection variable, we can express the joint distributions in the source and target domains, respectively, as:

$$\begin{aligned} & \mathbb{P}_S[X,Y] = \mathbb{P}[X,Y \mid s=1] = \mathbb{P}[s=1 \mid X,Y]\,\mathbb{P}[Y \mid X]\,\mathbb{P}[X] \\ \text{and} \quad & \mathbb{P}_T[X,Y] = \mathbb{P}[X,Y] = \mathbb{P}[Y \mid X]\,\mathbb{P}[X]. \end{aligned}$$

By definition, the support of the biased sample is contained in the support of the target domain [32]. Note that this is the essential difference between sample selection bias and covariate shift, where the target space is expected to be covered by the source support. Note that this scenario is substantially different from *missing values* [23, 114]. Rather than imputing single missing feature values for examples, sample selection bias deals with entirely missing examples.

There are different scenarios to consider under which the sample was obtained, that is, there are different dependencies of the selection variable s and an example $(x,y) \in \mathcal{X} \times \mathcal{Y}$ [100, 158]:

- The sampling method is completely independent of the features and labels. This means that s is independent of x and y, i.e., $\mathbb{P}[s=1 \mid x,y] = \mathbb{P}[x,y]$. In this case, the sample represents the ground truth and is not biased. The statistical literature speaks of *missing completely at random (MCAR)* [91].

- The sampling method is independent of the labels, i.e., $\mathbb{P}[s \mid x, y] = \mathbb{P}[s \mid x]$. This case is known as *missing at random (MAR)* [91]. In practice, these assumptions can be fulfilled by including all variables that led to the decision to include a sample. For example, in a medical treatment study, including the variables the doctor used to decide who obtains the treatment results in a MAR bias [158].

- The sampling method is independent of the features, i.e., $\mathbb{P}[s \mid x, y] = \mathbb{P}[s \mid y]$. This scenario corresponds to a prior probability shift [3, 30].

- No independence assumption can be made between x, y and s. This scenario is coined *missing not at random (MNAR)* [91] and constitutes the more severe case as it can induce multiple types of biases [100]. Here, no unbiased model can be learned from the biased sample unless we can access an additional but hidden variable x_s controlling the selection, i.e., $\mathbb{P}[s \mid x_s, x, y] = \mathbb{P}[s \mid x_s]$.

If a MAR bias is observed, a weighting technique similar to the importance reweighting approach for covariate shift correction can be employed to train a model suitable for the target domain. Following the *bias correction theorem* presented by Zadrozny [158], for a model with parameters θ and a loss function l, minimizing the loss on the target test set is equivalent to minimizing it on an appropriately weighted training sample because:

$$\mathbb{E}_{(X,Y)\sim\mathbb{P}_T}[l(X, Y, \theta)] = \mathbb{E}_{(X,Y)\sim\mathbb{P}_S}\left[\frac{\mathbb{P}[s=1]}{\mathbb{P}[s=1 \mid X]} l(X, Y, \theta) \,\middle|\, s = 1\right].$$

Hence, we can train an unbiased classifier on the biased sample if we weigh each training instance x, since $\mathbb{P}[s = 1]$ is constant for all x, with a weight $1/\mathbb{P}[s = 1 \mid x]$ [14]. While the results are guaranteed to be correct with perfect weights, those weights need to be estimated, which commonly introduces inaccuracy [32, 92]. The key to determining those weights is to model s. This can be done using prior information on the selection criteria or learning s using an unlabeled sample of the rejected instances (i.e., those instances with $s = 0$) [14, 131, 136].

In the case of MNAR bias, the selection bias can depend on features and labels. Using an additional sample of rejected instances, Zadrozny and

Elkan [159] suggest training one classifier that predicts the selection variable s before using it alongside the other observed features to train a model for the class labels. Tran and Aussem [144] demonstrate that importance reweighting is a valid option under the MNAR setting, even when the hidden variable x_s is only partially observed in the target set.

The problem we attempt to tackle, as formulated in Section 1.1, falls under the category of selection bias. We make no assumptions as to the dependencies of the selection variable and additionally restrict the bias detection and mitigation to only the source data. No information on potential target domains or samples thereof is provided. To the best of our knowledge, we are pioneering this field.

3.6 Imbalanced Data

In many real-world datasets with multiple classes, some classes might dominate others that are heavily under-represented. During the learning process, a machine learning model would most likely focus on the majority classes and accept errors in the minority classes since they contribute less to the loss function. This problem is called *imbalanced data* [77, 137].

Depending on the application, the correct prediction of the minority classes can be of great importance. For example, consider rare events such as loan defaulting in credit scoring models or detecting a rare disease in a large pool of blood samples. In both cases, we need to ensure that the positive samples are not dominated during the learning process.

To solve the imbalanced data problem, researchers typically choose to transform it into a distribution shift problem where the target distribution of labels is known explicitly: It should be approximately uniform. Hence, the problem can be seen as a special case of selection bias, and reweighting strategies can be used to emphasize the minority classes, such as the loan defaulters in the credit scoring examples. Alternatively, imbalanced data can be treated as a prior probability shift with a known shift [77, 137].

3.7 Domain Generalization

In contrast to domain adaptation, *domain generalization* aims to generalize a model from one or multiple distinct but similar source domains to any unknown differently distributed target domain [163].

The problem of domain generalization was first introduced by Blanchard, Lee, and Scott [16] and Muandet, Balduzzi, and Schölkopf [102] later coined the term. It is frequently used in applications such as object recognition, where a trained model might need to generalize to new environments or viewpoints, as well as the similar scenarios of action recognition, face recognition, speech recognition, and others [163].

Formally, given datasets drawn from similar but distinct source domains, the task of domain generalization is to train a model that minimizes the prediction error on an unseen target domain. During training, the target domain is not known. Depending on the number of source domain datasets, researchers distinguish between the more common *multi-source domain generalization* [16] and the less common *single-source domain generalization* [69, 147].

Popular approaches to solving the domain generalization problem include domain alignment, where the multiple source domains are exploited to learn domain-invariant features [102], data augmentation through a transformation of the provided examples, or ensemble strategies. We refer to Zhou *et al.* [163] for a recent and comprehensive overview of different approaches.

Single-source domain generalization is closely related to our research. The major difference is that domain generalization aims to generalize a specific model toward all related domains. In contrast, we aim to correct a biased dataset towards the unbiased ground truth. Our approach is independent of the choice of model or even learning task.

3.8 Fairness in Machine Learning

The literature on *fairness in machine learning* [52, 97, 134] focuses on biases towards certain individuals or groups of people resulting in unfair or discriminating model predictions. The characteristics of these groups, e.g., race, gender, age, or income of the individuals, have to be pre-defined and are considered "protected" attributes. Fairness problems are typically selection biases or covariate shifts. However, they assume categorical datasets with specific pre-defined features causing the biases. These changes result in fundamentally different mitigation strategies, as we present below.

"Fairness" is difficult to define and quantify, and many potential definitions have been proposed. Following Mehrabi *et al.* [97], those definitions can be roughly categorized into (i) *individual fairness* [44] where similar individuals should receive similar predictions, (ii) *group fairness* [83] where all groups (based on pre-defined criteria) should be treated equally, and (iii) *subgroup fairness* [78, 79] which aims to combine both. The choice of a suitable metric assessing fairness depends on the practitioner's interpretation of what fair means in their specific context; there is no one-size-fits-all solution [49].

Unfairness in machine learning predictions typically stems from either bias in the dataset or algorithmic bias. Methods to achieve fairness in the dataset have been proposed and are independent of subsequent tasks and models [97]. They mainly include attempts to improve data transparency by documenting the exact data gathering process, as well as standard descriptive statistics [54], visualization [18, 19, 88], or tests for the underrepresentation of certain clear-cut groups (like age or race groups) in tandem with mitigating sampling strategies [48, 75].

Approaches to improve algorithmic fairness are typically domain- and task-specific and can operate either as pre-, in-, or post-processing steps [97]. Bellamy *et al.* [11] provide guidance on when in the machine learning cycle the bias compromising the fairness of a trained model is best corrected (i.e., during pre-, in-, or post-processing) and offer a framework that integrates implementations of state-of-the-art approaches. Other toolkits to identify biases have been proposed: FairML [2], Themis [53], and FairTest [143] are auditing tools that test predictive models for biases with respect to protected attributes. Aequitas [123] audits the dataset rather than the model and employs several metrics to identify fairness breaches. Additionally, there are other toolkits that aim to not only identify but also mitigate the bias:

Themis-ML [8] and Fairness Comparison [50] both contain a subset of those bias mitigation strategies implemented in the AI Fairness 360 toolkit [11].

Bias mitigation techniques during pre-processing included in the AI Fairness 360 toolkit are the following. Kamiran and Calders [75] suggest either suppressing those attributes that correlate strongly with the sensitive ones, “massaging the dataset” by changing some labels to lower the influence of the unfair bias, reweighting towards a discrimination-free dataset, or sampling instead. Zemel *et al.* [160] learn suitable representations that obfuscate sensitive information while preserving as much individual information as possible. Calmon *et al.* [25] combine previous ideas into a probabilistic framework that allows trading off discrimination control against data utility.

Overall, these bias mitigation techniques rely on discrete features and particular protected attributes that are not allowed to impact the resulting predictions. Similarly to those problem settings presented in Figure 3.1, the bias mitigation strategies to achieve fairness have an ideal of what is fair in mind, that is, some form of ground truth to strive for. As such, they are substantially different from the problem we face.

3.9 Dataset Shift Detection

Independently of the concrete type of shift between source and target dataset, traditional machine learning techniques fail to adapt since they rely on the i.i.d. assumption of their inputs. Since the model development is carried out on one or multiple training and test sets that stem from the same source data, a shift will remain undetected during training [161]. To avoid poor performance of the trained model on the target data, it is crucial to test for shifts routinely during model deployment.

Assuming a gradually incoming stream of target data points the model should be applied to rather than a fixed target dataset, detecting a shift from as few examples as possible is key as it implies an early alarm.

For the first target point, the problem of detecting a distribution shift is essentially *outlier detection* [27]. The more target points come in, the more confident statistical two-sample tests or multiple univariate tests comparing both target and source samples can detect different distributions. However, those statistical tests scale poorly to large high-dimensional datasets [117]. Lipton, Wang, and Smola [90] suggest incorporating dimensionality reduction techniques and propose *black box shift estimation (BBSE)* to detect prior probability shifts. Rabanser, Günnemann, and Lipton [117] build on BBSE and investigate the impact of different methods and dimensionality reduction techniques on general dataset shift detection.

While these methods seem highly effective based on the presented experimental results, they require unlabeled target data to identify a distribution shift. In contrast, we focus on detecting biases inherent in the source data without any target data availability.

4 Single-Cluster Selection Bias Identification and Mitigation

The research presented in this chapter has been adapted from

> K. Dost, K. Taskova, P. Riddle, and J. Wicker, "Your best guess when you know nothing: Identification and mitigation of selection bias," in *20th IEEE International Conference on Data Mining (ICDM 2020)*, IEEE, 2020, pp. 996–1001. DOI: 10.1109/ICDM50108.2020.00115.

The results of this chapter are available in the GitHub repository github.com/KatDost/Imitate and the proposed algorithm is contained in the PyPI package imitatebias.

4.1 Introduction

Machine learning typically assumes that training and test set are independently drawn from the same distribution (*i.i.d. assumption*). However, this assumption is often violated in practice which creates a bias. As demonstrated in Chapter 3, many attempts to identify and mitigate the impact of this bias on a model have been proposed, but they usually rely on ground-truth information and build upon the assumption that the researcher is aware of the bias.

But what if the problem already appears in the data-gathering process? Unexpected selection biases can occur during the data collection phase. For example, consider the fisherperson collecting information about fish species and counts in a lake that we discussed in Chapter 1. Her choice of the fishing net might induce a bias if it limits the characteristics of fish that can be caught, such as their size. Costly re-measurement or fragile domain adaption approaches can often be prevented if the bias is identified during the data collection. For example, the fisherperson could have replaced her net if she had been aware of the bias she was creating.

Most machine learning techniques consider the data as given and mitigate the bias of the model instead of the dataset itself. In contrast to prior work, we aim to solve the problem of selection bias identification and mitigation on the dataset itself when (a) we may not know if we have a bias and (b) we have no ground-truth information on our dataset. Mitigating the bias in the dataset itself allows us to detect biases already in the early stage of data gathering, which might help the researchers improve the data quality by avoiding bias in the first place.

Although we do not know how the ground truth is distributed and hence do not know what to strive for, we believe that in many cases, there are indicators for a selection bias hidden in the biased dataset itself. Intuitively, we would expect a trained model to perform well on the domain it is designed for, and we allow a certain amount of error around the fringes and would not expect it to perform on entirely different data. This describes a Gaussian-like shape of density. We also expect a reasonably smooth data distribution, particularly for larger datasets. For example, for the fish measurements, the bias created by the choice of the net would cause a smooth distribution for larger fish and then a sudden drop in the fish size distribution when looking at smaller fish counts. This would violate the smoothness assumption.

Motivated by these intuitions, in this chapter, we propose IMITATE (*Identify and MITigATE Selection Bias*), a technique that checks the data distribution and generates points to match a smooth Gaussian probability density. If the artificial points focus on specific areas, this could indicate a selection bias where these areas are underrepresented in the sample. The researcher can verify these areas by either using additional data from other sources or by extending her data gathering. Although designed in a way that supports the data collection process, IMITATE is also applicable at a

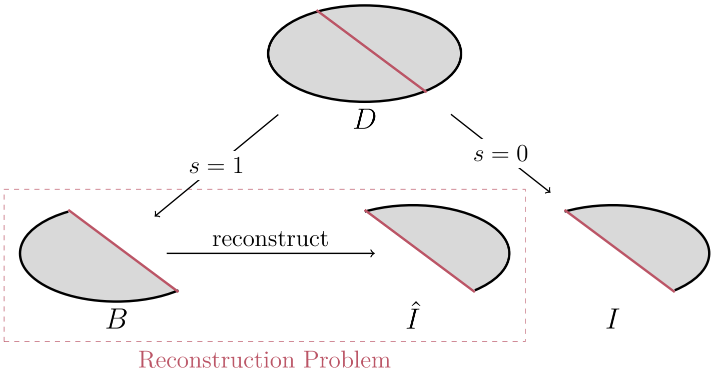


Figure 4.1: **Problem 2:** A subset B of a dataset D is drawn according to a selection attribute s. The task is to reconstruct $I = D \setminus B$, resulting in a dataset $\hat{I}$.

later stage as a preprocessing step that helps to prevent an induced bias in a model trained on the biased dataset.

The remainder of this chapter is organized as follows: Section 4.2 formalizes the general problem statement introduced in Section 1.1 and refines it to the additional assumptions made here. Section 4.3 introduces the IMITATE algorithm before Section 4.4 studies its behavior experimentally. Finally, Section 4.5 concludes with a discussion.

4.2 Problem Statement

Leaning on the problem formulated in the sample selection bias literature (see Section 3.5), we propose a method to solve the following problem statements that works in an unsupervised manner, i.e., it does not exploit any information about different classes. If it is applied in a supervised setting where several classes are present, the data is split according to the label, and then the method is applied separately. Since the supervised setting offers possibilities for evaluation and comparison beyond domain expert consultation or interpretation, we formulate both problem statements here. See Figure 4.1 for a visualization.

Problem 1 (Supervised Setting)**.**
Let $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\} \subset \mathbb{R}^n \times L$ *be an (unknown)* n*-dimensional real-valued labeled dataset consisting of* m *feature-label pairs* $(\mathbf{x}_i, y_i)$*.* D

is representative of an underlying distribution $\mathcal{D}$ that we consider as the ground-truth. A biased subset $B \subseteq D$ is drawn as follows: A selection variable s decides for each tuple $(\mathbf{x}_i, y_i) \in D$ if it is contained in B ($s = 1$), or discarded ($s = 0$). We consider s to be dependent on the present features or the class label. Given only B, the goal is to approximate $I := D \setminus B$ by a set $\hat{I}$ such that the gap between the accuracy of a classifier trained on D and that of a classifier trained on $B \cup \hat{I}$ is minimal.

Problem 2 (Unsupervised Setting)**.**
Assume the same setting as in the supervised case for the one-class case $|L| = 1$. Given only B, the goal is now to approximate $I := D \setminus B$ as well as possible such that the distribution of $B \cup \hat{I}$ reflects $\mathcal{D}$.

As described above, a solution for Problem 2 can be extended to one for Problem 1 by treating every class label separately. If Problem 2 were solved and returned a good approximation of I, a similar classifier performance would be guaranteed. Note that the assumption that only B is available is a very strong restriction, but it enables us to detect *potential selection biases* of which researchers might not have been aware. The outcome needs to be carefully evaluated together with domain experts or validated using additional data from other sources.

4.3 Proposed Method

The problems introduced in Section 4.2 are important but hard problems and cannot be solved in all cases, but we will show in this section that, in many cases, there is something we can do. For example, a dataset measures the occurrences of different flower types together with geospatial coordinates, but for a certain area there are no measurements. In the case that the measurements are missing because the bespoken area is restricted and not publicly accessible, solving Problem 2 makes sense, and our proposed method helps overcome the bias by imitating the measurements in this area. But if the data in this zone is missing because it is a lake and there are no flowers, an attempt to “reconstruct” the area would conceal the true distribution.

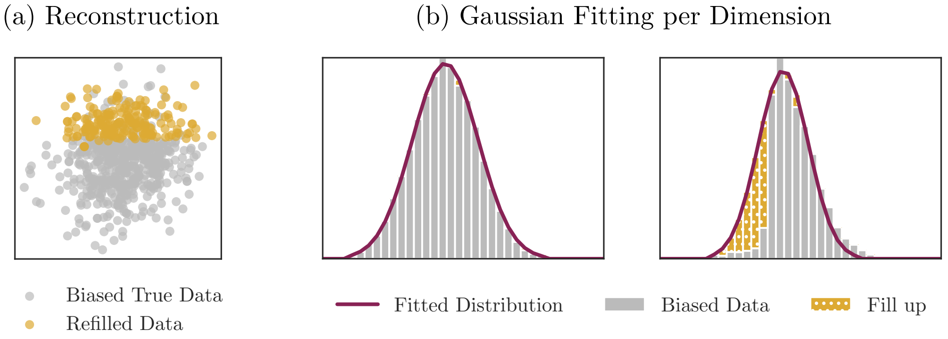


Figure 4.2: The figure shows the central idea of IMITATE: (a) shows a dataset (grey) with a clear bias that has been "reconstructed" (gold). (b) shows both dimensions separately (x-axis left, y-axis right). The dataset is represented by a histogram (grey), a normal distribution density is fitted to the histogram (pink line), and the gap between fitted and present density shows where to generate points (gold).

The only way to distinguish between these two cases is to consult domain experts.

Aiming to solve the problems wherever useful, we introduce IMITATE (*Identify and MITigATE Selection Bias*), a simple, modular and extendable approach.

IMITATE mainly checks for underrepresented parts in the distributions to identify missing zones and monitors the confidence in its own results to decide if the output should be reported or discarded since it probably reflects noise or algorithm-inherent problems. Figure 4.2 shows an example of the central idea. Given the biased dataset (grey in Figure 4.2a), IMITATE measures the density for each (transformed) variable separately, e.g., in the form of a histogram (Figure 4.2b, grey bars). It then fits a Gaussian density function to the histogram bins (Figure 4.2b, pink lines) and fills up the gap between present and fitted distribution with generated data points (golden in both Figures 4.2a and 4.2b).

Although fitting only one Gaussian to the observed data is a strong assumption that might not hold true for all datasets we might possibly encounter, it is a valid starting point due to the following reasons. First, Bareinboim *et al.* [9] prove theoretically that the true class label distribution cannot be recovered from the biased dataset alone without utilizing additional data or assumptions, so some assumption is necessary. Second, following the central limit theorem, numerical real-world observations frequently are approximately Gaussian which makes normal distributions very common [94].

However, we can assume that not all distributions we might encounter are normally distributed. To avoid misleading results, in this case, we need to test if a Gaussian fits the data ‘reasonably well’ and refuse any further outputs if not. Since the observed dataset is potentially biased, skewing its distribution, the acceptable margin necessarily needs to be sufficiently large. Hence, if the true data distribution is similar to (but not exactly) a Gaussian, this distinction will likely not be detected. But since we can expect smoothing over the data distribution to improve the data quality regardless, the implications of assuming a Gaussian distribution are overall benevolent.

Algorithm 1 gives an overview of the main components of the algorithm. IMITATE takes a (biased) labeled dataset as input and separates the classes (Line 4). The subset X' is then transformed into another coordinate system where underrepresented parts of the distribution might be more clearly visible. For each dimension separately, the data density is represented by, e.g., a histogram that we use to fit a distribution density (Line 8). The gap between this estimated (unbiased) distribution $\hat{\mathcal{D}}$ and the present distribution is the area that we identify as $\hat{I}$. We then generate random data points in this area such that B including these points is distributed according to that estimated distribution $\hat{\mathcal{D}}$ (Line 11). The algorithm then estimates its confidence in the produced set of points and decides whether to keep or discard them (Line 14).

Note that we can choose all necessary parameters for IMITATE by either using a domain expert or by selecting the parameter set with the highest confidence. If we use confidence, we cannot guarantee that the algorithm outputs the best possible solution but the one in which it is most confident. The remainder of this section discusses the different elements of Algorithm 1 in detail.

Algorithm 1 IMITATE: Simplified main algorithm

```
Input: A (biased) labeled dataset $B = (X, y) \subset \mathbb{R}^n \times L$
Output: A set of added labeled datapoints $\hat{I} = (\hat{X}, \hat{y})$
function IMITATE($X$, $y$)
    $\hat{X}, \hat{y} \leftarrow [\ ]$
    for all classes $c \in L$ do
        $X' \leftarrow \{x_i \in X \mid y_i = c\}$                    ▷ split classes
        ▷ Remove outliers, transform coordinate system
        $X' \leftarrow$ TRANSFORM($X'$)
        for all dimensions $d \in \{1, \ldots, n\}$ do
            ▷ Represent density over a grid, e.g., by KDE
            $R_d \leftarrow$ REPRESENTDENSITY($X'_d$)
            $F_d \leftarrow$ FITDENSITY($R_d$)                         ▷ Fit a Gaussian
            $G_d \leftarrow F_d - R_d$                                 ▷ gap fitted vs. true data
        end for
        ▷ Generate points according to the gap distribution
        $\hat{X}'_c \leftarrow$ FILLGAP($G_1, \ldots, G_n$)
        ▷ Transform back to the original coordinate system
        $\hat{X}_c \leftarrow$ TRANSFORMBACK($\hat{X}'_c$)
        ▷ Estimate overall confidence in the result, remove points with low individual confidence
        $\hat{X}_c$, conf$_c \leftarrow$ REMOVELOWCONFIDENCE($\hat{X}_c$)
        ▷ Store remaining generated points
        $\hat{X}$.APPEND($\hat{X}_c$)
        $\hat{y}$.APPEND($[c] * |\hat{X}_c|$)                         ▷ add $c$ until $|\hat{y}| = |\hat{X}|$
    end for
    conf $\leftarrow$ [conf$_1, \ldots,$ conf$_{|L|}$]
    return $\hat{X}$, $\hat{y}$, conf
end function
```

4.3.1 Transformation

Given the input data belonging to one class, the feature space might need to be transformed into another space that gives a better view of the present probability densities. In order to do so, we first use the local outlier factor technique [21] to remove outliers as a preprocessing step. This technique is beneficial since it is neighborhood-based and hence supports arbitrary dataset shapes (e.g., banana-like shapes) and does not require parameter choices.

In the examples provided in Figure 4.2, there is a bias based on a selection variable that only relies on one of the features. This is easy to detect and does not need to be transformed in advance. However, if the selection variable depends on several features, bias in the densities over the axes will be less visible. Hence, we transform the data in another coordinate system in which the densities are aligned in a way that reveals missing areas (Figure 4.2, gold).

Independent component analysis (ICA) [70] aims to reconstruct individual signals if only a weighted sum of them is known. It searches for a transformation to a new space described by independent components separating the individual signals. According to the *central limit theorem* [66], those components that show the least similarities of the data density to a Gaussian are most likely individual signals, whereas the Gaussian-like components indicate a mix of signals. Based on that observation, ICA searches for the independent components that show the least Gaussian density for a dataset.

Since we are interested in the components with the most visible deviation from the normal distribution, we use ICA as a heuristic and transform the dataset into the space defined by the components. Note that if all dimensions are normally distributed, ICA will not be able to find a solution. That helps us prevent Imitate from artificially generating a bias that was not present beforehand.

4.3.2 Density Representations

The previous step outputs the data transformed into a new coordinate system. If we want to search for missing areas in the data distribution, we first need to find a discrete representation of the density that we have in the dataset, so we focus on one coordinate axis at a time. The most straightforward choice is a histogram. Histograms work well in many cases because they show a very clear drop of neighboring bin heights if there is a clear border between the biased and the present zone (see Figure 4.2b top). However, unless the dataset is very large, they are sensitive to the choice of bin sizes and positions in the sense that the resulting histogram changes a lot if these parameters are slightly altered.

For smaller datasets (or large datasets with classes that contain only small amounts of examples) IMITATE uses *kernel density estimators (KDE)* with Gaussian kernels to represent the density. The estimator is then evaluated over a 1-dimensional equidistant grid, which results in a similar representation to the histogram bins, but it is smoother.

We let the user choose the density representation. As a rule of thumb, we experienced in our experiments that classes of up to 5000 instances are well represented by a KDE, whereas for larger sets, the difference diminishes, and the histogram representation is much faster. Either way, this step returns a 1-dimensional grid together with evaluations that are considered representative of the entire grid cell. Note that the granularity of the grid, i.e., the number of bins/grid cells, can be considered a user-defined constant or be chosen according to the highest confidence value.

4.3.3 Distribution Fitting

Based on a discrete representation of the data probability density, it should be possible to identify locations where data might be missing due to a bias. Therefore, we fit a distribution to that representation such that the density reveals these locations.

Many observations in real-life applications can be at least approximated by a normal distribution [94] since we can assume that several independent factors contribute to the output due to the central limit theorem. This is the main inspiration for IMITATE. Assuming that the original dataset D is normally distributed, the algorithm tries to fit a normal probability density function to the observed dataset B.

Given the density representatives $r_1, \dots, r_{\#\text{bins}}$ of the dataset over a 1-dimensional grid with cell centers $g_1, \dots, g_{\#\text{bins}}$ (the output of the density representation step; Section 4.3.2), we are aiming to fit a Gaussian to the estimates. A Gaussian is generally defined as a function of the form $g(x) = a \exp\left(-(x-b)^2/2c^2\right)$ for $a, b, c \in \mathbb{R}$ and $c \neq 0$. In the case of $a = 1/(\sigma\sqrt{2\pi})$, $b = \mu$ and $c^2 = \sigma^2$ the Gaussian equals the probability density of a normally distributed random variable with mean μ and variance σ^2.

IMITATE initializes the parameters such that the highest bin fits the peak of the Gaussian and then uses uses the position of the highest bin as an initial estimate for the mean μ, calculates the variance σ^2 of the present data based on that mean, and adjusts the scaling factor a to match the highest bin value. Starting with these initial values, we use a weighted *least squares optimizer* [99] to solve the optimization problem

$$\min_{a,b,c} \sum_{i=1}^{\#\text{bins}} w_i \left(g(g_i; a, b, c) - r_i\right)^2$$

with the weights $w_i = r_i^2$.

The weights are designed in a way that the optimizer puts more emphasis on high bins and is granted more freedom on the areas where very little or no data is present, i.e., the areas where we suspect the data is missing due to bias. We omit further details on the choice of weights here but provide them in our repository. Looking at the example presented in Figure 4.2b (right), equal weights would result in a probability density that is more shifted to the left in order to capture the bin heights below 1 correctly. Although that would be the closest fit to the present dataset, it does not indicate the potentially biased regions. Once the optimal parameters for $\hat{a}, \hat{b}, \hat{c}$ are estimated, we evaluate $g(g_i; \hat{a}, \hat{b}, \hat{c})$ over the same grid and return the fitted values $f_1, \dots, f_{\#\text{bins}}$.

In many cases, it is not possible to find one well-fitted Gaussian, e.g., if the dataset consists of two clusters. If the gap between the fitted and the present distribution becomes very large, we assume that the result is not reliable. To overcome this, we constrain $|\hat{I}| \leq \eta \cdot |B|$ for a user-defined η. In our experiments, we used $\eta = 1$, which seems to be a reasonable choice. If no solution can be found, this step returns $f_1, \dots, f_{\#\text{bins}} = r_1, \dots, r_{\#\text{bins}}$, the input values.

4.3.4 Generation

Having transformed the data density into the representation $R = r_1, \dots, r_{\#\text{bins}}$ and the fitted Gaussian evaluated over the same grid, $F = f_1, \dots, f_{\#\text{bins}}$ (the outcomes of the steps described in Sections 4.3.2 and 4.3.3, respectively), Imitate next generates points that can lift the present density to the fitted one when added to the training set. These r_i and f_i have

Algorithm 2 IMITATE: Fill in the Gap

Input: Gap vectors G_d and fitted density vectors F_d over 1D-grids with cell centers $g_1^d, \ldots, g_{\#\text{bins}}^d$ for each dimension $d \in \{1, \ldots, n\}$ restricted to one class c

Output: A set $\hat{X}_c'$ of generated data points for this class

```
 1: function fillGap($G_1, \ldots, G_n$)
        ▷ Determine the number of points to be generated
 2:     $\#\text{points} \leftarrow \max_d \|G_d\|_1$
 3:     for all dimensions $d \in \{1, \ldots, n\}$ do
 4:         $\#\text{points}_d \leftarrow \|G_d\|_1$                ▷ Points for this dimension
            ▷ Convert to cumulative density function
 5:         $G_d', F_d' \leftarrow \text{convertToCDF}(G_d, F_d)$
            ▷ Get mixed CDF for this dimension
 6:         $p_d^G \leftarrow \#\text{points}_d / \#\text{points}$
 7:         $\text{CDF}_d \leftarrow p_d^G \cdot G_d' + (1 - p_d^G) \cdot F_d'$
            ▷ Draw #points coordinates according to $\text{CDF}_d$
 8:         $x_d \leftarrow \text{random}(\text{CDF}_d, \#\text{points})$
 9:     end for
        ▷ Combine single coordinate vectors and return
10:     return $\hat{X}_c' = \left[x_1^T, \ldots, x_d^T\right]$
11: end function
```

been calculated for each dimension and class separately and help decide in which grid cells data needs to be generated. Figure 4.2b marks these cells in gold. The size of the golden bins indicates the number of points that are required to match the fitted distribution (pink line).

Algorithm 2 presents an overview of this step. For each dimension d and class c, we measure the gap between fitted and present density in order to determine how many points need to be added until this dimension's fitted density is achieved, i.e., $\#\text{points}_d = \sum_i \max\{f_i - r_i, 0\} = \|G_d\|_1$ where $\|\cdot\|_1$ denotes the $\ell 1$-norm (Line 2). Note that we use the maximum to focus only on the bins that need to be filled up. It allows us to ignore the case when $f_i < r_i$ for any i in which points from the input dataset would have to be removed in order to match the fitted density. The dimension with the highest gap determines how many points will be added in total, i.e., $\#\text{points} = \max_d \#\text{points}_d$.

Each dimension contributes its own coordinates to the final points. The grid cells are drawn in a way that the fitted density for this dimension is achieved. At first, the gap G_d is filled. If more coordinates need to be generated (i.e., because $\#\text{points}_d < \#\text{points}$), the remaining ones are drawn according to the fitted density F_d. To achieve that result, we obtain the mixed cumulative density function (*CDF*, Line 7) as

$$\text{CDF}_d = p_d^G \cdot \text{CDF}(G_d) + \left(1 - p_d^G\right) \cdot \text{CDF}(F_d)$$

for $p_d^G = \#\text{points}_d / \#\text{points}$ and use it to draw the cells in which the coordinates will then be drawn uniformly. That yields a vector x_d containing #points coordinates for dimension d such that their addition to the original points B fulfills the fitted probability density over the grid (Line 10).

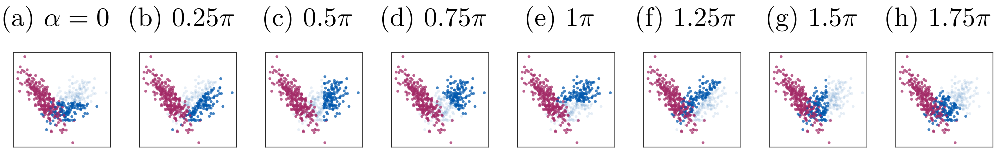
Once the coordinate generation is performed for each dimension, the results are combined into a set $\hat{X}'_c = \left[x_1^T, \ldots, x_d^T\right]$ of points for class c where $\cdot^T$ denotes the transposition of a vector. The result of this step can be seen in Figure 4.2a: The golden points are the ones that IMITATE generated.

Note that the coordinate-wise generation of points is only able to model convex shapes. If the dataset has the shape of a ring (and IMITATE is not supposed to fill in the hole), another method needs to be found. This drawback will be addressed in future research (see Section 4.5).

4.3.5 Confidence

Having generated a set of points, we want to know if the results are good at all or should be discarded.

As the output of the algorithm, we expect clearly identified zones that are densely filled with generated points. If the points are spread over a wide area and rather singletons than clusters, that most likely only reflects noise in the data or a bad choice of the underlying grid (Section 4.3.2). We hence use a heuristic for the confidence of IMITATE in its output that compares the spread of the generated points to the spread of the dataset, i.e., for each generated point p we measure $d_{10\text{NN}}(p)$, the average distance to the 10 nearest neighbors. As a baseline for comparison, we use a random subset D' of the input dataset D with $|D'| = |\hat{I}|$, and average over the same calculated distance yielding $\mu_{10\text{NN}}(D')$ and the standard deviation $\sigma_{10\text{NN}}(D')$ from that mean. Other methods of comparison (such as the comparison to the spread



(a) $\alpha = 0$ (b) 0.25π (c) 0.5π (d) 0.75π (e) 1π (f) 1.25π (g) 1.5π (h) 1.75π

Figure 4.3: We created synthetic biases on a dataset by rotating a horizontal cutting plane by an angle α. The points that are faded out describe I, the bold ones give B.

of the class in the entire set D) have been tested. We refer to our repository for a test of different confidence options.

All points p with $d_{10\text{NN}}(p) \geq \mu_{10\text{NN}}(D') + \sigma_{10\text{NN}}(D')$ are discarded right away. The confidence score is then defined as the average inverted 10-NN density, $\text{conf} = 1 \big/ \bar{d}_{10\text{NN}}(p)$.

A result will be discarded entirely if 10 or fewer points have been generated since the 10-NN-based distance measurement is not valid anymore, or if no points are left after the individual checks. If several parameter settings (e.g., grid granularities) have been tested, we use the result with the highest confidence.

4.4 Experiments and Discussion

To carry out actual real-world experiments, we would need to find already biased datasets together with their ground truth. However, they are hard to find (which is why we need methods like Imitate, after all!). Therefore, we study the behavior of the proposed method mainly on synthetic datasets where we can isolate the effects that we want to investigate and then test Imitate on real-world datasets with an artificially created bias. In the end, we give an example of a real-world scenario and discuss remaining limitations.

4.4.1 Synthetic Data

For the experiments, we generated synthetic 2D datasets D with two classes (blue and wine) and, unless advised otherwise, 10000 samples and 5% label noise. The bias was created by a plane rotating around the center of one of the classes: p (default: $p = 0.05$) denotes the proportion of data points above that plane that remained in B; the rest was removed. The other class and all points below the plane are contained in B. Figure 4.3 shows the datasets with biases as described in the blue class where the planes were rotated. The rotation angles α are given in the captions. For the grid granularity, $\#\text{bins} \in \{5, 8, 11, \ldots, 29\}$ have been tested in each example, and we use the result with the highest confidence. We repeat each experiment 10 times to make up for randomness and generate 10 datasets for each parameter setting.

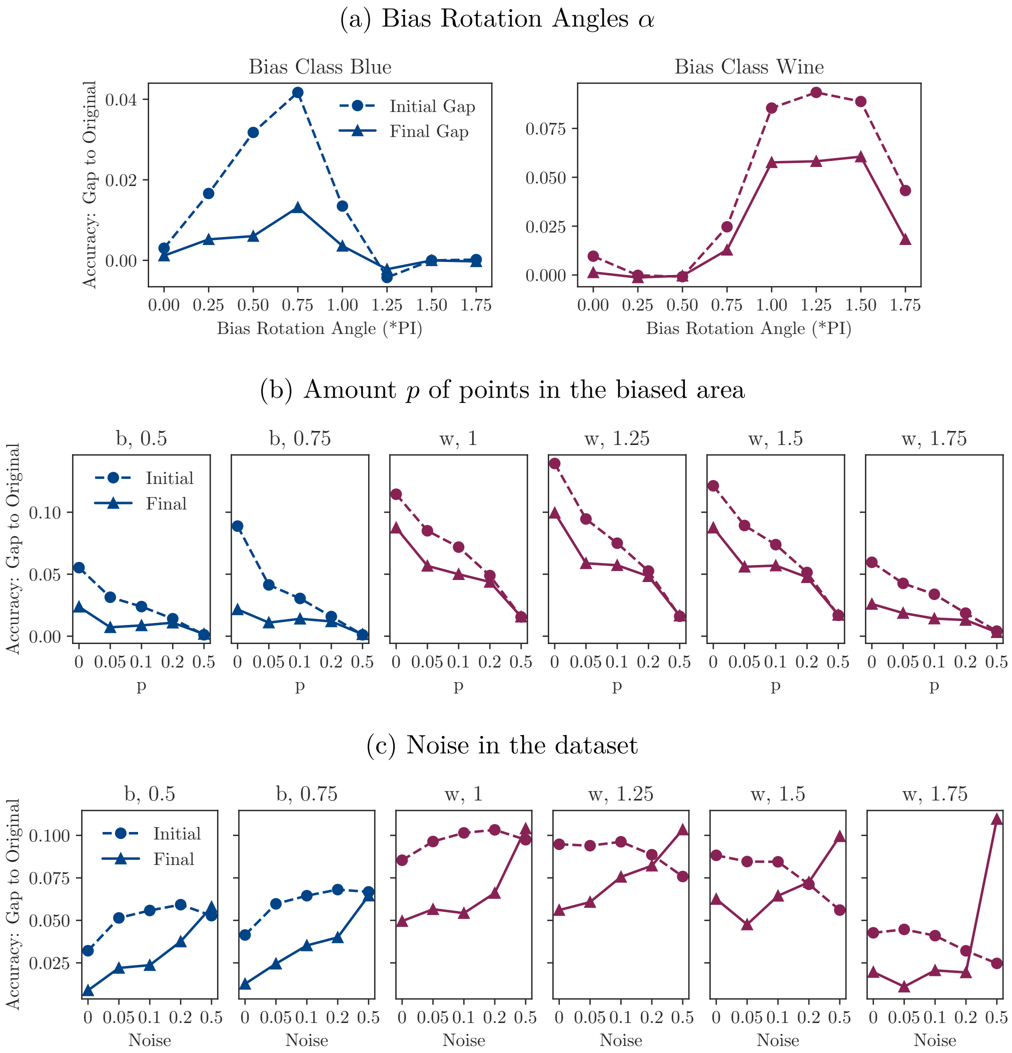
We alter all three parameters (α, p, and the dataset noise) and investigate the improvement in accuracy of a linear *support vector machine (SVM)* trained on $B \cup \hat{I}$ vs. B alone. report the improvement in accuracy in Figure 4.4. As a baseline, we train a linear SVM on the unbiased original dataset D and denote its accuracy on an unbiased test set as acc_D. Similarly, we train linear SVMs on B and on $B \cup \hat{I}$ and denote their performance on the same unbiased test set as acc_B and $\text{acc}_{B \cup \hat{I}}$, respectively. The dashed line shows the initial accuracy gap $\text{acc}_D - \text{acc}_B$, the solid line shows the final accuracy gap $\text{acc}_D - \text{acc}_{B \cup \hat{I}}$ after application of IMITATE. The smaller the value of the solid line, the better IMITATE performed the point generation.

Rotation α

As can be seen from the dashed line in Figure 4.4a, the bias does not always affect the performance of the classifier. Nevertheless, if it does, IMITATE is able to find a set $\hat{I}$ that gives a substantial improvement of $B \cup \hat{I}$ over B alone. In the case of an $\alpha = 1.25\pi$ rotation bias in the blue class, we even see an improvement in the classifier performance if parts of the data are removed. The cases with high initial gaps are considered interesting cases since their bias mitigation is the most necessary. We use these cases for the following experiments.

Amount p of points in the biased area

Figure 4.4b shows that for various amounts $p \in [0, 0.5]$ of points that remain in the biased area after application of the rotation bias, IMITATE can achieve a substantial improvement in performance. Note that if p is high enough,



(a) Bias Rotation Angles α

(b) Amount p of points in the biased area

(c) Noise in the dataset

Figure 4.4: Experiments on synthetic data using synthetic rotation biases. The plots (a), (b), and (c) explore the behavior of IMITATE for different rotation angles (α), amounts of remaining points in the biased area (p), and noise in the dataset, respectively. Reported is the gap in accuracy between a classifier trained on the original, unbiased dataset D and one that is trained on B only (dashed line) and on $B \cup \hat{I}$ (solid line). If it hits 0, we managed to reconstruct the performance of the unbiased classifier fully. For (b) and (c), the type of synthetic bias is given by the title $(c, \alpha/\pi)$, which means a rotation bias in class $c \in \{\text{blue (b)}, \text{wine (w)}\}$ with angle α.

Dataset	**Predicted Attribute**	**Biased Set B**
Abalone	Sex	Viscera weight < 0.144
Banknote*	Class	Variance > 0.32
Car**	Class	Persons > 3
Statlog (Shuttle)	Class==Rad Flow	Time > 54.5
Skin* [13]	Class	R ≤ 170.5

Table 4.1: Dataset overview. *Due to the small dimensionality of the dataset, we did not omit the attribute used for the bias split. **The categorical variables were transformed into integers as preprocessing.

our technique to create a bias is not sufficient anymore to cause a biased classifier.

Label Noise in the dataset

The higher the label noise in the dataset, the less the synthetic bias affects the results. We can see that trend in Figure 4.4c. It is also clearly visible that Imitate helps more when there is only a small amount of noise present and can even decrease the performance of a classifier for noisy data. That is the expected result, as too much noise overshadows the true probability density and leads to an incorrectly fitted density.

4.4.2 Real-World Data

For the evaluation of our method on real-world data, we used classification datasets from the *UCI machine learning repository*[1][41] and created a bias as follows: To make sure that our split is relevant for the classification, a decision stump was trained on each dataset, and we used the same split for the bias and removed the corresponding attribute afterward. B is then the data in the larger leaf, and I is the smaller one. If that procedure had removed a class (almost) entirely, we used a decision stump on the data without the original attribute. This procedure is consistent with that presented in Zadrozny [158]. Table 4.1 summarizes the datasets together with the criteria for the bias.

[1] Thanks to NASA for allowing us to use the Shuttle dataset.

Dataset	**Baseline** **Classifier**	**Initial Gap** $\mathrm{acc}_D - \mathrm{acc}_B$	**Final Gap** $\mathrm{acc}_D - \mathrm{acc}_{B \cup \hat{I}}$
Abalone	SVM (linear kernel)	0.151	0.089*
	SVM (RBF kernel)	0.151	0.107*
	Decision Tree	0.154	0.103*
Banknote	SVM (linear kernel)	0.218	0.176*
	SVM (RBF kernel)	0.205	0.162*
	Decision Tree	0.196	0.156*
Car	SVM (linear kernel)	0.133	0.144
	SVM (RBF kernel)	0.131	0.138
	Decision Tree	0.146	0.152
Shuttle	SVM (linear kernel)	0.607	0.466*
	SVM (RBF kernel)	0.590	0.424*
	Decision Tree	0.566	0.482
Skin	SVM (linear kernel)	0.003	-0.010*
	SVM (RBF kernel)	0.002	-0.010*
	Decision Tree	0.005	-0.009*

Table 4.2: We show the initial accuracy gap between a baseline classifier trained on the ground-truth dataset D and one trained on the biased set B and compare it to the final gap (after application of Imitate). * indicates that the result is statistically significantly better than the other (t-test at a significance level of 1%).

We measure the performance of Imitate by the same accuracy gaps we used for synthetic data. Different baseline classifiers were used, i.e., SVMs with a linear and an RBF kernel, as well as a decision tree. In order to obtain more reliable results, we repeated each experiment 10 times and report the average result, which is displayed in Table 4.2. The * indicates that one result was significantly better than the other based on a t-test at a significance level of 1%.

For the Abalone, Banknote, Shuttle, and Skin datasets, Imitate could significantly improve the biased dataset in order to obtain better classification results. The Skin dataset allowed us to improve over the original result. Although that is a desirable result, it implies that Imitate did not reconstruct the original data but generated new data. The fact that it could improve the performance indicates that there might very well be a bias on the original dataset! On the Car dataset, our technique does not improve

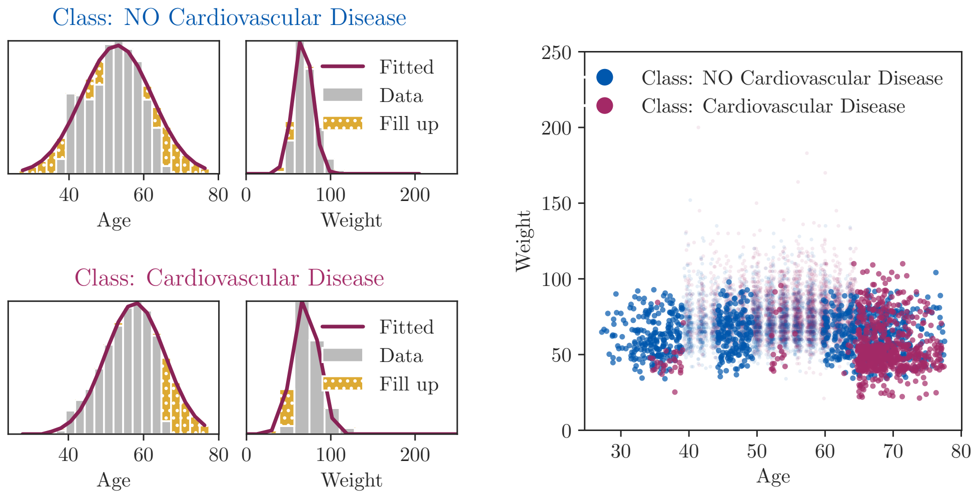


Figure 4.5: We apply Imitate to Kaggle's Cardiovascular Disease dataset, restricted to the age and weight features. The two left plots show the data densities per axis for both classes (healthy and sick), respectively. The right plot demonstrates the final result ($\hat{I}$ is bold, B is faded out).

the data quality. That is due to the fact that the Car dataset is discrete and hence very sensitive to the choice of the underlying grid (see Section 4.3.2). Imitate mainly fills in the gaps between the true values (with high confidence) instead of focusing on other underrepresented areas. Discrete datasets are a weakness of Imitate that we need to overcome, e.g., by adding noise or dimensionality-reduction, as discussed in Section 4.5.

4.4.3 Use-Case: Cardiovascular Disease

In order to show how Imitate can be applied in the data gathering process, we use Kaggle's Cardiovascular Disease dataset[2]. The dataset contains 70000 medical examination measurements together with factual information on the patient and the target variable states the presence or absence of cardiovascular disease. Since we want to be able to visualize it easily, we restrict the dataset to the age and the weight of a patient as well as the class label and apply Imitate to it. Figure 4.5 shows the result: the original data B is faded out, and the generated data points $\hat{I}$ are the bold ones.

[2] Source: kaggle.com/sulianova/cardiovascular-disease-dataset

The result clearly shows three trends: (i) There is a large amount of data missing for patients over 65 years, especially for ones suffering from cardiovascular disease. It is well known that the risk of cardiovascular disease increases with the age of the patients, but the dataset shows a hard border here. Hence we consider this a reasonable result. (ii) Similarly, patients below the age of 40 are clearly under-represented, particularly healthy ones. This might be because younger people are at lower risk and hence do not need to screen for cardiovascular disease regularly unless there is a concrete reason. (iii) Especially for the healthy patients but also for the other ones, we see a generated area below a weight of 60kg. This result could have several reasons that a domain expert should carefully assess, e.g., it might mirror the observation that there are more over- than underweight people in the dataset. That is possibly due to the fact that being overweight increases the risk of cardiovascular disease, which is why overweight people are more likely to go for a corresponding examination, but maybe the bias here also reflects a trend in the underlying weight distribution of the entire population in the data collection area.

If we want to validate these results, we could either consult a domain expert or use additional datasets to check the distributions of the underlying population (i.e., the potential patients of the location where the data was collected) in terms of weight or age and compare it to what we found. These underlying distributions can then be exploited for the typical weighting approaches or other covariate shift methods to train a classifier that is reliable for the entire population, not only for the patients of this specific location.

4.4.4 Limitations

We saw that Imitate performs well in many cases and helps us identify potential biases. However, we need to be careful with either discrete datasets or datasets with several clusters per class for the same reason: Imitate fills in the gap between the clusters and thereby overshadows potential biases inside the clusters.

Figure 4.6 shows an extreme case. If the two clusters were closer together, the effect would be less harmful. If they were much further apart from each

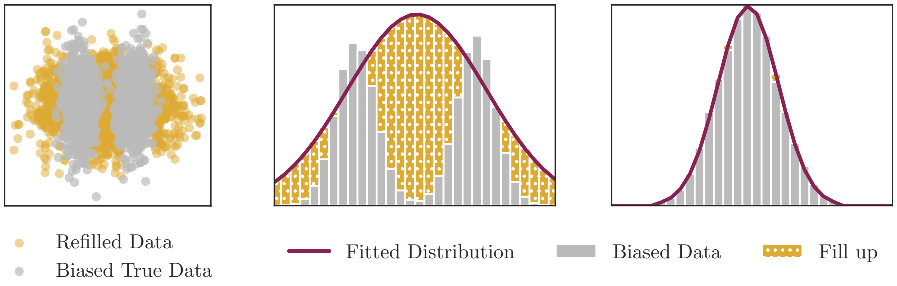


Figure 4.6: Drawbacks of Imitate: We applied our method to a synthetic dataset consisting of two clusters (grey). Imitate fills in the gap between the clusters (gold).

other, the restriction that we allow Imitate to at most double the points (see Section 4.3.3) kicks in and results in no correction for this particular dimension at all. That means that biases in this dimension are not analyzed, but at least the result is not harmful.

A discrete dataset basically yields the same problem, but it can be mitigated by the right choice of the underlying grid, as the discreteness can be smoothed out if the grid is coarse enough and well positioned.

Another drawback of Imitate is hard domain-dependant boundaries, e.g., in the example explained at the beginning of Section 4.3 where we wanted to "complete" a dataset of flower measurements in an area with a lake. The lake here is a hard boundary, and no reconstruction over the lake area should be made. In another identical dataset, the gap could have occurred due to a restricted area where no measurements could be taken, but there are flowers. We cannot expect Imitate to distinguish between those identical datasets automatically, but we should allow for user-given hard boundaries in the density fitting process.

We will address all of these problems in future research. See the following section for a short discussion of strategies to solve or at least mitigate them.

4.5 Conclusion

In this chapter, we introduced Imitate, a simple, modular, and extendable approach to identify and mitigate selection bias in the case that we may not know if (and where) we have a bias, and hence no ground-truth information is available. In contrast to comparable methods that consider the data as given and exploit background information to learn a back-shifted classifier, Imitate can be used in the data-gathering process to identify a potential bias right away.

Experiments showed that Imitate can yield meaningful results and can support the data collection process by pointing out potential biases in an early stage, but it is also capable of bias mitigation in a later stage of the data mining process. We discovered that fitting one Gaussian per (transformed) feature often helps but does not always lead to success. The major problems we identify are the following:

Discrete Datasets. If the dataset is discrete and not continuous, Imitate's performance relies heavily on the choice of the underlying grid for the density estimation and fitting. An equidistant grid is hard to adjust and might not always be suitable. Extensions of Imitate hence should include an automated way of finding a well-suited grid for each dimension individually. Other options to smooth out the discreteness here could be adding noise or dimensionality reduction. See Section 5.2.4 for an automated way to choose the grid, and Chapter 6 for an adaptation to discrete data.

Clusters. If the dataset consists of several clusters per class, in the dimensions that separate the clusters, Imitate will either fill in the gap between the clusters or not do anything at all if the clusters are too far away from one another. The second case is fine as long as there are dimensions where the clusters overlap; the first case is a problem. Improvements to Imitate should take that into account and either apply a pre-clustering and treat each cluster separately or fit a mixture model of several Gaussians depending on the number of clusters that show in a particular dimension. The following chapter tackles this challenge.

Hard Boundary. We cannot determine if a dataset has a hard boundary somewhere (or if such a boundary is a sign of a bias) as boundaries are domain-related. However, we will extend Imitate in Section 6.4.3 to allow users to set constraints representing these boundaries and consider them during the density fitting. One way would be by adjusting the weights in the optimization accordingly. If the dataset is shaped like a ring with a circular boundary in the middle (e.g., the lake problem in Sections 4.3 and 4.4 – “Limitations”), a re-adjustment of the weights will not be sufficient. In this case, a solution might be to transform the dataset into a higher-dimensional space that can separate both areas by a plane.

Overall, Imitate is the first method to identify and mitigate selection bias when no ground truth or additional knowledge is required. We see in Imitate a promising start of a new direction of research that is modular enough to allow for extensions and improvements.

5 Multi-Cluster Selection Bias Identification and Mitigation

The research presented in this chapter has been adapted from

> K. Dost, H. Duncanson, I. Ziogas, P. Riddle, and J. Wicker, "Divide and imitate: Multi-cluster identification and mitigation of selection bias," in *Advances in Knowledge Discovery and Data Mining - 26th Pacific-Asia Conference (PAKDD '22)*, ser. Lecture Notes in Computer Science, vol. 13281, Springer Cham, 2022, pp. 149–160. DOI: 10.1007/978-3-031-05936-0_12.

The results of this chapter are available in the GitHub repository github.com/KatDost/Mimic and the proposed algorithm is contained in the PyPI package imitatebias.

5.1 Introduction

In order to identify and mitigate selection biases where no additional information is available, in the previous chapter, we proposed Imitate, a technique that, given a biased dataset, aims to estimate the ground-truth distribution and generate data points to augment the dataset accordingly. While we demonstrated Imitate's ability to improve model performance through pre-augmentation on several examples, it is limited by a major assumption: the underlying ground truth is expected to be normally distributed. In practice, this strongly limits the applicability of Imitate as it

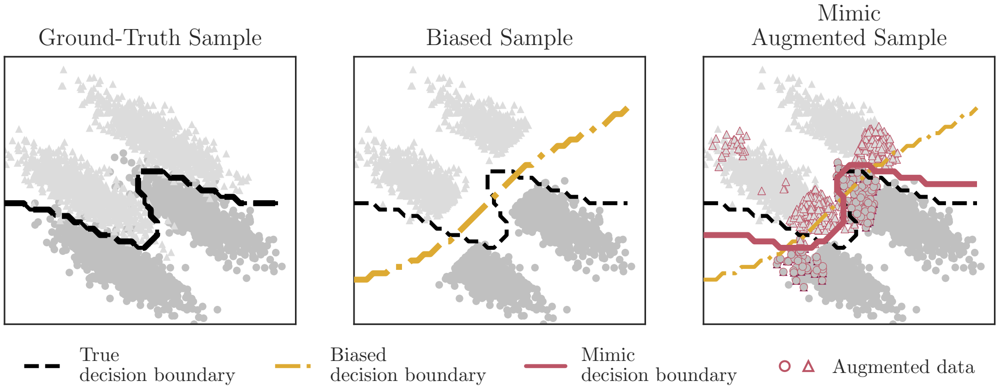


Figure 5.1: Decision boundaries of support vector machines trained on three different datasets: a sample representative for the ground truth (left), a biased subset (center), and the biased subset augmented with our algorithm, MIMIC (right).

is neither flexible enough to model non-Gaussian distributions nor can it capture datasets consisting of several clusters.

In this chapter, we introduce MIMIC (*Multi-IMItate Bias Correction*), a multi-cluster solution for the identification and mitigation of selection biases that exploits IMITATE as a building block. Modeling data as a mixture of possibly biased and overlapping multivariate Gaussians MIMIC overcomes IMITATE's limitations and greatly increases its applicability. The parameters of these Gaussians bridge between the estimated and the present distribution and can indicate underrepresented regions in the data that are likely to correspond to a selection bias. Generating points in these regions helps mitigate the effect of the bias and push the decision boundary towards the ground truth (see Figure 5.1).

Although attempting to solve the same problem as stated before in Section 4.2, MIMIC is substantially different from IMITATE in its approach. Here, we relax IMITATE's requirement of normal distributions and assume each class of D consists of a mixture of Gaussians. In other words, we assume that each class of the dataset can be represented by a set of possibly overlapping Gaussian clusters. Since Gaussian mixtures are very flexible in the distributions they can model, particularly when the number of clusters is not limited, we could expect them to fit a biased dataset reasonably well without ever pointing out biases. Hence, modeling the ground truth with a mixture based on only the biased subset requires us, as a first step, to

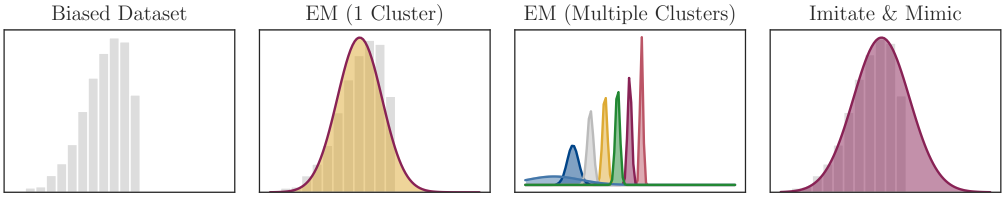


Figure 5.2: When facing a biased sample (1st plot from left), the EM algorithm will fit one (2nd) or multiple (3rd; here controlled by BIC) Gaussians to minimize the error on the presented data. Imitate and Mimic (4th) instead use the histogram bin heights as weights for the fitting procedure and capture the underlying ground truth more closely.

determine the number of clusters in the biased dataset and to group the data points accordingly. This makes the problem particularly challenging to solve.

The remainder of this chapter is organized as follows: We refer back to Section 4.2 for the problem statement and to Chapter 3 for related research fields and skip both here. Our proposed method, Mimic, is introduced in Section 5.2, and we discuss the implicit assumptions and expectations the algorithm makes at the end of this section. In a set of experiments in Section 5.3, we demonstrate the shortcomings of existing techniques and highlight the potential of Mimic in these scenarios. Section 5.4 concludes the chapter with a discussion.

5.2 Proposed Method

Aiming to provide a bias mitigation strategy for a wide range of problems, in this chapter, we assume that ground-truth data consists of a mixture of multivariate Gaussians. Although this is still a limiting assumption, it substantially widens the range of datasets that can be modeled when compared to the Imitate algorithm. Before analyzing each Gaussian for potential biases, we need to find a suitable mixture model for the ground truth based solely on the biased dataset.

If no bias is present in the dataset, *Gaussian mixture models (GMMs)* [35] can fulfill the task as they are able to identify the optimal Gaussians to describe a presented dataset given suitable initial cluster centers. These centers (and the number of clusters) could be found using, for example, the

Bayesian information criterion (BIC) [57]. In the case of a selection bias, however, one biased cluster might be split into several Gaussian clusters as that mixture fits the presented dataset better, as shown in Figure 5.2. Assume a clinical study testing the impact of a new drug on test and control groups. While GMM breaks the group of participants into many small clusters as it models the presented datasets, we need to find clusters that give an indication of where some data might be missing and thereby indicate that, e.g., women below a certain age did not participate due to safety concerns. Therefore, we need to develop a novel strategy to cluster biased datasets into separate potentially overlapping Gaussians that capture the ground truth rather than the biased presented data.

The central idea for MIMIC is simple as illustrated in Figure 5.3: We start with a large number of clusters and let IMITATE indicate where data might be missing. In contrast to agglomerative clustering, [61] which iteratively merges the closest clusters, we operate on a point basis. If data is available in another cluster to fill in the gap, we let the cluster grow by assigning these data points until it is approximately normally distributed or no suitable data points can be found. In this case, we found a potential selection bias and generate data points to mitigate it. Once all initial clusters have been fully grown, a merging procedure purges duplicates and combines suitable clusters to overcome locally optimal solutions. This process is carried out for every class of the initial dataset (if any) separately, but we describe it for only one class in the following in order to simplify. See Algorithm 3 for an overview and the following for a detailed discussion of the components.

5.2.1 Initialization [Alg. 3; Lines 2-3]

Starting with only the biased dataset B, the initialization step divides it into a large number of initial clusters that MIMIC uses to search each of them for non-normality. It then uses this information to "steal" data points from other clusters into this one and grow it. If the initial clusters are already sufficiently normal, no direction for growth can be identified. Therefore, after pre-processing the data with *local outlier factor (LOF)* [21] for higher cluster quality, MIMIC starts off with non-Gaussian initial clusters like those

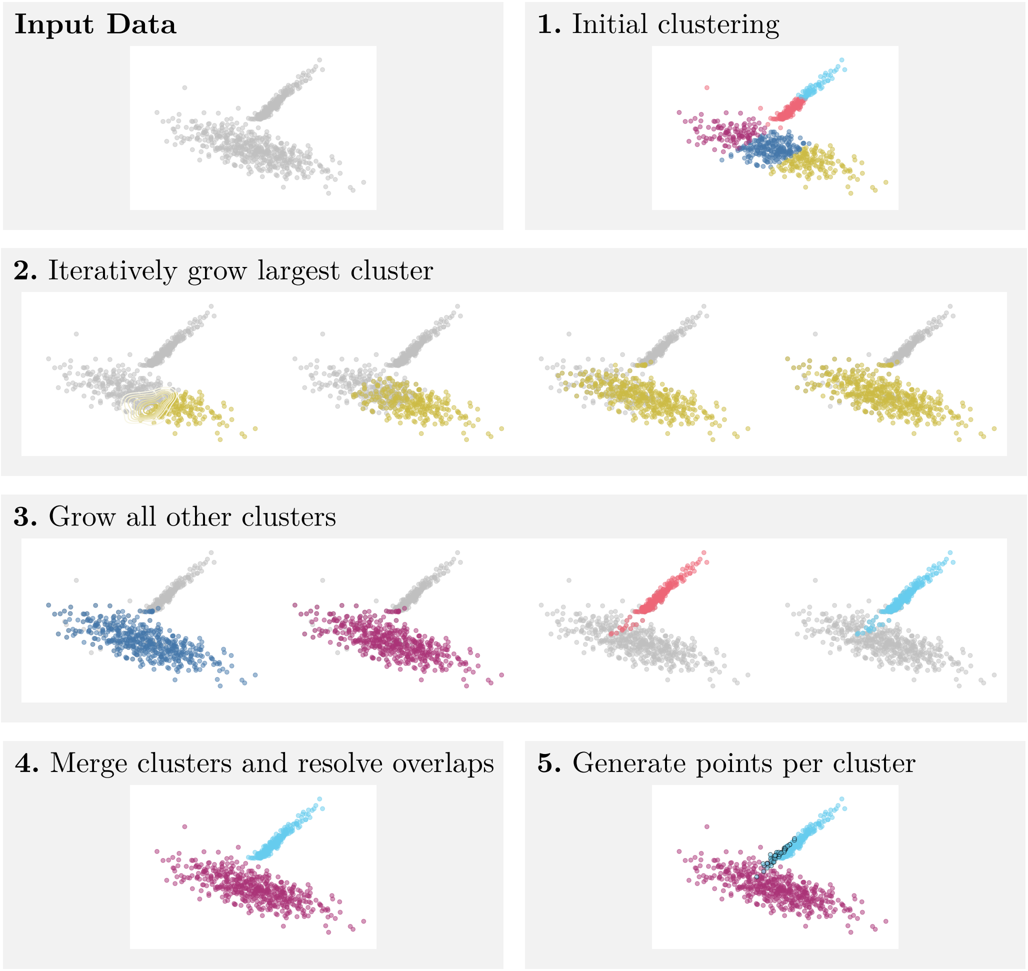


Figure 5.3: Overview over MIMIC: Given a potentially biased dataset, MIMIC clusters it (Step 1), grows the largest cluster first (Step 2), followed by all other clusters (Step 3). Finally, MIMIC merges the grown clusters and resolves overlaps (Step 4) before applying IMITATE to each cluster individually to generate points that mitigate the bias (Step 5).

obtained from *k-means*. k-means brings two major advantages: First, it is fast. Second, it is simple enough to cut overlapping clusters and capture also non-overlapping parts. These parts are essential to enable MIMIC to grow the clusters correctly later on. A high number of initial clusters additionally increases the probability to capture an initial cluster that can later be grown, even if overlaps exist. In order to use a sufficient number of initial clusters, we use twice the number that maximizes the *silhouette score* [57] and split further if we detect two density peaks in a histogram instead of one. The data is pre-processed using LOF in order to improve the quality of the initial clusters and to obtain a measure of density that is later used in deciding on

Algorithm 3 MIMIC: Main algorithm

```
Input: A biased dataset $B$
Output: Parameters $\theta_i = (\mu_i, \Sigma_i)$ for each cluster $i$ and a set $P$ of generated
    points that mitigate the bias
 1: function MIMIC($B$)
        ▷ Remove outliers using LOF (Sec. 5.2.1)
 2:     $B' \leftarrow$ REMOVEOUTLIERS($B$)
        ▷ Initialize clustering using k-means with large $K$ (Sec. 5.2.1)
 3:     $l \leftarrow$ INITIALIZECLUSTERING($B'$)
 4:     $\theta \leftarrow \emptyset$
 5:     $L \leftarrow$ LARGESTVALIDCLUSTER($l$)                    ▷ (Sec. 5.2.2)
        ▷ Grow every valid cluster. A cluster is valid if it is large and dense
        enough and has neither been processed before nor subsumed by a
        previous iteration (Sec. 5.2.5)
 6:     while $L$ exists do
 7:         $l$, $\theta_L \leftarrow$ GROWCLUSTER($L$, $B'$, $l$)
 8:         $\theta \leftarrow \theta \cup \theta_L$
            ▷ Select the largest valid cluster based on the updated labels $l$ (if
            possible)
 9:         $L \leftarrow$ LARGESTVALIDCLUSTER($l$)                ▷ (Sec. 5.2.2)
10:     end while
        ▷ Merge clusters if it improves normality (Sec. 5.2.6)
11:     $\theta \leftarrow$ MERGE($\theta$, $B'$)
        ▷ Generate data to mitigate the bias (Sec. 5.2.7)
12:     $P \leftarrow$ AUGMENT($\theta$, $B$)
13:     return $\theta$, $P$
14: end function
```

cluster validity. We chose LOF since it is parameter-free and identifies local outliers rather than global ones which is particularly important if clusters are not equally dense and spread. From here on, the outlier-free dataset is denoted as B' and is passed on to the next step together with the initial labels l.

5.2.2 Identifying Valid Clusters [Alg. 3; Lines 5, 9]

Once a large number of initial clusters has been found, MIMIC grows them into Gaussian clusters where possible using points from B. Aiming to secure reliable performance during the subsequent fitting of a multivariate normal

Algorithm 4 growCluster (Sec. 5.2.5)

```
Input: Label $L$ to be grown, outlier-free dataset $B'$ with labels $l$
Output: Updated labels $l$, parameters $\theta_L$ for cluster $L$
function growCluster($L$, $B'$, $l$)
    repeat
        $B'_L \leftarrow B'|_{l=L}$                                ▷ Cluster $L$
        ▷ Run Imitate on $L$ to obtain $G_L$ that represents where data might be missing on a grid-basis, the number of missing data points $n_L$ and the parameters $\theta_L$ of the fitted normal distribution (Sec. 5.2.3 and 5.2.4)
        $G_L$, $n_L$, $\theta_L \leftarrow$ Imitate($B'_L$)
        ▷ Score all remaining data points based on if they are likely to help improve the fit of the Gaussian
        $s \leftarrow$ score($B' \setminus B'_L$, $G_L$, $\theta_L$)
        ▷ Identify $n_L$ suitable candidates in batches $b_i$ and sample based on $s$
        for all batches $b_i$ with $\sum_i b_i = n_L$ do
            $C_i \leftarrow$ sample($B' \setminus B'_L$, $b_i$, $s$)
            ▷ Assign a batch of candidates to the cluster if it improves the likelihood of the model fitting the data
            if $\mathbb{P}[\theta_L \mid B'_L \cup C_i] > \mathbb{P}[\theta_L \mid B'_L]$ then
                $l(C_i) \leftarrow L$                     ▷ Update $l$ for accepted $C_i$
            end if
        end for
    until $l$ did not change
    return $l$, $\theta_L$
end function
```

distribution, we filter out all clusters that are either (i) too small (fewer than 10 data points in our implementation) or (ii) too widespread with low density (that is, if the cluster's LOF lies below the 3σ-interval of the average cluster LOF). Note that the latter is a necessary measure, as we can expect to obtain unreliable results when fitting a normal distribution to a set of singletons. Additionally, we reduce the computational burden by ensuring that no cluster is grown more than once and no cluster that has been fully subsumed in previous iterations is processed. Thereby, we reduce the number of duplicate clusters we obtain and focus on the most promising ones. Each iteration selects the largest valid cluster and grows it as described below until no valid clusters remain.

The largest cluster is determined based on a label vector l that is updated in each iteration. If, for example, an initial cluster contained 50 data points, but 45 have been assigned to another cluster, it will be discarded as being too small (Criterion (i)). Similarly, if the center of an initial cluster has been assigned to a different cluster, the leftovers will be too widespread to be processed (Criterion (ii)). This is a necessary measure as we can expect to obtain unreliable results when fitting a normal distribution to a set of singletons.

5.2.3 Extending Imitate: From Grid to Parameterized Gaussians [Alg. 4; Line 4]

Given a cluster L, IMITATE estimates a multivariate Gaussian (see Section 4.3) and indicates based on a grid where (and how many) points need to be generated in order to smooth out the cluster's density and have it resemble the fitted Gaussian. Note that the IMITATE algorithm, as described in the previous chapter, continues to operate on the grid representation, which would result in a high complexity given our repeated IMITATE calls and does not allow for precise probability assignments. Hence we adjust: Assume we fitted one Gaussian (μ_i, σ_i^2) for each of the d independent components i in the ICA-transformed space. In other words, for each of the components, IMITATE provides us with a univariate density

$$f_i(x_i) = \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{x_i - \mu_i}{\sigma_i}\right)^2\right].$$

The joint probability function f for independent densities $f_1, \ldots, f_d$ is the product $f(x) = \prod_i f_i(x_i)$ for a data point $x \in \mathbb{R}^d$ in ICA-space. This term can be transformed to

$$\begin{aligned}
f(x) &= \prod_{i=1}^{d} f_i(x_i) \\
&= \prod_{i=1}^{d} \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{x_i-\mu_i}{\sigma_i}\right)^2\right] \\
&= \frac{1}{(\prod_i \sigma_i)\cdot(\sqrt{2\pi})^d} \exp\left[\sum_i -\frac{1}{2}\left(\frac{x_i-\mu_i}{\sigma_i}\right)^2\right] \\
&= \frac{1}{\sqrt{(\prod_i \sigma_i^2)\cdot(2\pi)^d}} \exp\left[-\frac{1}{2}\sum_i \frac{(x_i-\mu_i)^2}{\sigma_i^2}\right] \\
&= \frac{1}{\sqrt{\det\Sigma\cdot(2\pi)^d}} \cdot \exp\left[-\frac{1}{2}(x-\mu)^{\mathrm{T}} \begin{bmatrix} \frac{1}{\sigma_1^2} & & \\ & \ddots & \\ & & \frac{1}{\sigma_d^2} \end{bmatrix} (x-\mu)\right] \\
&= \frac{1}{\sqrt{\det\Sigma\cdot(2\pi)^d}} \exp\left[-\frac{1}{2}(x-\mu)^{\mathrm{T}}\Sigma^{-1}(x-\mu)\right]
\end{aligned}$$

for $\mu = (\mu_1, \ldots, \mu_d)$ and $\Sigma \in \mathbb{R}^{d\times d}$ with diagonal $(\sigma_1^2, \ldots, \sigma_d^2)$ and 0 elsewhere. This is the density formula of a multivariate Gaussian parameterized by (μ, Σ).

Assume that such a multivariate Gaussian (μ, Σ) has been found in the ICA-space obtained by a transformation $x \mapsto Ix =: x'$ for a data point $x \in \mathbb{R}^d$ in the original space and the ICA matrix $I \in \mathbb{R}^{d\times d}$. To identify biases in the original data space, we need to transform the Gaussian back to the original data space. We can now insert the transformation term into the definitions of mean and covariance matrix and exploit the linearity of the expectation:

$$\begin{aligned}
\mu &= \mathbb{E}[x'] = \mathbb{E}[Ix] = I\mathbb{E}[x] \\
\Leftrightarrow I^{-1}\mu &= \mathbb{E}[x] \\
\text{and} \quad \Sigma &= \mathbb{E}\left[(x'-\mathbb{E}[x'])^{\mathrm{T}}(x'-\mathbb{E}[x'])\right] \\
&= \mathbb{E}\left[(Ix-\mathbb{E}[Ix])^{\mathrm{T}}(Ix-\mathbb{E}[Ix])\right] \\
&= I\mathbb{E}\left[(x-\mathbb{E}[x])^{\mathrm{T}}(x-\mathbb{E}[x])\right] I^{\mathrm{T}} \\
\Leftrightarrow I^{-1}\Sigma(I^{\mathrm{T}})^{-1} &= \mathbb{E}\left[(x-\mathbb{E}[x])^{\mathrm{T}}(x-\mathbb{E}[x])\right]
\end{aligned}$$

which yields a Gaussian in the original space with parameters $(I^{-1}\mu, I^{-1}\Sigma(I^{\mathrm{T}})^{-1})$.

This back-transformation enables us to assign cluster probabilities that are neither grid-based nor require the storage of grid information or the ICA transformation matrix for each cluster.

5.2.4 Extending Imitate: Automated Grid Selection [Alg. 4; Line 4]

Additionally, we adjust IMITATE's method of selecting the grid granularity: Instead of repeating the entire modeling and augmentation process and using the results with the highest confidence score (as it was done in IMITATE), we use the *corrected Akaike information criterion (AICc)* [57] to select, for each dimension, the grid over which a histogram represents the data best. This adjustment is necessary since MIMIC uses repeated calls of the IMITATE fitting procedure, and the inflicted computational expense of the confidence-based strategy would be infeasible.

The decision on which information criterion to use is based on preliminary experiments developed under our supervision by Duncanson [43]. We investigated the quality of the selected number of bins for several information criteria: the Akaike information criterion (AIC) and its corrected version AICc, the Bayesian information criterion (BIC), and the Hannan-Quinn criterion (HQC). See Granichin, Volkovich, and Toledano-Kitai [57] for the definitions. We generated 1000 standard Gaussian datasets of random size $n \sim \mathcal{U}(5000, 50000)$ with artificial biases removing $h \sim \mathcal{U}(0, 100)\%$ of the data above a threshold $t \sim \mathcal{U}(0, 1)$. For all tested numbers of bins in $\{5, \dots, 100\}$ we evaluate the KL-divergence between the ground-truth $\mathcal{N}(0, 1)$ and the Gaussian IMITATE fits to the histogram representation of the biased data. The results are shown in Figure 5.4 and indicate that all information criteria perform similarly and better than the confidence strategy described in Chapter 4. We select AICc since it seems to have a slight edge over its competitors.

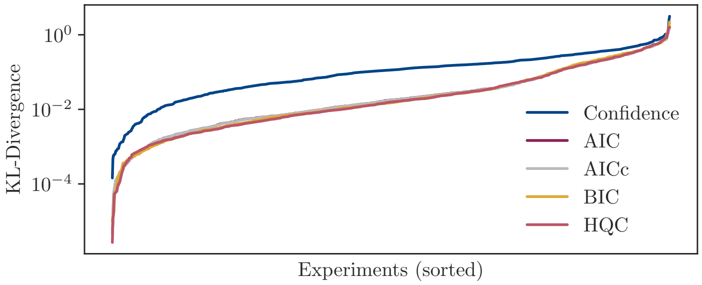


Figure 5.4: Test of different information criteria to determine the optimal number of histogram bins for IMITATE

5.2.5 Growing Clusters [Alg. 4]

For a cluster L, IMITATE provides us with a multivariate Gaussian θ_L and a grid G_L indicating where and how much (n_L) data might be missing. As outlined in Algorithm 4, both are passed on to a scoring function that estimates for each point p outside L how well it contributes to filling in the gap between the present (h) and fitted (f) density (first term), and how likely it belongs to that distribution (second term):

$$s(p) = d \log[\max\{f(p) - h(p), 0\} + 1] + \log[f(p) + 1],$$

where d denotes the number of features and puts more emphasis on filling the gap for higher dimensions. Using the score, MIMIC then searches for n_L fitting candidates in batches b_i to overcome locally optimal solutions.

A batch of candidates C is drawn randomly with probabilities based on the score values s and added to the cluster if adding it improves the likelihood of the fitted Gaussian (i.e., the parameters θ_L describing the cluster) given the assigned data points (i.e., $B'_L \cup C$). In other words, we aim to find $\arg\max_C \mathbb{P}[\theta_L \mid B_L \cup C]$ and add it to the cluster if

$$\mathbb{P}\left[\theta_L \mid B'_L \cup C_i\right] > \mathbb{P}\left[\theta_L \mid B'_L\right].$$

For the sake of brevity and readability, we denote $X_C \coloneqq B_L \cup C$. In order to avoid underflow errors, we use logarithms. Exploiting Bayes' theorem and since log-transformation preserves maxima, this term can be expressed as:

$$\begin{aligned}\arg\max_C \mathbb{P}[\theta_L \mid X_C] &= \arg\max_C \log\left(\mathbb{P}[\theta_L \mid X_C]\right) \\ &= \arg\max_C \log\left(\frac{\mathbb{P}[X_C \mid \theta_L]\cdot\mathbb{P}[\theta_L]}{\mathbb{P}[X_C]}\right) \\ &= \arg\max_C \log\left(\frac{\mathbb{P}[X_C \mid \theta_L]}{\mathbb{P}[X_C]}\right) \\ &= \arg\max_C \left(\log\mathbb{P}[X_C \mid \theta_L] - \log\mathbb{P}[X_C]\right),\end{aligned}$$

since $\mathbb{P}[\theta_L]$ does not depend on C and can hence be omitted in $\arg\max_C$.

Using the Imitate output, we obtain histogram values h and fitted Gaussian densities f over a grid. With these, the first term, $\log\mathbb{P}[X_C \mid \theta_L]$, can be approximated via the grid representation as follows:

$$\begin{aligned}\log\mathbb{P}[X_C \mid \theta_L] &= \log\prod_{p\in X_C}\mathbb{P}[p \mid \theta_L] \\ &\approx \log\prod_{\text{grid cells } c} f(c)^{h(c)} \\ &= \sum_{\text{grid cells } c} h(c)\cdot\log f(c),\end{aligned} \tag{5.1}$$

which is a term that can be efficiently computed without the risk of underflow errors. The second term, $\log\mathbb{P}[X_C]$, can be transformed into $\log\int_{\theta_L^i}\mathbb{P}[X_C \mid \theta_L^i]\cdot\mathbb{P}[\theta_L^i]\,\mathrm{d}\theta_L^i$ using the law of total probability. We simplify the term by assuming that only one data-generating model exists that we parameterize as follows: The mean μ_0 is the grid center, and the covariance matrix Cov_0 is set up as a diagonal matrix, ensuring the grid borders are the minimal axis-aligned bounding box for a Gaussian around μ_0 truncated at the usual 3 standard deviations. We denote the parameters of the Gaussian (μ_0, Cov_0) as θ_L^0 and obtain the simplified term $\log\mathbb{P}[X_C \mid \theta_L^0]$. By evaluating the corresponding probability density f_0 for all grid cell centers, this term can be calculated similarly to Eq. 5.1 as follows:

$$\begin{aligned}\log\mathbb{P}[X_C] &\approx \log\mathbb{P}[X_C \mid \theta_L^0] \\ &\approx \log\prod_{p\in X_C} f_0(\text{cell\_center}(p))\cdot s \\ &= |X_C|\cdot\log s + \sum_{p\in X_C}\log f_0(\text{cell\_center}(p)),\end{aligned}$$

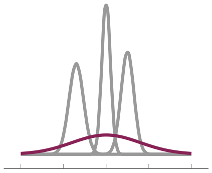
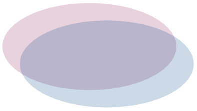
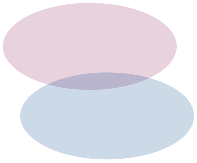
Overgrown cluster	**Symmetric overlap $>$ 80%**	**Symmetric overlap $<$ 20%**	**Any other case**
 ⇒ remove overgrown cluster	 ⇒ merge clusters	 ⇒ do not consider for merging	? ⇒ merge if it increases Gaussianity

Table 5.1: Four different cases Mimic's merging procedure considers (from left to right): 1. overgrown clusters (vine, left) are removed, 2. overlapping clusters are merged, 3. hardly overlapping clusters are not considered for merge, and 4. all other clusters are only merged if it results in a more Gaussian cluster.

where s is the size of each grid cell. Although only an approximate calculation, this term is able to balance off Equation 5.1 and yields easily computable meaningful results.

In our implementation, we restart the sampling (with replacement) of a rejected batch twice in order to avoid "unlucky" choices. If points have been added, Mimic fits another multivariate Gaussian and repeats the process until no further points are added. The parameters of the last fitted Gaussian represent this cluster.

5.2.6 **Merging** [Alg. 3; Line 11]

Once the parameters for all clusters have been obtained, we make sure not to have duplicate clusters or those that are locally optimally normal but can be combined into a better fit. Additionally, Mimic risks overgrowing clusters if the initial clustering was particularly poor, e.g., if it captures the overlapping area of two clusters. Here, the point density is higher, and the Imitate procedure will demand to grow the cluster in all directions simultaneously such that it never reaches a Gaussian-like shape and continues to grow, absorbing more and more data. See Table 5.1 – Case 1 for a visualization. Such a cluster L is typically characterized by a very wide probability distribution reaching low-density values for all points, such that the points p with $L = \arg\max_i \mathbb{P}[p \mid \theta_i]$ exhibit a substantially larger distance to each other than average. To detect them, we evaluate the within-cluster-nearest-neighbor distance of each cluster and remove those whose distance exceeds a 3-standard-deviation band around the average distance. We identify and remove these overgrown clusters as a first step of the merging procedure.

The overlap $o(i, j)$ of two clusters i and j can be quantified by counting the points in the dataset for which the cluster membership is not entirely clear and weighting them using their probabilities:

$$o(i, j) = \frac{\sum_p \mathbb{P}[p \mid \theta_i] \mathbb{1}_{\mathbb{P}[p|\theta_i] < \alpha \mathbb{P}[p|\theta_j]}}{\sum_p \mathbb{P}[p \mid \theta_i]}$$

for a factor $\alpha > 1$ ($\alpha = 10$ in our implementation) where $\mathbb{1}$ denotes the indicator function. Note that o is not symmetric in its arguments which follows the intuition of overlap. Imagine two clusters in 2D arranged like a fried egg: while the yolk fully overlaps with the egg white, the reverse direction would not hold true.

Based on the parameters for all clusters, Mimic calculates the overlap between each (ordered) pair of clusters. The ones with high symmetric overlap (that is, $o(i, j) > \beta$ and $o(j, i) > \beta$ for $\beta \in [0, 1]$; $\beta = 0.8$ in our implementation) are merged right away into a cluster with parameters $((\mu_i + \mu_j)/2, (\Sigma_i + \Sigma_j)/2)$ since they can be expected to be duplicates. This corresponds to Case 2 in Table 5.1.

Clusters with a small symmetric overlap, i.e., $o(i, j) < \gamma$ and $o(j, i) < \gamma$, are not considered for merging for the sake of computation efficiency (Table 5.1 – Case 3).

All other clusters with a small, possibly one-sided overlap (that is, $o(i, j) > \gamma$ and $o(j, i) > \gamma$ for a small $\gamma \in [0, \beta]$; Case 4 in Table 5.1) are merged only if, after a probabilistic cluster assignment, the Imitate fitting error e of the merged cluster is lower than the weighted sum of the individual errors, i.e., if $e_{i \cup j} < (\#i \cdot e_i + \#j \cdot e_j)/(\#i + \#j)$ where $\#i$ counts the points with label i. In order to address the randomness of the involved ICA, we repeat this test 10 times and use a majority vote for the final decision. The merging procedure is repeated until no further clusters are merged.

In our implementation, we use hardcoded $\alpha = 10$, $\beta = 0.8$, and $\gamma = 0.2$. Note that these values reduce the number of merge tests that need to be carried out and hence reduce the computational burden while sacrificing little to no quality. Parameter tuning is not necessary as the merge tests determine the results, not the parameters.

5.2.7 Data Augmentation [Alg. 3; Line 12]

After receiving the final cluster parameter sets from the merging step, Mimic probabilistically assigns the data points to the clusters and generates points for each cluster separately to "fill in the gap" between the found and the fitted distribution as in Imitate. Points with probability 0 do not belong to any cluster and are marked as outliers. The final cluster parameters, together with the set of generated data points, yield the final output of Mimic.

5.2.8 Assumptions and Expectations

Selection Biases cannot be reconstructed without making some kind of assumption regarding the ground truth and/or the nature of the bias [9]. Hence, Mimic assumes a ground truth that can be modeled by a mixture of (possibly overlapping) multivariate Gaussians, which, in contrast to existing techniques, requires neither a ground-truth sample nor knowledge of the bias. This freedom, however, comes at a cost and forces some implicit requirements:

(i) The data cannot contain categorical, binary, or discrete features with few values (unless the histogram bins align with them), as fitting a Gaussian would not be meaningful (this limitation is inherited from Imitate),

(ii) B itself cannot consist only of Gaussian clusters or Mimic will not be able to identify growth directions,

(iii) several strongly overlapping biased clusters might not be disentangled correctly, and

(iv) the bias in each cluster is expected to have a convex shape as our component-wise analysis fails otherwise.

Lastly, biases can be misleading, pointing towards a different Gaussian than the true one and causing Mimic to introduce new biases into the data. We aim to suppress that behavior by refusing to take action if the Gaussians do not fit reasonably well (as in Imitate). This, however, causes conservative results with bias reconstructions pointing toward the right locations rather than correcting entirely. That is the reason for only small improvements in classification accuracy as can be seen in the experimental results. In practice, however, this is enough to point a practitioner toward potential problems in the data that can be corrected upon confirmation.

5.3 Experiments and Discussion

In order to investigate Mimic's ability to improve classifier performance, we set up all experiments similarly: we train three classifiers on a biased training set B, the augmented biased training set $B \cup \hat{I}$, and an unbiased training set D. The accuracy acc_B, $\mathrm{acc}_{B\cup\hat{I}}$, and acc_D of all three classifiers, respectively, is then evaluated on an unbiased test set with the hope that $\mathrm{acc}_{B\cup\hat{I}} > \mathrm{acc}_B$. After providing details on the experimental setup, we assess the impact of different characteristics of datasets on the performance.

5.3.1 Experimental Setup

In our experiments, we compare Mimic not only to the biased accuracy as a baseline but also for augmented biased datasets $B \cup \hat{I}$ where $\hat{I}$ is obtained using (i) augmentation with Imitate, (ii) clustering and augmentation with Mimic, and (iii) clustering with GMM and augmentation with Mimic which we denote as "GMMimic". GMM selects the number of clusters (from 1 to 20) that achieve the best BIC and initializes using k-means.

Classifiers. As classifiers, we use decision trees (DT), support vector machines with RBF-kernel (SVM), and random forests (RF) with 100 trees. All parameters are kept at sklearn's default values [110].

Datasets. We use synthetic datasets since they allow us a high level of control, and real-world datasets to demonstrate that Mimic is indeed applicable in practice. Real-world datasets are taken from the *UCI machine*

Dataset	**Predicted Attribute**	**Biased Set B**	**Omitted Features**
Wholesale Customers	Region	Frozen > 409.5	Channel
Vertebral Column	Normal/ Abnormal	Spondylolisthesis Grade ≤ 14.855	-
Banknote	Class	Variance > 0.32	-
Diabetes (130 US hospitals)	Diabetes Med	#Medications > 9.5	All but Age*, #LabProcedures, #Procedures, #Medications, #Outpatient, #Emergency, #Inpatient, #Diagnoses
Skin Segmentation	Class	R ≤ 170.5	R**

Table 5.2: Description of real-world datasets used in the experiments. *The categorical values were transformed into numerical variables. **This attribute was omitted to achieve consistency with the IMITATE experiments.

learning repository [1, 41, 138]. Semi-artificial biases are created as in the previous chapter by splitting into B and I using a decision stump (the larger subset is taken for B). This way, the impact on the classification accuracy is guaranteed. We specify the predicted attribute as well as the created bias in Table 5.2.

Synthetic datasets are generated using a specified number of clusters per class and dimension. Each cluster is generated as a multivariate Gaussian with a random covariance matrix (via the Cholesky decomposition) and mean. All means are generated within the unit cube and pushed away from the center using a parameter that controls the *spread* of the clusters. If not explicitly mentioned, we used a medium spread of 100. Biases are created as described for IMITATE for two randomly selected dimensions per cluster: A hyperplane is rotated through the cluster center by a random angle. Data points above that plane associated with the cluster are omitted in B. Note that this is a hard bias which we decided to use as it challenges our method further (see Section 4.4 for a study on the impact). In order to ensure that the bias has an impact on the classification accuracy, we select only those randomly generated datasets and biases that inflict at least a 10% accuracy drop with the SVM classifier. We generated datasets of size 5000 and provide all methods, parameters, and seeds necessary for the generation in the provided code.

Performance Measure. All synthetic experiments are repeated 30 times to compensate for the randomness in the dataset generation, and we report the median results to account less for unfortunate synthetic datasets. Experiments on real-world datasets are repeated 10 times as there is no dataset generation step involved. Here, we report the mean together with 90% confidence intervals. We measure the performance as the *improvement over the biased accuracy* and normalize using the unbiased accuracy, i.e., $(\mathrm{acc}_{B\cup\hat{I}} - \mathrm{acc}_B)/(\mathrm{acc}_D - \mathrm{acc}_B)$.

5.3.2 Results

Subsequently, we investigate the influence of different dataset characteristics on the performance. While varying the parameters mentioned in the experiments, we keep all others fixed to isolate the variables in question.

Unbiased Datasets

Being able to mitigate a selection bias is important; however, if Mimic is presented with an unbiased dataset, it should not "correct" it. We count the number of points being generated for unbiased and corresponding biased 2D datasets with random spreads between 100 and 200, and we normalize the counts with the dataset size for comparability. Figure 5.5 shows that substantially fewer data points are generated for the unbiased datasets. We suspected that these data points result from histogram inaccuracies and confirm that suspicion by applying Imitate's purging strategy (that is, it removes the generated data points that are not distributed densely enough): the generated points for the unbiased datasets do not focus on certain areas and are hence removed as noise. Almost no points remain on the unbiased dataset, while there are about 20% of generated points left on the biased

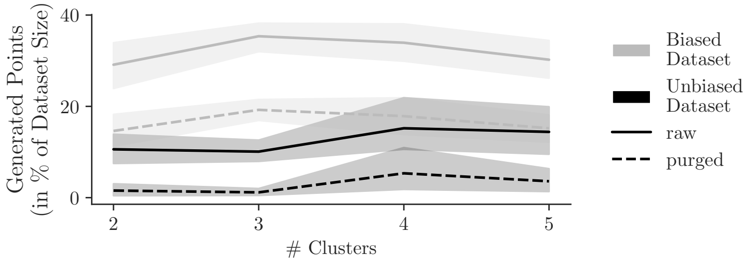


Figure 5.5: Comparison of MIMIC's behavior with a present (light gray) and absent (dark gray) bias on synthetic data. Lines indicate the mean; bands are 95% confidence intervals.

datasets. This trend is consistent regardless of the number of clusters in the datasets.

Dimensionality

The dimensionality of synthetic datasets is closely related to their difficulty as higher dimensions naturally increase the distance between clusters even while under the same cluster-to-center distances. Figure 5.6 demonstrates this, as lower dimensionalities typically exhibit poorer performance than higher ones, but this effect vanishes with larger numbers of clusters. GM-Mimic and MIMIC show similar performances for a larger number of clusters, while MIMIC clearly dominates when only a small number of clusters is present, regardless of the dimensionality. IMITATE shows strong performance in this case, too, but decreases rapidly since it operates with only one cluster.

We observe the highest improvements among all three models for random forests. This effect is likely to be observed because random forests more draw more fine-grained decision boundaries. Hence, adding generated points is more likely to have an impact.

Cluster Overlap

The center-to-cluster distances of the clusters directly affect the difficulty of the clustering task as they control the overlap. In order to investigate the influence, we adjust the spread parameter in the dataset generation and illustrate the results in Figure 5.7. GMMimic and MIMIC both show

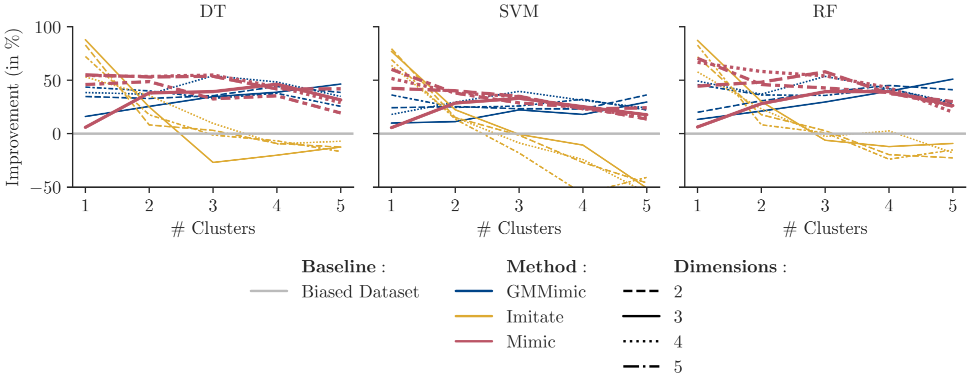


Figure 5.6: For each classification method, we compare the impact of the dataset dimensionality and the number of clusters on the performance.

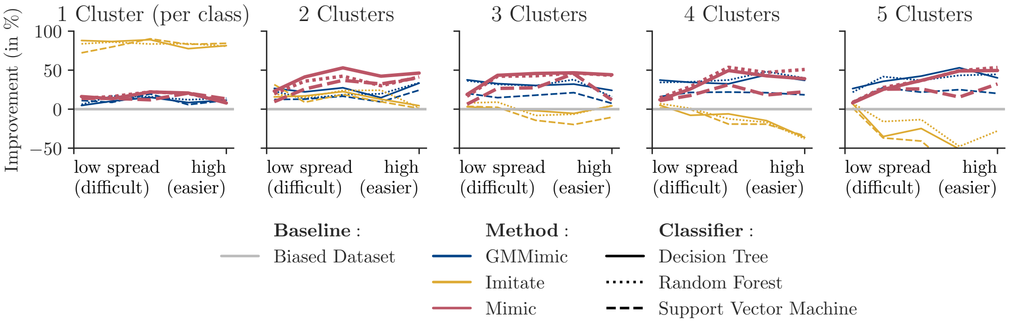


Figure 5.7: Datasets in 2D with two classes have been generated with different numbers of clusters per class. The spread (on the x-axes) indicates how much the clusters are being pushed away from the center, and a low spread corresponds with a high overlap. Even with a large number of clusters, MIMIC performs consistently well. However, high overlaps seem to be addressed better by GMMimic.

improvements even for a large number of clusters and high overlaps. MIMIC demonstrates its strength, particularly for better-isolated clusters where it improves the classification accuracy by up to 50% of the drop due to the bias. The center-to-cluster distances of the clusters directly affect the difficulty of the clustering task as they control the overlap. IMITATE excels in the one-cluster case as it is designed for this case and produces less conservative results than its competitors.

Real-Life Datasets

Figure 5.8 summarizes the results on five real-world datasets. For most datasets, we can see MIMIC's potential to improve the classifier accuracy

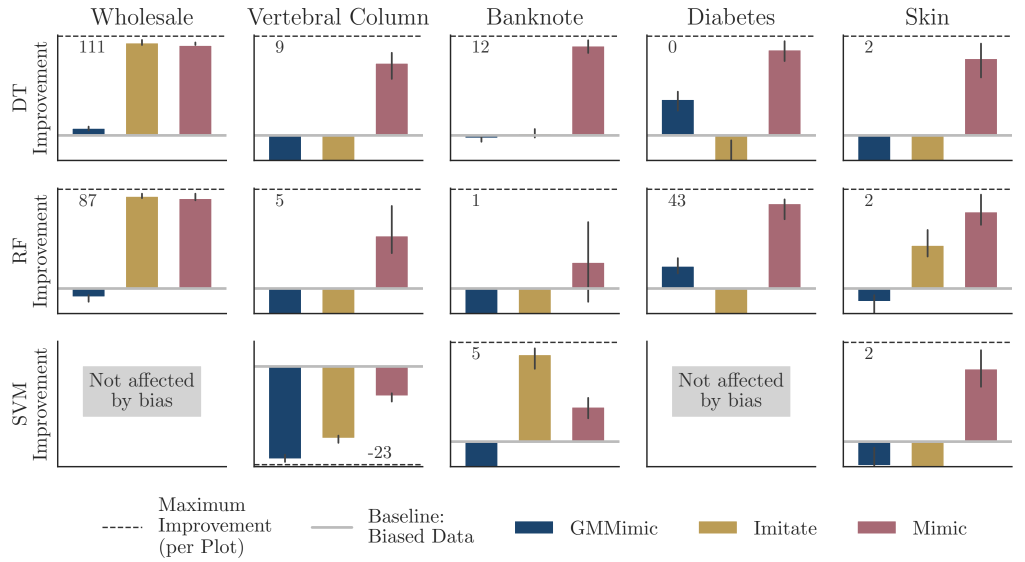


Figure 5.8: We compare the degree to which the classifier accuracy can improve when different augmentation techniques are used. The baseline (gray line) represents the accuracy when the classifiers are trained on the biased dataset alone. The numbers annotating the dashed lines indicate the y-axis value at the top of the plot window – 100% corresponds to training on a ground-truth sample. Note that we omit the y-axis labels and replace them with the dashed line indicating the maximum improvement (maximum y-value) for each plot. The bottom of the plots is cut off unless MIMIC's performance is displayed there for easier comparison. The black lines are 90% confidence intervals and indicate significant differences from the baseline if they do not touch it.

substantially, in most cases more than its competitors. A few observations are noteworthy: On the Wholesale dataset, IMITATE performs well since it consists of only one cluster per class. The Vertebral Column dataset seems particularly hard for all methods as the semi-synthetic bias removes 70% of the majority class points (which, therefore, cannot be reconstructed by any method), leaving an almost balanced classification problem with full overlap and an imbalanced test set. Here, the tree-based methods essentially select the majority class, and MIMIC is able to tip the scales favorably but cannot help the SVM. Overall, although GMMimic demonstrates solid performance on the synthetic dataset, it does not seem to generalize well to real-world datasets.

5.3.3 Limitations

Overall, the experiments show that the application of an augmentation technique can provide a meaningful improvement on a biased dataset. While Imitate is designed for datasets with only one cluster per class, GMMimic and Mimic can improve upon its performance when dealing with multi-cluster datasets. The experiments on synthetic datasets with artificial biases point towards a similar performance of GMM- and Mimic-based data augmentation. On the real-world datasets, however, we do not see this confirmed: Mimic can further improve the classification performance. Further research could investigate where which method tends to be superior and particularly if a symbiosis of both can be beneficial, e.g., with GMM as an initial model and a Mimic-inspired merging strategy and augmentation. Existing pitfalls of all methods are their inability to deal with discrete, binary, and categorical data. One-hot encoding together with a dimensionality reduction via PCA as a pre-processing step might help improve classification performance but involves a loss of information and interpretability of the generated points. We explore this option in the following chapter.

Mimic relaxes Imitate's assumption that the ground-truth dataset consists of only one Gaussian per class. Instead, it can model multiple Gaussian clusters or even approximate non-Gaussian clusters with mixture models. This makes Mimic applicable to a substantially wider range of datasets. However, not all distributions can be approximated well as a mixture of Gaussians. Future extensions should include an automated test of applicability as well as approaches applicable to a wider range of distributions.

5.4 Conclusion

Machine learning models inherit selection biases from datasets causing them to predict inaccurately if the biases remain undetected. Existing bias mitigation strategies require certain kinds of knowledge of the bias or the ground-truth. In real-world scenarios, however, this requirement often cannot be met. A first attempt to detect and mitigate selection biases in a "blind" setting has been made with the Imitate algorithm, although it is limited to datasets with only one Gaussian cluster per class.

In this chapter, we introduced Mimic, a technique that uses Imitate as a building block but overcomes these limitations and can model a wider range of datasets exploiting mixtures of Gaussians. As such, multi-cluster modeling of many non-normally distributed datasets is now possible.

Although limitations still exist as discussed in Section 5.3, we believe that Mimic is a major step forward towards automated bias identification and mitigation in the case that no knowledge of the bias or the ground-truth exists.

6 An Application: Assessing and Preventing Bias in Growing Chemical Databases

The research presented in this chapter has been adapted from

> K. Dost, Z. Pullar-Strecker, L. Brydon, K. Zhang, J. Hafner, P. Riddle, and J. Wicker, "Combatting over-specialization bias in growing chemical databases," *Journal of Cheminformatics*, vol. 15, no. 53, 2023. DOI: 10.1186/s13321-023-00716-w.

The results of this chapter are available in the GitHub repository github.com/KatDost/Cancels and the proposed algorithm is contained in the PyPI package imitatebias.

6.1 Introduction

In domains where gathering data requires time-intensive experiments, predicting likely outcomes for experiments helps concentrate efforts on the right experiments. One example is the development of effective yet sustainable and environmentally-friendly products, e.g., pesticides, that (hopefully) fulfill their purpose and then quickly degrade into harmless non-toxic compounds over time. Experiments involve long-term studies of each compound's effect and observation in soil under different environmental conditions. Ruling out compounds that might not bring the desired chemical properties or degrade into toxic by-products is an essential aspect of the development process.

Similar challenges arise in other areas of chemical research and development, such as the design of new pharmaceuticals, fragrances, or commodity chemicals.

However, predictive models learn from and specialize to the data provided to them [24, 129]. While this specialization is useful up to the point where the desired domain is accurately captured [65, 80], the models can over-specialize. Starting from the initial dataset, a trained model will only be able to make reliable predictions in densely populated areas of the compound space, leaving the remaining areas outside of the model's applicability domain. As a consequence, it will suggest a set of experiments well within its applicability domain, shifting the overall data distribution towards in-domain data. Should the model be re-trained after obtaining the new experimental results, it will put more emphasis on the now densely populated areas, further shifting the data distribution. After a few iterations of dataset growth, we can observe that the applicability domain is either consistent or shrinking despite the additional data [58], and new potentially interesting areas of the compound space will never be explored. For example, in density-based applicability domain techniques using relative thresholds [5, 122], the density ratio between dense and sparse areas changes – and rightfully so since a trained model will increasingly focus on dense areas and become less reliable on sparse ones. This scenario is a self-reinforcing type of selection bias where the model chooses to obtain new results for compounds it can already predict reliably and therefore slows down or even stops learning.

A similar effect can be observed when humans rather than models choose the compounds to experiment with [31]. Jia *et al.* [73] argue that anthropogenic factors play a key role in the compound selection process for experiments and hence the development of datasets. More than on the cost, availability, or ease of use of available candidate compounds, researchers tend to base their selection on their past successes and that of their colleagues or research articles. This results in a specialization spiral, iteratively narrowing down the scope within which models and humans can make informed decisions.

Active learning [127] is a tool that aims to break the cycle by selecting the most informative experiments for the model instead. Although active learning has been shown to suffer from shifts in distribution [105], it is capable of slowly expanding the compound space and will eventually even explore beyond the desired degree of specialization. In addition, active learning is always model-dependent. This is a major drawback since datasets, especially those requiring long-term experiments, can and will be used for different purposes over time, and it is often infeasible to gather new data specifically for a model.

Instead, in this paper, we suggest CANCELS (*CounterActiNg Compound spEciaLization biaS*), a model-free and even task-free method to generally point out potential shortcomings of the data and improve the quality without losing the desired specialization to a specific domain. CANCELS is an extension of the IMITATE and MIMIC algorithms that overcomes their restriction to real-valued tabular data. CANCELS adapts ideas from both and extends them to select data from a pre-defined pool rather than generating which allows us the freedom to select meaningful compounds worth experimenting with from a data quality standpoint.

Possible applications for CANCELS include *computer-aided drug design (CADD)* [101, 125]. These methods greatly support the drug discovery and development process by modeling the behavior of compounds, but, as is common in all data-based methods such as machine learning, they can only make reliable predictions for compounds that are similar to what those models trained on [85]. This might be one of the key reasons why, despite the progress of CADD methods in recent years, still only a small fraction of the chemical compound space has been explored in the search for drug candidates (as stated by Mouchlis *et al.* [101]). While *de novo* drug design [7, 85, 126, 129] aims to base the candidate search on a broader space, it also relies on the quality of the underlying dataset [76, 111], and it disregards the distributions of the resulting compound set and their implications for future predictors or generators [80]. CANCELS can help select additional compounds to test in order to improve the dataset quality for future drug design cycles while still testing the most promising candidates for today's search.

The remainder of this chapter is organized as follows: The following section adjusts the problem stated in Section 4.2 formally to the new setting

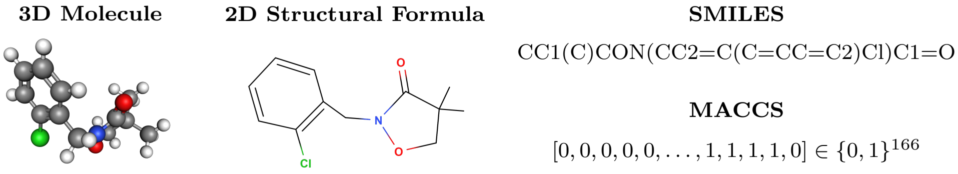

3D Molecule
2D Structural Formula
O
N
O
Cl
SMILES
CC1(C)CON(CC2=C(C=CC=C2)Cl)C1=O
MACCS
$[0, 0, 0, 0, 0, \ldots, 1, 1, 1, 1, 0] \in \{0, 1\}^{166}$

Figure 6.1: Different representations of the chemical compound 'clomazone'. A molecule is essentially a graph where nodes are atoms, and edges are connections. This graph can be embedded in a 3D or 2D space. Alternatively, SMILES encode the structure of the compound in a string. Numeric representations are popular as they allow for standard data mining tools. One example is MACCS fingerprints, where each entry of the array indicates the presence/absence of a particular substructure [135].

in cheminformatics. Section 6.3 reviews related works on active learning in chemistry and biases in the chemical compound space that are specific to this application and are not covered in Chapter 3. Section 6.4 introduces the Cancels algorithm. Section 6.5.2 presents and discusses experimental results. Finally, Section 6.6 concludes the chapter.

6.2 Problem Statement

Both Imitate and Mimic generate examples to mitigate potential biases they detect. In cheminformatics, however, generating data in this fashion is not straightforward. One major reason is that the chemical compound space is not a real-valued space, as compounds are small, three-dimensional objects consisting of different atoms connected via different bonds. Figure 6.1 shows different ways to represent a compound.

However, there are restrictions on which combinations of bonds and atoms are feasible and stable. To avoid generating infeasible compounds, we aim to select from a large pool of (unlabeled) but known-to-be-feasible compounds instead. Formally, we state the problem we aim to solve as follows:

Problem 3 (Pool-based)**.**
Let D be an (unknown) compound dataset (potentially with labels or properties) that is representative of an underlying distribution that we consider to be the ground truth. Given only a biased subset $B \subset D$ and a pool P of candidate compounds, the task is to select a set of compounds $P_{sel} \subseteq P$ such

that a model trained on $B \cup P_{sel}$ would provide minimally different outputs (such as predictions, clusters, etc.) from one trained on D.

To solve the adjusted problem, we propose Cancels, which utilizes (parts of) both Imitate and Mimic while overcoming their limitations in this context.

6.3 Related Research in Chemistry

In this section, we review fields that deal with related problems and highlight the differences to our problem statement. The topics we include are bias detection with ground-truth samples and active learning, particularly for chemistry. Additionally, we discuss biases in the chemical compound space.

6.3.1 Active Learning in Chemistry

Active learning is a semi-supervised machine learning setting that utilizes information from a trained model to infer the samples which would most improve the model [127]. The main aim is to train models using fewer labels than would be required for random sampling, as these are often expensive to obtain. Since, similarly to Cancels, active learning also selects additional data points, we compare both approaches here.

An active learning strategy consists of an initial model, usually trained on a small amount of randomly selected data; a query strategy, which is responsible for identifying the most informative samples; and a setting, which determines how those samples are obtained. A wide variety of query strategies have been proposed in prior work, but uncertainty-based strategies are the most common [127]. These strategies evaluate the confidence of the model on each sample, and samples with the lowest confidence (highest uncertainty) are considered the most informative. New samples can be obtained from an unlabelled pool (*pool-based*) or synthesized de novo (*query-synthesis*). In practice, pool-based active learning is typically preferred as synthesized samples are often difficult to label or simply invalid [10].

In cheminformatics, active learning has demonstrated the potential to improve the quality of models while reducing the amount of data required

[132]. For example, Smith *et al.* [132] used active learning to train a model for molecular energetics that outperformed a model trained using random selection while using only 10% of the available labels. Active learning has also been applied to the fields of drug discovery [120], toxicity prediction [60], chemogenomics [121], and others [162].

In contrast to the approach presented in this paper, active learning attempts to select samples which *improve the current model*. The selected samples are not necessarily transferable to other models [140]. Additionally, active learning intentionally seeks to bias the dataset towards informative samples and does not aim to explore the space or improve the dataset quality.

6.3.2 Bias in the Chemical Compound Space

Hert *et al.* [65] aim to quantify the bias of screening libraries towards biogenic molecules, given an estimate of the entire space and a specified optimal dataset, i.e., the optimal bias. To measure the bias, they assess the similarity between the observed and the optimal dataset. Given that the chemical space is estimated to contain at least 10^{60} molecules with 30 or fewer heavy atoms [17], stretching even today's largest databases across that space to achieve the often idealized uniform distribution [7, 58] would result in very sparse coverage. The authors hence postulate that, as opposed to the aim to cover the entire space uniformly, biases toward specific domains are essential to enable a successful performance of models and researchers within those domains. In agreement with this, in this chapter, rather than aiming to cover the entire compound space, we suggest a technique that mitigates the bias within an observed dataset while preserving its bias within the compound space. Therefore, despite improving the dataset quality, we preserve the dataset's specialization to its domain.

Sieg, Flachsenberg, and Rarey [129] investigated multiple benchmark datasets for structure-based virtual screening and discovered that they are all inherently biased since they have grown depending on human decisions based on individual assumptions and goals. When screening for specific properties, these biases persist and eventually find their way into models trained on these datasets resulting in negatively impacted model performance [76]. Attempts to mitigate the dataset biases during screening evolves around different sampling techniques, or strategic omission of features [129]. While those are feasible approaches in large databases, they mean a substantial loss of information in small datasets [71] such as those we are working with. Here, the long-term goal must be to smooth out the biases within the dataset domain and improve the data quality in the future.

6.4 Proposed Method

When presented with a potentially biased dataset, we would like to identify present biases and mitigate them in subsequent experiments. The Imitate and Mimic algorithms presented in the previous chapters deal with this problem for real-valued, numeric, and tabular data but are not applicable to the chemical compound space. Compounds can be represented in a variety of different ways, e.g., as SMILES, molecules, or MACCS fingerprints, but none of these representations fit Imitate's and Mimic's criteria. Additionally, to mitigate a bias, both algorithms generate data that smooths out the distribution of the biased dataset. However, random generation of chemical compounds will most likely not result in meaningful and feasible compounds. We address both problems with our novel algorithm, Cancels (*CounterActiNg Compound spEciaLization biaS*).

The idea behind Cancels is to represent the compounds in the potentially biased dataset as *molecular access system (MACCS)* fingerprints because of their widespread use, fixed lengths, efficiency to compute, and solid performance in a diversity of applications [135]. Based on a comparison of different compound representations, we found that MACCS fingerprints also perform well in our case (see Sec. 6.5.2 and Fig. 6.10 for details). We then use *principal component analysis (PCA)* to strongly reduce the dimensionality of the data and obtain Gaussian-like distributions. In the PCA space, Imitate can be applied, with adaptations (as discussed below), and point to potential biases. Data to mitigate the bias could be generated in this space but not transformed back to the original space leaving the output hardly interpretable. Instead, we propose to use the PubChem [81] database

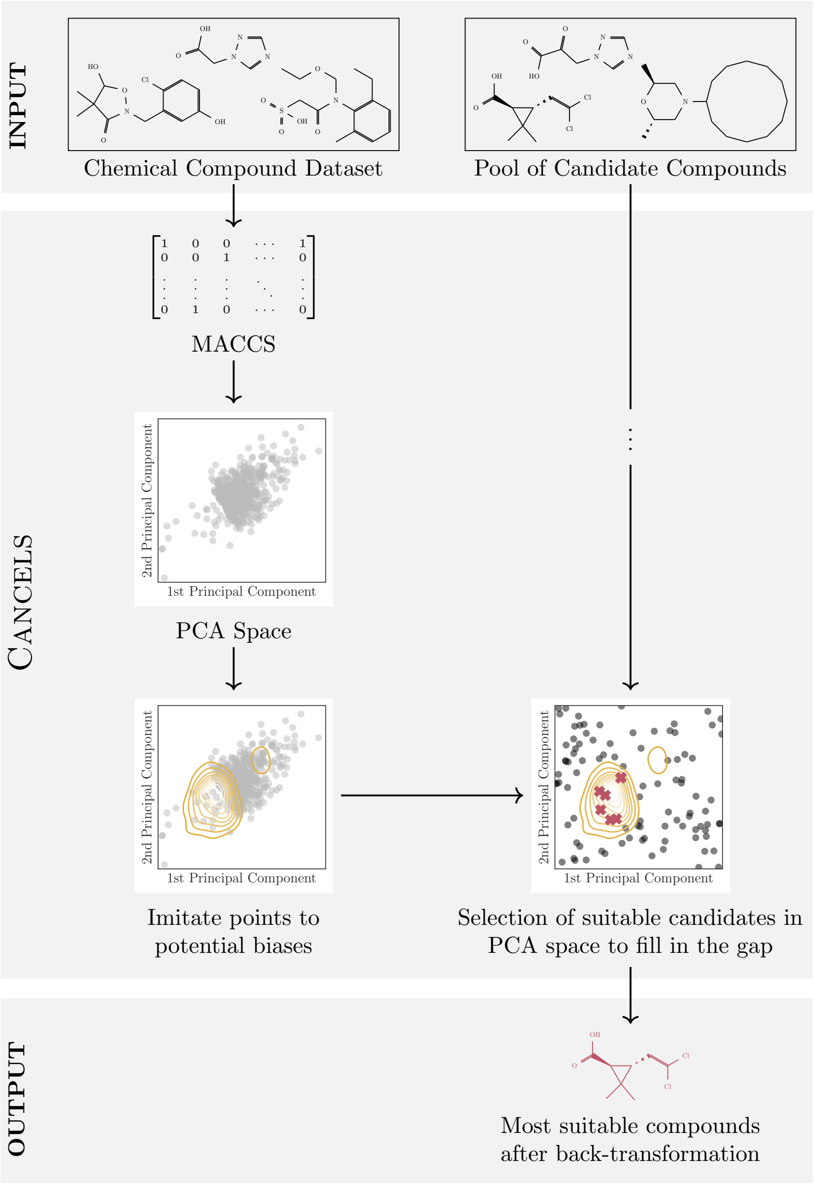

INPUT
Chemical Compound Dataset
Pool of Candidate Compounds
CANCELS
MACCS
2nd Principal Component
1st Principal Component
PCA Space
Imitate points to potential biases
Selection of suitable candidates in PCA space to fill in the gap
OUTPUT
Most suitable compounds after back-transformation

Figure 6.2: Overview over CANCELS.

as an unlabeled pool of candidates and project each of them into the PCA space. Rather than generating new data, CANCELS chooses from the candidates. As a result, we not only ensure that a back-transformation to the original compound space is possible but also that the selected candidates to mitigate the bias are indeed feasible compounds. Figure 6.2 summarizes the procedure. The remainder of this section discusses all involved steps in detail.

6.4.1 Data Transformation

Starting from a potentially biased set of compounds, we represent each of them using the MACCS fingerprint since it provides us with a fixed-length feature representation. MACCS fingerprints have been shown to include correlated features causing distance measurements to be flawed [84]; however, we subsequently reduce the dataset dimensionality and thereby mitigate the effect of related features. Cancels uses PCA to reduce the compound dataset expressed as MACCS fingerprints to the first n_{PC} principal components. If n_{PC} is sufficiently small (see Fig. 6.11 for a comparison of different values; we use $n_{\mathrm{PC}} = 5$ in our experiments), we can observe continuous non-discrete distributions over the axes to which Imitate can be applied.

6.4.2 Bias Identification

Once the compound dataset is transformed into PCA-space, Imitate exploits the orthogonality of the principal components and analyzes the dataset distribution over each of them separately. Histograms or kernel density estimation (KDE) evaluated over a grid approximate the data's probability density. KDE is preferable for small datasets since it is less sensitive to the choice of the grid, whereas histograms are substantially faster to evaluate. Similar to Imitate, we choose the type of density estimation based on the dataset size (with a threshold of 1000 compounds) and select the grid granularity that optimizes the corrected Akaike information criterion [57].

Using the density estimates on the grid as the targets and their square as weights, Imitate fits a scaled and truncated Gaussian that models observed data as closely as possible but might over-estimate areas that are under-represented in the data. This discrepancy between observed data and fitted Gaussian points to potential biases. Imitate's weighted optimization is the key to this result: It puts more emphasis on higher density values during the optimization allowing room for error on lower densities under the premise that densely populated areas are more 'trust-worthy' than sparse ones. However, there is no guarantee that Imitate identifies areas as biased that are actually populated in the compound space.

6.4.3 Extending Imitate: Boundaries

To alleviate the problem that IMITATE points to areas of the compound space that do not contain feasible compounds, we need to derive a method to provide the optimization process with boundaries. Luckily, the goal is to smooth out the distribution to obtain a Gaussian density. While this problem has only one global optimum, it has multiple local optima that bring equally smooth Gaussians at the cost of filling in more compounds. If IMITATE converges to a globally optimal solution that is outside the feasible compound space, we redirect it to the next best solution within the space unless the quality gap between the solutions is too extreme. The boundaries of the feasible compound space are extracted from the pool that is used to select bias-mitigating solutions.

In order to give the user control over the acceptable quality gap, we suggest a parameterized solution. Instead of using constrained optimization, we adjust the optimization target and weights. Out-of-bounds optimization targets are set to 0, and their weight is set to $w > 0$ times the highest within-bounds weight (see Section 4.3 for details on the weights and optimization). A small w will have little impact on the optimization, and the obtained Gaussian is not likely to change. The larger w is, the more strongly the optimization is forced to find a different solution. Intuitively, w quantifies the acceptable quality gap since errors on out-of-bounds targets can be translated to errors in high-accuracy regions with respect to the grid and the size of the out-of-bounds region.

In preliminary experiments, we generated multiple synthetic datasets with 1000 data points in four Gaussian clusters and two dimensions. The Gaussian means and covariance matrices were generated randomly (the latter via the Cholesky Decomposition), and points were drawn from these clusters in random ratios. We observed that $w = 10^3$ performed reasonably well among all synthetic datasets and decided to use it for our experiments since it is sufficiently strong to move the optimizer to a suitable within-bounds optimum unless there is no other reasonable solution. See Figure 6.3 for a comparison of different choices for w. Once the Gaussian has been redirected, compounds need to be identified that are capable of filling in the gap.

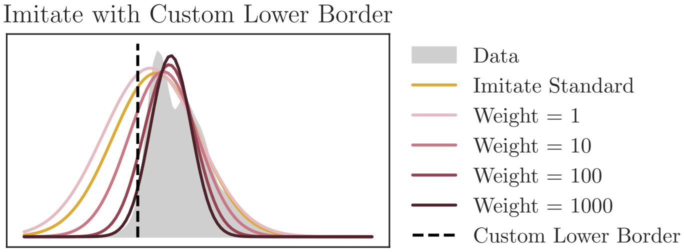


Figure 6.3: Comparison of different weights for Imitate with a custom boundary.

6.4.4 Identifying Compounds to Fill in the Gap

Univariate Gaussians fitted to each component separately can be combined into a multivariate Gaussian (see Section 5.2.3 for details) pointing to biases in PCA space. To mitigate these biases, compounds need to be identified that, when added to the dataset, smooth out its distribution by filling in the gap between present data and fitted Gaussian.

The Mimic algorithm iteratively uses Imitate to find flaws in initial clusters, scores and adds points mitigating these flaws until it finds a bias-aware Gaussian clustering of the data. In each step, after obtaining a new target Gaussian from Imitate, Mimic scores all available points from other clusters and uses the scores to randomly select candidates to be added to the cluster. It stops once adding further points would not improve the fit of the Gaussian.

Cancels adapts this procedure and exploits Mimic's scoring function to select compounds from the pool transformed into the same PCA space. Note that PCA as a dimensionality reduction technique is not invertible. Hence we need to store the mapping of pool compounds from the original to the PCA space in order to infer knowledge from the chosen candidates. Given the target Gaussian from the previous steps, Cancels scores each compound c in the pool with

$$s(c) = \mathbb{1}_{f(c)d(c)\neq 0} \left(\log f(c) + n_{\mathrm{PC}} \log d(c) \right),$$

where $f(c)$ is the density assigned by the Gaussian truncated at the triple standard deviation, $d(c)$ measures the discrepancy between fitted Gaussian and available data at this point, and $\mathbb{1}$ is the indicator function outputting

1 if the index condition holds true and 0 otherwise. After normalization, the calculated scores can be used as probabilities to randomly select compounds from the pool without replacement. Cancels stops sampling compounds when adding further compounds would not improve the fitness of the Gaussian, that is when the likelihood of the Gaussian given the training set together with the additional data does not increase, or the pool is exhausted.

Finally, Cancels uses the stored mapping to obtain the original representation of the selected compounds. These compounds can be interpreted as suggestions of which experiments to carry out next, but since they have been selected randomly based on the calculated probability distribution, a direct interpretation might not be optimal. However, the selected compounds describe underrepresented areas. Analyzing their characteristics can help the researcher gain insights into which kinds of experiments fell short in the past, and manual selection of experiments that fill in this gap can be a valuable compromise between improved data quality and meaningful experiments with interesting results.

If the pool of candidate compounds is rather small, alternatively, a researcher might prefer to use the normalized scores for the entire pool directly and, rather than sampling from it, choose manually subject to additional criteria such as availability, price, or other properties not represented by the fingerprint. Note that adding only the compounds with the highest scores does not necessarily smooth out the dataset's distribution but has the potential to create a new bias. Instead, the researcher would need to choose a large amount of highly-scoring compounds, some medium-score compounds, and even a few compounds with low scores. To simplify this process, we suggest repeatedly choosing a few compounds with high scores, adding them to the dataset, retraining Cancels, and scoring the remaining pool until a desired number of compounds has been identified.

6.5 Experiments and Discussion

To showcase what Cancels can reveal about a dataset and what insights can be won, we apply it to multiple datasets and analyze its results. Our use-case for this paper is biodegradability; however, Cancels could also be applied to other domains such as drug development. Although Cancels makes suggestions as to which compounds might be interesting to obtain labels for, analyzing these recommended compounds and their characteristics grants us more than that: It teaches us about weaknesses of the dataset and underrepresented areas that might cause a lowered model reliability regardless of the trained model. To quantitatively evaluate Cancels' performance, though, we need to train a model to evaluate changes in accuracy. Note that no matter what we evaluate, Cancels is, in any case, provided with only the MACCS fingerprints of the datasets and has no access to labels or further data characteristics. We introduce the experimental setup before presenting and discussing the results.

6.5.1 Experimental Setup

In this section, we introduce our general experimental setup. We might deviate from this setup in single experiments depending on the question we aim to answer. All deviations are listed in the following section for the sake of reproducibility. Unless stated otherwise, we use the setup introduced here. Our implementation, together with all experiments, results, and plots, is publicly available in our repository for the sake of reproducibility of results and to support further research.

Datasets. The main datasets we analyze in this paper are the EAWAG-SOIL [87] (short: SOIL) and EAWAG-BBD (short: BBD) datasets extracted from the enviPath platform [142, 150–152]. Both datasets contain biodegradation pathways capturing the chemical changes of a given starting compound (we refer to this as a "root compound") during biotransformation. SOIL was collected from publications and contains 343 root compounds. BBD stems from expert workshops and contains 248 root compounds. We prepare both datasets by extracting the compounds' MACCS fingerprints, and, to investigate the dataset development over time, join the year of publication of each pathway to its root compound where possible (299/343 root compounds in SOIL have years and 215/248 in BBD) as well as use categories from the PubChem database [81]. As prediction labels, we use the

208 transformation rules in enviPath, try to apply each of them to a compound, and assign 208 labels stating if the rule was applicable and observed (= 1), applicable but not observed (= 0), or not applicable (= missing and excluded from evaluation counts).

For a large-scale experiment demonstrating how the application of Cancels can help improve the classification accuracy, SOIL and BBD are too small to yield statistically reliable indications. Instead, in this case, we use the substantially larger Tox21 dataset containing 11093 chemical compounds tested in 12 pathway essays [68, 95]. The dataset was gathered in the 2014 Tox21 Data Challenge to pool resources to replace animal testing with model predictions in the future. The compounds are pre-assigned 12 binary labels indicating if a compound was active (= 1) or inactive (= 0) in each of the tested essays. Similarly to SOIL and BBD, we obtain MACCS keys as input features as pre-processed by Stepišnik *et al.* [135].

To put SOIL and BBD and their development over the years in a frame of reference, we downloaded all unique SMILES from the PubChem database to obtain an estimate for the span and the density of the compound space.

As pools for Cancels to select compounds from, we use the subset of PubChem with an 'agrochemical' flag to be able to extract the same use categories we obtained for SOIL and BBD. When experimenting with Tox21, we split it into subsets, so no external pool is necessary (see the following section for details).

Classifiers, Evaluation, and Stability. Tox21 is a dataset with multiple labels; hence we use a multi-label classifier to predict its labels. To achieve the most stable performance among runs and reduce the effect of randomness induced by the classifiers, we train *ensembles of classifier chains (ECCs)* [119] with 10 chains per ensemble. We evaluate the classifier performance using *multilabel accuracy* (short: accuracy)

$$\mathrm{acc} = \frac{\#\mathrm{TP} + \#\mathrm{TN}}{\#\mathrm{TP} + \#\mathrm{TN} + \#\mathrm{FP} + \#\mathrm{FN}} \,.$$

Here, #TP and #TN count the number of correctly predicted positive and negative labels, respectively. Similarly, #FP and #FN count the number of mispredicted labels.

To achieve statistical stability and ensure the significance of observed patterns, we repeat every experiment 100 times under different dataset splits and report the average results together with 95% confidence intervals.

6.5.2 Results and Discussion

Cancels is a method that, given only an unlabeled dataset, searches for biases and underrepresented regions and suggests additional compounds that can improve the dataset quality. As such, we will use Cancels as a tool to identify flaws in the dataset and investigate if the suggested compounds can indeed help improve the performance of subsequently trained models. This section investigates several questions ranging from if the bias spiral discussed in the introduction can indeed be observed in the datasets to what can be won by using Cancels. Unless specified explicitly, all experiments have been set up as outlined in Section 6.5.1.

How did the datasets develop over time? Independent of if a model is in place to support the choice of which experiments are the most promising or not, we can make the most reliable assumptions on the outcome of experiments for compounds that are similar to those we observed before. We hypothesize that this reliability shapes the process of further experimentation and hence induces specialization to the part of the compound space that is already well populated while exploration of other parts of the compound space falls short.

This hypothesis seems to be confirmed for the development of the SOIL and BBD datasets. Figure 6.4 illustrates the development of the root compound datasets from the year 2000 to 2015. We use the PubChem database as a lower boundary for the space of feasible compounds (i.e., PubChem measures the already discovered compound space). The true space is even larger but has not yet been fully explored [101]. Regardless, neither SOIL nor BBD covers the entire space – the datasets are specialized to their respective domains. Both datasets consist of one main group of compounds and a second group that is structurally different from the first one. In SOIL, this smaller group mainly corresponds to sulfonamides, typically acting as

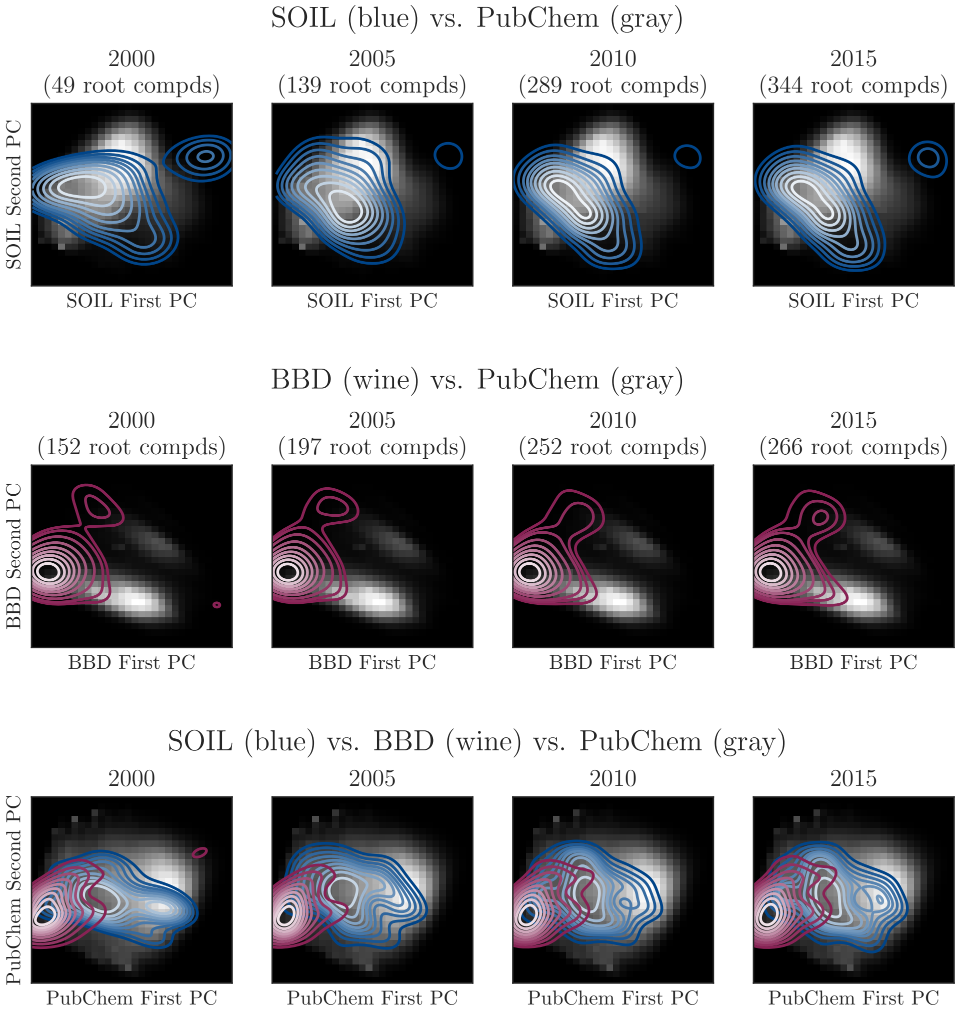


Figure 6.4: Qualitative dataset development for SOIL and BBD root compounds in relation to the compound space represented by a bivariate histogram of the PubChem dataset and visualized in the PCA spaces obtained from SOIL (top), BBD (center), and PubChem (bottom). In all three datasets, white represents the highest density.

antibacterial and antifungal agents. In BBD, it corresponds to compounds containing groups of multiply oxidated elements such as sulfates and nitro compounds. We can observe that, although compounds are continuously being added to the datasets, their distributions seem stationary, and the gaps between the main and the small groups are never closed.

Figure 6.5 further quantifies this suspicion. For both datasets, during the first years, the average distance of compounds to the center decreased, indicating that compounds were added close to the center in the already

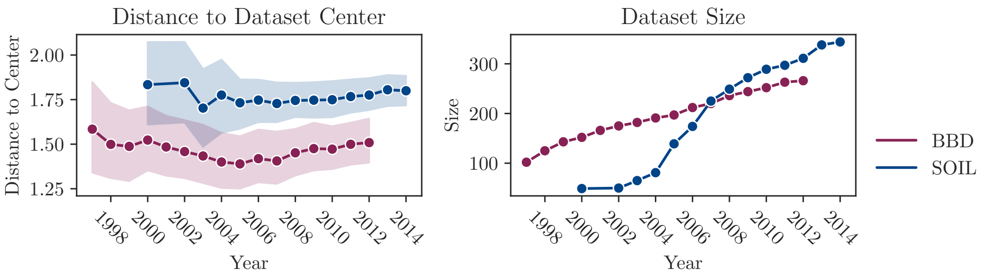


Figure 6.5: Quantitative development of SOIL and BBD root compounds in terms of the compound's average distance to their center in 10-dimensional PCA space (left) and their dataset size (right).

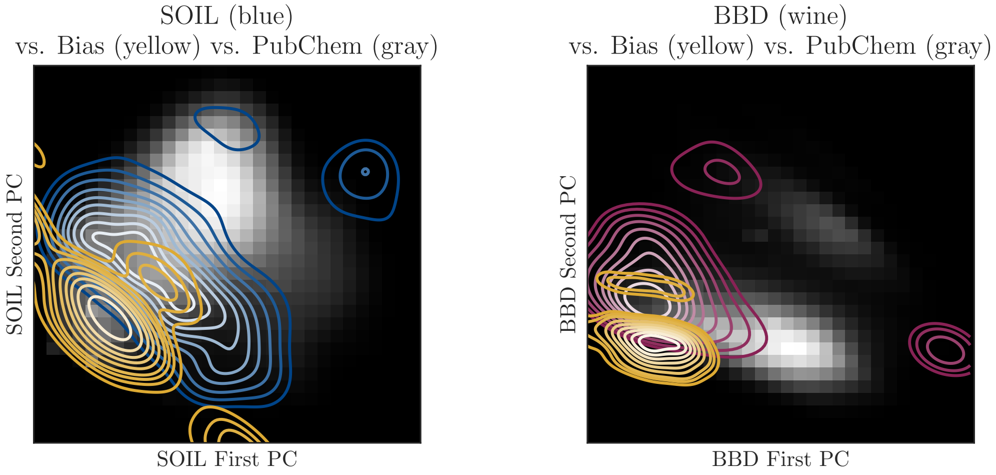


Figure 6.6: Potential biases detected by Cancels for SOIL (left) and BBD (right) visualized in their respective PCA spaces against the PubChem compound space.

populated areas. In later years, the average distance to the center has a slight upward trend; however, the standard deviation decreases at the same time, indicating a shift of the center to another already populated area. In both cases, no new areas of the compound space are being explored, although new compounds are continuously being added. Additionally, a small standard deviation implies that a model specializes to a small area while other more sparsely populated areas are less reliably predictable.

Which underrepresented regions can Cancels detect? Application of Cancels to the SOIL and BBD datasets reveals the underrepresented regions displayed in yellow in Figure 6.6. When comparing the datasets and those regions to the entire compound space estimated using PubChem, we can see that mitigating these biases, while potentially improving the dataset quality, does not generalize towards covering the entire chemical space but rather smooths out the dataset's distribution locally while retaining the specialization to the dataset's domain.

One interesting observation is that Cancels suggests adding compounds on the outer ranges of PubChem rather than its center. Sampling new compounds randomly would result in a distribution shift towards that of PubChem and the dataset would lose its focus on the domain for which it is designed.

Note that the indicated areas focus on regions within the compound space due to the boundaries introduced in Section 6.4, so finding suitable compounds that mitigate this bias is possible.

Which kinds of compounds does Cancels suggest to mitigate the bias? To fill in the underrepresented regions identified in the previous experiment, we offer Cancels a pool of compounds to choose from. This pool is assembled from those compounds in the PubChem database that carry an 'agrochemical' flag. The reduction to this subset was necessary to enable us to extract the same auxiliary information from the pool data that is already available for the SOIL and BBD datasets. Figure 6.7 displays the frequency of relevant, non-exclusive labels for the entire pool (in gray) as well as the input dataset (SOIL in blue, BBD in wine) and the top 20 and top 50 candidate compounds to mitigate the bias.

We observe a shift towards fungicides and herbicides for SOIL and biocides and fungicides for BBD in the recommendations for both datasets. This is a meaningful result since both categories are under-represented in the datasets by design, but they seem relevant to add as they are structurally similar in order to train models on the datasets. Comparison with the entire pool shows that Cancels specifically targets compounds belonging to these categories – they do not reflect a general trend of the pool. Note that these results have been obtained although Cancels was never presented with these categories but only the MACCS representations of compounds.

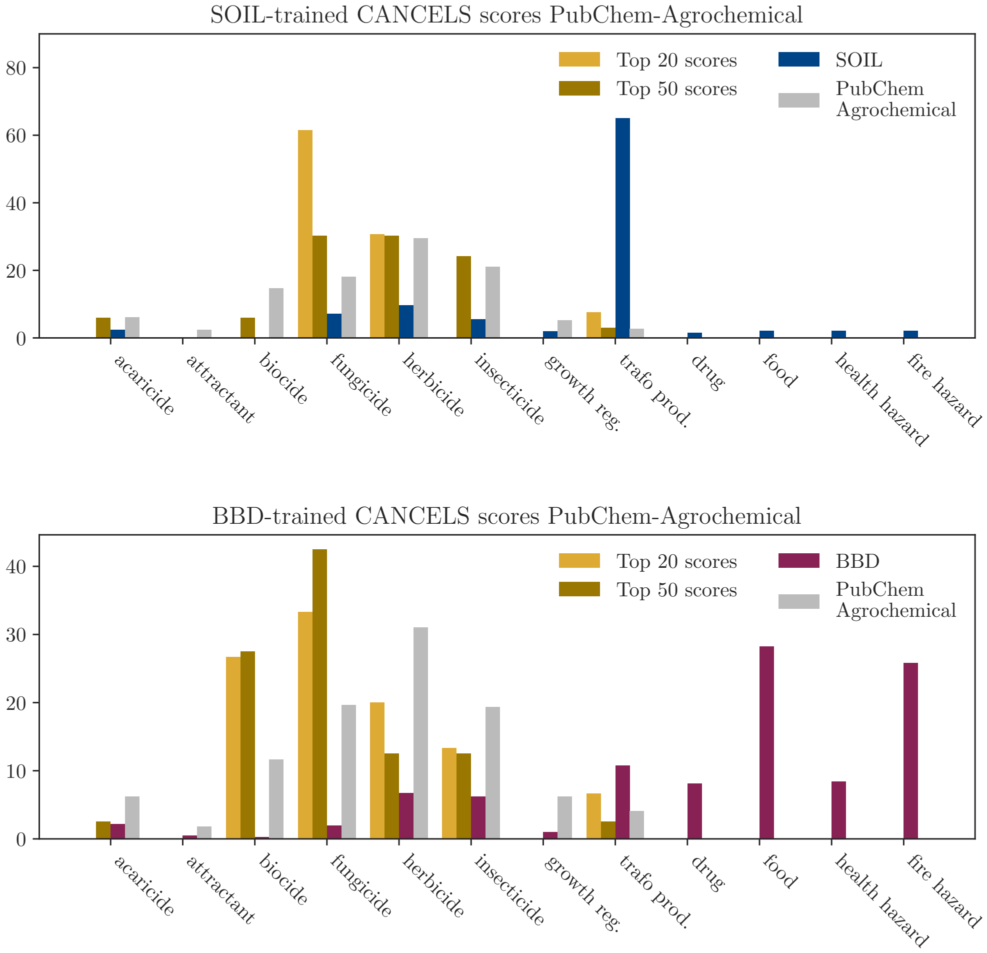


Figure 6.7: Qualitative evaluation of the top 20 and top 50 compounds suggested by Cancels to mitigate the detected biases in SOIL (top) and BBD (bottom) in comparison to the respective dataset's compounds and the agrochemical subset of PubChem. Note that categories are non-exclusive.

Cross-Check: Does Cancels perform as expected? To cross-check that Cancels is working as intended, we carry out an additional experiment. Training a kernel density estimator to model the dataset's density, we sort all compounds by their assigned densities. Holding out the $x\%$ of the dataset with the lowest density, we use Cancels on the rest and score the held-out compounds. Intuitively, removing data from a dataset should reduce its quality and result in high scores for the removed data aiming to retrieve the original dataset quality.

The results are shown in Figure 6.8. We see that for low percentages x, the scores are generally low. This is expected since outliers will be removed

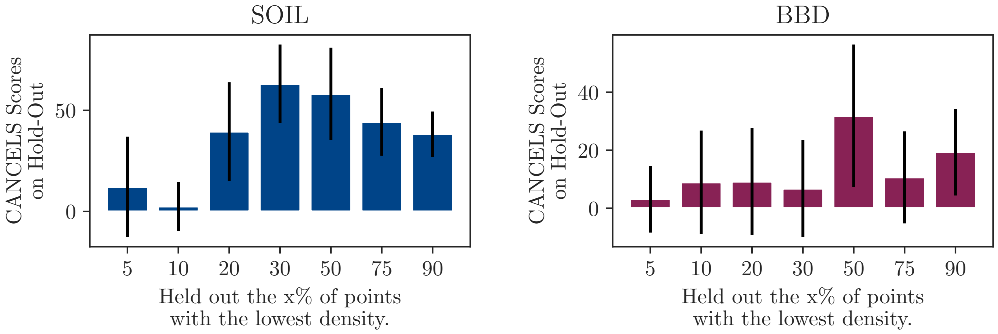


Figure 6.8: While holding out $x\%$ of the SOIL (left) and BBD (right) datasets, we train Cancels on the rest. Bar heights represent average scores of the holdout set with their corresponding uncertainty intervals (black lines).

first and cannot be expected to score highly. For high x, the average scores are decreasing again. This is also expected since Cancels is applied to a very small portion of the dataset only and, by design, makes conservative estimates resulting in high scores only for some of the removed compounds. The peak is at $x = 50\%$ where both effects are minimal. Overall, Cancels' general behavior fits our expectations.

We notice a few irregularities in the patterns deviating from a smooth ascent to and descent from the $x = 50\%$ peak. These irregularities stem from a change in the underrepresented area Cancels points to and are an indication of a bias in the dataset: If the dataset was smooth and unbiased, removing those $x\%$ of compounds with the lowest density would narrow the dataset to its center (or, if there are multiple clusters, to their centers) equally from all sides. In this case, the estimated Gaussian would stay consistent over all $x \leq 50\%$ and potentially even for higher ones. Hence, since we observed jumps, we can conclude that a bias must be present even from this perspective.

Can Cancels improve the model performance? To assess the relevance of the compounds suggested by Cancels, we use the Tox21 dataset (see Section 6.5.1) due to its size and set up an experiment as follows: In each of 100 runs, we randomly hold out 40% of the dataset as a test set, offer 40% of the remaining data as a pool and use the rest for training. Based on the training set, we select additional compounds from the pool in four different

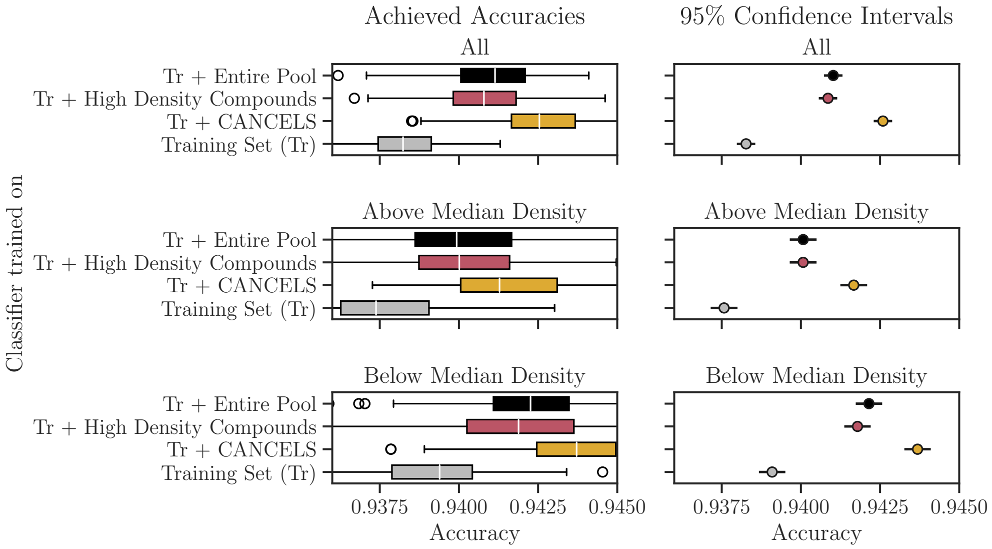


Figure 6.9: Dividing the Tox21 dataset into a training set, a pool, and a test set, we train a classifier on either the training set only, the training set together with the entire pool, the training set plus Cancels-based compound selection, and the training set plus a selection that feeds the biases instead of mitigating it. The box plot (left) displays the results in terms of accuracy when evaluating the trained models on the test set. A confidence interval plot (right) indicates that compound selection using Cancels is significantly better than all other options.

scenarios: We can select (i) no additional compounds, (ii) n_{Cancels} compounds suggested by Cancels, (iii) n_{Cancels} compounds that feed rather than mitigate the bias based on density-based random sampling (i.e., we sample based on the dataset distribution directly), or (iv) all available additional compounds (i.e., the entire pool).

A classifier is then trained on the training set together with each selection of additional compounds and evaluated on the test set.

Figure 6.9 shows that compound selection using Cancels is not only better than continuing to feed the bias but also than using the entire pool. A repeated measures ANOVA with posthoc Tukey HSD test [36, 64] confirms that these results are statistically significant under significance level $\alpha = 0.01$.

However, the accuracy differences are small. We attribute this effect to the experimental design: Since we had no additional dataset with the same labels available, we had to divide the Tox21 data into a training set, a test set, and a pool. This places Cancels in a particularly difficult situation.

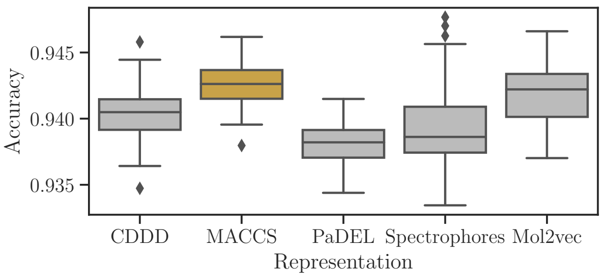


Figure 6.10: Influence of different compound representations on Cancels' performance.

First, the pool is equally biased and hence does not contain the compounds required to correct the bias beyond the data domain. At most, it can rebalance parts of the space. Second, the test set also is biased. Therefore, even if a bias is mitigated, it will not pay off as those parts of the data space for which it matters are also underrepresented in the test set. Despite these difficulties, we observe an improvement in accuracy for the Cancels-selected compounds, which is quite remarkable.

Splitting the test dataset along the compounds' median density reveals that this effect is particularly strong in the low-density areas. This is an essential result since it supports the exploration of the space that breaks the bias spiral and has the potential to lead to global rather than local optimization.

How does the compound representation affect the performance? Using a MACCS fingerprint as a compound's feature representation for training a model is widely popular [135] due to the computational speed and the solid performance in different applications. However, Cancels' compound feature representation is independent of that used by the model. To investigate which representation performs best in Cancels, we repeat the previous experiment with the following competitors to MACCS fingerprints: (i) *Continuous data-driven descriptors (CDDD)* [153] obtained from an RNN autoencoder, (ii) *PaDEL* [155], a set of 1875 2D and 3D molecular properties, (iii) *Spectrophores* [55] calculated from 3D properties of molecules using affinity cages, and (iv) *Mol2vec* [72], a neural network-based embedding similar to the word2vec models used in natural language processing trained to embed structures co-appearing frequently near each other in latent space.

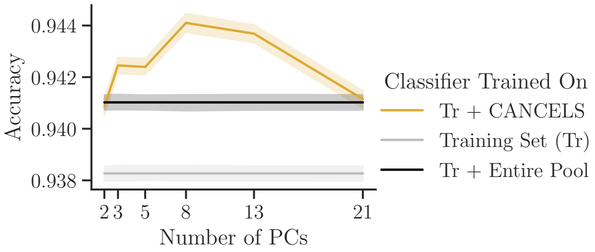


Figure 6.11: Influence of the number of principal components used in Cancels' dimensionality reduction on Tox21.

For all competitors, we obtained the pre-processed datasets from Stepišnik *et al.* [135].

Figure 6.10 illustrates the results: The differences between representations are small. MACCS and Mol2vec perform slightly better than the rest, and MACCS fingerprints additionally show a smaller variance among runs. Ultimately, the right choice of feature representation depends on the application and should be investigated individually, but in our use case, using MACCS fingerprints for Cancels seems well justified.

How does the number of principal components influence the performance? Choosing the correct number of principal components for PCA in an unsupervised setting is difficult since we have no feedback on which number performs best. Intuitively (and following the central limit theorem), the smaller the number n_{PC} of principal components, the more closely our dataset distribution will resemble a Gaussian as more individual signals are combined. At the same time, the higher n_{PC}, the more variance in the dataset we can explain using the components. That is, a dataset can be modeled perfectly if its dimensionality matches n_{PC}, but information will be lost if the dimensionality is reduced. We can see both aspects in Figure 6.11 where there is a peak around $n_{\mathrm{PC}} = 8$ indicating that the results presented here (with $n_{\mathrm{PC}} = 5$) could have been better, but our estimated value is reasonable. To choose a suitable value for n_{PC}, as a rule of thumb, we suggest trialing different values and visualizing the dataset distribution over the resulting components. A solid choice is the largest value that shows Gaussian-like distributions over all components. In future research, we will investigate how to choose n_{PC} automatically.

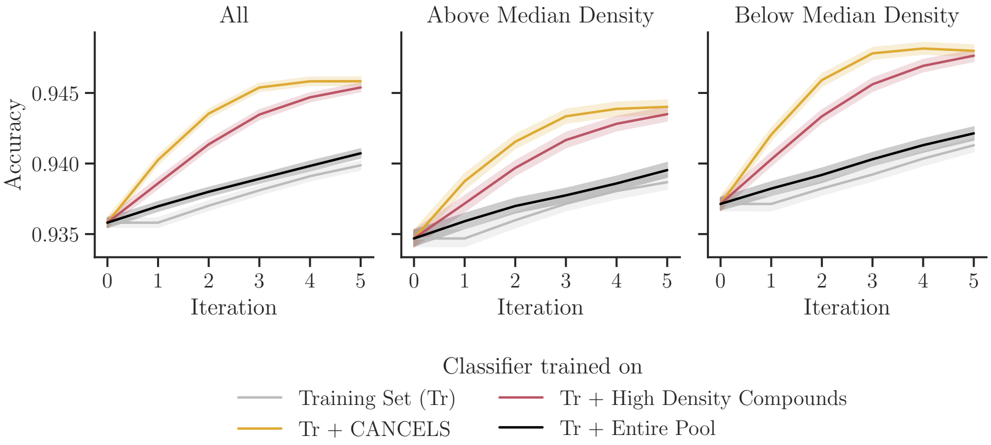


Figure 6.12: Iterative application of Cancels and all competing baselines (see Fig. 6.9) on the Tox21 dataset: In each of the five iterations, the compound selection takes place based on the training set and the selected compounds from previous iterations. For Cancels, the accuracy improves upon all other selection strategies.

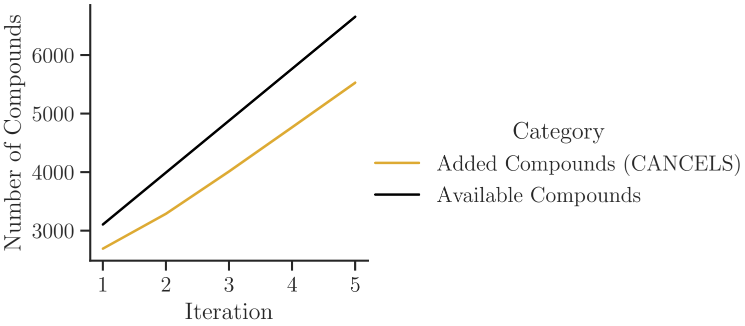


Figure 6.13: Number of added compounds in an iterative application of Cancels.

Can iterative application of Cancels improve the accuracy even further? The previous experiments showed an improvement in accuracy for Cancels-based compound selection, especially in lower-density areas of the data space. To investigate the long-term effect, we carry out a similar but iterative experiment where we randomly split the pool into five equally-sized sub-pools. In each of five iterations, we select additional compounds from the corresponding sub-pool based on the training set and the selections from all previous iterations. Note that, as before, we select the same number of points for both Cancels-based sampling and sampling based on the data density in every iteration to ensure a fair comparison.

Figure 6.12 and 6.13 summarize the impact of Cancels on each of the iterations. Firstly, we observe that three iterations seem sufficient to smooth out the dataset distribution. Additional iterations have no effect, and the accuracy is saturated. After three iterations, CANCELS has selected only about 4000 compounds and still largely outperforms the entire pool with about 7000 compounds. The red line ("Tr + High Density Compounds") stands for training on the training set together with a random sample from the pool. Since the pool follows the same distribution as the dataset, sampling from it will mostly result in compounds in dense areas, but few compounds from sparse areas can also find their way in, so the red line eventually catches up with CANCELS. This effect is an anomaly due to our experimental design and will no longer be observed if the pool's distribution does not match that of the dataset and the test set. In summary, selecting the right compounds not only improves the data quality but also is substantially more economical as it means carrying out fewer experiments.

In practice, improving the dataset quality is not the only goal – a researcher also aims to make decisions regarding their data collection based on their current interests, projects, and goals. To achieve a healthy balance, we suggest one or two iterations of CANCELS after each interest-driven addition to the dataset before the dataset is fit for its upcoming tasks.

6.6 Conclusion

Predictive modeling can support the development process of new chemicals; however, those models specialize to the data provided, and solid performance can only be guaranteed in densely populated areas of the compound space. Avoiding carrying out experiments with a very uncertain result, new additions to the dataset will most likely stem from already densely populated areas where the prediction reliability is high. Over the years, this results in a stronger over-population of already over-populated areas and a shrinking applicability domain of trained models inducing a specialization bias.

To break this spiraling specialization cycle, in this paper, we propose CANCELS, a novel technique to investigate a dataset independently from a specific model, create awareness of underrepresented areas, and suggest additional compounds that can help mitigate the bias. So far, CANCELS is unique in many regards: (i) It generally improves the dataset quality in a model-independent fashion while other methods are only designed to support the training process of one specific model, (ii) while generalizing the dataset and enabling further targeted exploration of the compound space, Cancels does not lose the desired specialization to a certain domain when suggesting additional compounds, and (iii) Cancels' outputs are interpretable and can be used to investigate different aspects of datasets as demonstrated in our extensive set of experiments.

Our various experiments indicate that on two real-world datasets, SOIL and BBD, a continuous specialization can indeed be observed, which renders these datasets a valid use-case for Cancels. Validation of Cancels on the Tox21 dataset shows that careful selection of future experiments can not only reduce the total amount of experiments to be carried out but also improve the performance of predictive models by a significant margin.

All results presented in this paper have been obtained based solely on the compounds' MACCS keys. Future research will investigate how auxiliary information can be integrated in an effective way where available. Additionally, we aim to make Cancels fully automated for the simplest usage possible. As such, we aim to automatically infer parameters such as the number of principal components from the dataset and context, for example, using information criteria that incorporate a measure of Gaussianity but penalize for every dimension lost. Overall, we hope that Cancels can be of use to researchers to help understand the datasets they are dealing with and to improve their quality early on to improve their usability universally.

7 Conclusion

We are at a point in time where we begin to trust the outputs of our machine-learning models blindly, even in high-risk applications. However, the models are only as good as the data they are trained on, and dataset flaws creep into the models silently. This is mainly due to the standard model validation processes where training and validation occur on subsets of the same dataset. If the training set is biased, the validation set is equally biased, and the model will not send a warning.

The bias mitigation literature detects biases by comparing training and target data, but no approaches have been proposed that can send off an early warning based on the training set alone. To the best of our knowledge, we made the very first attempt to address the problem of selection bias detection when no ground-truth information is available.

We proposed three methods with different strengths that investigate the dataset's distribution and generate or select additional data points to smooth it out. If these points concentrate on several areas, this could indicate a potential bias.

In various experiments, we demonstrated the usefulness of the methods we proposed here. However, our most impactful contribution lies in challenging existing machine learning procedures that accept flawed data as given and treat symptoms rather than causes.

In this chapter, we review our achievements and contributions in Section 7.1, discuss limitations in Section 7.2 and provide potential avenues for future research, beyond improving the proposed models, in Section 7.3.

7.1 Contributions

In the previous chapters, we established the novel problem of selection bias identification and mitigation under the assumption that no ground-truth information is available. Since no knowledge of the ground truth can be expected, this problem is particularly challenging and likely to be impossible to solve in a general way that works for all biases and datasets. Nonetheless, we proposed three methods showcasing that some biases leave traces that can be observed in the resulting datasets.

We introduced techniques that are able to detect these traces and send off an early warning, i.e., already during data gathering. The early bias detection allows us to improve the data collection on the fly, avoiding the need for costly and fragile adaptation methods later on.

Our methods show remarkable results given our uninformed "guess": Even if the data sample correctly reflects the ground truth, our methods point to potential issues for subsequent model training based on this data, such as underrepresented areas. Filling in these identified gaps with generated data points or ones selected from a pool has been shown to improve the training behavior of machine learning models.

Designing our problem in a completely uninformed way constitutes a universally applicable preprocessing method that can be integrated into most machine learning pipelines. While existing bias mitigation strategies are typically tailored to one specific target domain or even a specific model, we operate independently of both. Instead, we improve the overall data quality and create awareness for potential issues, regardless of the data's future journey.

Our three proposed methods cater to different needs: Imitate (*Identify and MITigATE Selection Bias*) assumes that the ground truth could be modeled as one multivariate Gaussian per class. While this is a strong assumption that might not hold in complex datasets, Imitate convinces by being fast, interpretable, and expandable. Mimic (*Multi-IMItate Bias Correction*) expands Imitate's scope by modeling the ground truth as a mixture of potentially overlapping multivariate Gaussians per class. This assumption drastically increases the range of datasets and distributions that can be modeled, including multi-cluster settings. However, it comes at a cost in terms of running time and produces more conservative results. Cancels (*CounterActiNg Compound spEciaLization biaS*) is a version of the Imitate algorithm, specialized to chemical compound datasets, where we showcased the usefulness and interpretability of our methods. Using Cancels, we demonstrated that when adding bias-mitigating compounds from a pool of candidates to a dataset, the predictive performance of a trained model exceeds that of a model trained either on the original dataset or under the addition of the entire pool. Finally, we contribute publicly available, and easy-to-use Python+sklearn [110] implementations of all methods in the PyPI package imitatebias. Cancels will additionally be integrated into the enviPath website[1], where users can trial their datasets freely via a web interface.

7.2 Limitations

Although our work marks a promising start to a new area of research, our proposed methods should be expanded to be fully sufficient. We identify the following main limitations:

- Although our proposed methods have proven effective on biased datasets and behave differently when no bias is present, we are dealing with an under-defined problem. Only the biased dataset is available, with no information on the ground truth or its distribution. We successfully constrained the problem more tightly using the assumption of Gaussianity or a Gaussian mixture; however, this assumption might not always hold. Therefore, we aim to create awareness of a potential issue but ultimately rely on a domain expert's knowledge to decide whether the result is valid.
- Modeling the ground truth as a multivariate Gaussian is a rigid requirement on the data structure and limits the applicability of our method. Allowing for mixtures of Gaussians facilitates modeling more

[1] enviPath: https://envipath.org/

general data distributions. However, this generality comes at a cost: The more flexible the fitted model is, the more apt it is to model the existing data well instead of pointing to a bias, which is why Mimic's results are more conservative than Imitate's. Similarly, in preliminary experiments, we tried fitting a beta distribution instead – a family of distributions that is substantially more general as it combines exponential, normal, uniform, and gamma distributions. We observed that for all datasets, biased or not, parameters that fit the data very closely could be identified. Hence, there is a trade-off between the generality of the method and the clarity of the obtained results.

- For some datasets, for example, exponentially distributed ones, a Gaussian is a poor fit, and even a mixture of Gaussians will not model the data appropriately. Our current solution to this is to reject any output if the number of points required to smooth out the distribution exceeds the number of observed data points. In addition, an initial test to decide if our methods are applicable to the data would be helpful. One easily integrable option would be to fit different probability densities to the density estimation of the data, weighted by their histogram bin heights, as done in Imitate. If a different distribution is a substantially closer fit than a Gaussian, it could substitute the normal density before proceeding as usual. Note that a different data transformation (or none at all) might be required for this test as ICA is closely tied to the normal distribution.

7.3 Outlook

Lifting the previously identified limitations would be a direct continuation of this work. Beyond that, we would like to draw the reader's attention to the following interesting potential research avenues:

- Under our supervision, Duncanson [43] adjusted Imitate to learn a parametric representation of the selection bias and the Gaussian simultaneously using maximum likelihood optimization instead of the heuristic weighted optimization with histograms. Duncanson tested step functions, Fourier series, polynomial, and piecewise linear functions for the one-dimensional case. Except for the step function, these bias representations quickly become complex if the number of dimensions increases, with an overwhelming number of parameters to be tuned. However, the step function showed promising results in a few exemplary tests and should be investigated further in the future, particularly concerning scalability in higher dimensions.

- *Adversarial Learning* [15] is a research area that aims to make machine learning models more robust by exposing their vulnerabilities early on. To find these vulnerabilities, adversarial learning uses adversarial attacks on a model, carefully crafted small perturbations of the input data that are powerful enough to alter the model prediction. Adversarial learning and the research we presented share the goal of exposing model vulnerabilities and making a model more robust. However, under our supervision, La *et al.* [86] found that both approaches identify different flaws in the dataset. Future research could explore similarities and differences further and identify potential synergies that make a meaningful advance in either field.

- In many applications, such as drug development in chemistry, running experiments is costly and time-consuming. Therefore, the experiments to be run need to be chosen carefully. *Active learning* [127] is a research area that aims to identify those examples that, when labeled and added to the dataset, help improve the model the most. Since this is a substantially different goal from that in our thesis (as discussed in Section 6.3), active learning is unaware of biases and does not attempt to correct them. Active learning typically selects instances near the decision boundary to improve one specific model, but it does not operate on the dataset itself. However, a symbiosis between our research and active learning could sustainably improve the data quality while remaining purpose-driven and optimizing the model's performance.

- Lastly, we hope this research ignites a spark in the community that leads to a deeper investigation of the data we gather. Flawed data causes a flawed model causes flawed decisions, and it is not sufficient to accept the data as given without questioning it. We further hope that a large body of research supersedes this work, ultimately leading to automated data quality assurance that can become a fixed component in the machine learning development pipeline and the data-gathering process.

Bibliography

[1] N. Abreu, “Análise do perfil do cliente Recheio e desenvolvimento de um sistema promocional,” Portuguese, Master’s Thesis, University Institute of Lisbon, Portugal, 2011.

[2] J. A. Adebayo, “FairML: Toolbox for diagnosing bias in predictive modeling,” Master’s Thesis, Massachusetts Institute of Technology, MA, USA, 2016.

[3] R. Alaiz-Rodríguez and N. Japkowicz, “Assessing the impact of changing environments on classifier performance,” in *Advances in Artificial Intelligence*, Springer Berlin Heidelberg, 2008, pp. 13–24. DOI: 10.1007/978-3-540-68825-9_2.

[4] J. Angwin, J. Larson, S. Mattu, and L. Kirchner, *Machine bias: Risk assessments in criminal sentencing*, propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing, ProPublica, May 2016, accessed: 2022-11-18.

[5] N. Aniceto, A. A. Freitas, A. Bender, and T. Ghafourian, “A novel applicability domain technique for mapping predictive reliability across the chemical space of a QSAR: Reliability-density neighbourhood,” *Journal of Cheminformatics*, vol. 8, no. 69, 2016, Springer International. DOI: 10.1186/s13321-016-0182-y.

[6] A. Arnold, R. Nallapati, and W. Cohen, “A comparative study of methods for transductive transfer learning,” in *Seventh IEEE International Conference on Data Mining Workshops (ICDMW ’07)*, IEEE, 2007, pp. 77–82. DOI: 10.1109/ICDMW.2007.109.

[7] J. Arús-Pous, T. Blaschke, S. Ulander, J. L. Reymond, H. Chen, and O. Engkvist, "Exploring the GDB-13 chemical space using deep generative models," *Journal of Cheminformatics*, vol. 11, no. 20, 2019, Springer International. DOI: 10.1186/s13321-019-0341-z.

[8] N. Bantilan, "Themis-ml: A fairness-aware machine learning interface for end-to-end discrimination discovery and mitigation," *Journal of Technology in Human Services*, vol. 36, no. 1, pp. 15–30, 2018, Taylor & Francis. DOI: 10.1080/15228835.2017.1416512.

[9] E. Bareinboim, J. Tian, and J. Pearl, "Recovering from selection bias in causal and statistical inference," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, AAAI, 2014. DOI: 10.1609/aaai.v28i1.9074.

[10] E. B. Baum and K. Lang, "Query learning can work poorly when a human oracle is used," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, vol. 8, IEEE, 1992, pp. 335–340.

[11] R. K. E. Bellamy, K. Dey, M. Hind, S. C. Hoffman, S. Houde, K. Kannan, P. Lohia, J. Martino, S. Mehta, A. Mojsilović, S. Nagar, K. N. Ramamurthy, J. Richards, D. Saha, P. Sattigeri, M. Singh, K. R. Varshney, and Y. Zhang, "Ai fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias," *IBM Journal of Research and Development*, vol. 63, no. 4/5, 4:1–4:15, 2019, IBM. DOI: 10.1147/JRD.2019.2942287.

[12] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine Learning*, vol. 79, no. 1-2, pp. 151–175, 2010, Springer USA. DOI: 10.1007/s10994-009-5152-4.

[13] R. Bhatt and A. Dhall, *Skin segmentation dataset*, archive.ics.uci.edu/ml/datasets/skin+segmentation, UCI Machine Learning Repository, Jul. 2012, accessed: 2020-02-05.

[14] S. Bickel, M. Brückner, and T. Scheffer, "Discriminative learning for differing training and test distributions," in *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*, ACM, 2007, pp. 81–88. DOI: 10.1145/1273496.1273507.

[15] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recognition*, vol. 84, pp. 317–331, 2018, Elsevier. DOI: 10.1016/j.patcog.2018.07.023.

[16] G. Blanchard, G. Lee, and C. Scott, "Generalizing from several related classification tasks to a new unlabeled sample," in *Proceedings of the 24th International Conference on Neural Information Processing Systems*, vol. 24, Curran Associates Inc., 2011, pp. 2178–2186.

[17] R. S. Bohacek, C. McMartin, and W. C. Guida, "The art and practice of structure-based drug design: A molecular modeling perspective," *Medicinal Research Reviews*, vol. 16, no. 1, pp. 3–50, 1996, Wiley. DOI: 10.1002/(SICI)1098-1128(199601)16:1<3::AID-MED1>3.0.CO;2-6.

[18] D. Borland, W. Wang, J. Zhang, J. Shrestha, and D. Gotz, "Selection bias tracking and detailed subset comparison for high-dimensional data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 429–439, 2020, IEEE. DOI: 10.1109/TVCG.2019.2934209.

[19] D. Borland, J. Zhang, S. Kaul, and D. Gotz, "Selection-bias-corrected visualization via dynamic reweighting," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1481–1491, 2021, IEEE. DOI: 10.1109/TVCG.2020.3030455.

[20] D. Bourgeois, J. Rappaz, and K. Aberer, "Selection bias in news coverage: Learning it, fighting it," in *The Web Conference 2018 - Companion of the World Wide Web Conference, (WWW '18)*, ACM, 2018, pp. 535–543. DOI: 10.1145/3184558.3188724.

[21] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD '00)*, ACM, 2000, pp. 93–104. DOI: 10.1145/342009.335388.

[22] J. Burkardt, *The truncated normal distribution*, people.sc.fsu.edu/~jburkardt/ presentations/truncated_normal.pdf, Department of Scientific Computing, Florida State University, Oct. 2014, accessed: 2022-11-25.

[23] S. van Buuren and K. Groothuis-Oudshoorn, “mice: Multivariate imputation by Chained Equations in R,” *Journal of Statistical Software*, vol. 45, no. 3, pp. 1–67, 2011, FOAS. DOI: 10.18637/jss.v045.i03.

[24] A. Caliskan, J. J. Bryson, and A. Narayanan, “Semantics derived automatically from language corpora contain human-like biases,” *Science*, vol. 356, no. 6334, pp. 183–186, 2017. DOI: 10.1126/science.aal4230.

[25] F. P. Calmon, D. Wei, B. Vinzamuri, K. N. Ramamurthy, and K. R. Varshney, “Optimized pre-processing for discrimination prevention,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS ’17)*, Curran Associates Inc., 2017, pp. 3995–4004.

[26] R. Caruana, “Multitask learning,” *Machine Learning*, vol. 28, pp. 41–75, 1997, Springer USA. DOI: 10.1023/A:1007379606734.

[27] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, 2009, ACM. DOI: 10.1145/1541880.1541882.

[28] S. Chandra, A. Haque, L. Khan, and C. Aggarwal, “Efficient sampling-based kernel mean matching,” in *2016 IEEE 16th International Conference on Data Mining (ICDM ’16)*, IEEE, 2016, pp. 811–816. DOI: 10.1109/ICDM.2016.0095.

[29] C. W. Chiang and M. Yin, “You’d Better Stop! Understanding Human Reliance on Machine Learning Models under Covariate Shift,” in *13th ACM Web Science Conference 2021 (WebSci ’21)*, ACM, 2021, pp. 120–129. DOI: 10.1145/3447535.3462487.

[30] D. A. Cieslak and N. V. Chawla, “A framework for monitoring classifiers’ performance: When and why failure occurs?” *Knowledge and Information Systems*, vol. 18, pp. 83–108, 2009, Springer Berlin Heidelberg. DOI: 10.1007/s10115-008-0139-1.

[31] A. E. Cleves and A. N. Jain, “Effects of inductive bias on computational evaluations of ligand-based modeling and on drug discovery,” *Journal of Computer-Aided Molecular Design*, vol. 22, pp. 147–159, 2008, Springer Netherlands. DOI: 10.1007/s10822-007-9150-y.

[32] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, “Sample selection bias correction theory,” in *Algorithmic Learning Theory (ALT '08)*, ser. Lecture Notes in Computer Science, vol. 5254, Springer Berlin Heidelberg, 2008, pp. 38–53. DOI: 10.1007/978-3-540-87987-9_8.

[33] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, “Boosting for transfer learning,” in *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*, ACM, 2007, pp. 193–200. DOI: 10.1145/1273496.1273521.

[34] J. Dastin, *Amazon scraps secret ai recruiting tool that showed bias against women*, reuters.com/article/idUSKCN1MK08G, Reuters, Oct. 2018, accessed: 2022-11-18.

[35] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977, Wiley. DOI: 10.1111/j.2517-6161.1977.tb01600.x.

[36] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006, MIT Press.

[37] K. Dost, H. Duncanson, I. Ziogas, P. Riddle, and J. Wicker, “Divide and imitate: Multi-cluster identification and mitigation of selection bias,” in *Advances in Knowledge Discovery and Data Mining - 26th Pacific-Asia Conference (PAKDD '22)*, ser. Lecture Notes in Computer Science, vol. 13281, Springer Cham, 2022, pp. 149–160. DOI: 10.1007/978-3-031-05936-0_12.

[38] K. Dost, Z. Pullar-Strecker, L. Brydon, K. Zhang, J. Hafner, P. Riddle, and J. Wicker, “Combatting over-specialization bias in growing chemical databases,” *Journal of Cheminformatics*, vol. 15, no. 53, 2023. DOI: 10.1186/s13321-023-00716-w.

[39] K. Dost, K. Taskova, P. Riddle, and J. Wicker, “Your best guess when you know nothing: Identification and mitigation of selection bias,” in *20th IEEE International Conference on Data Mining (ICDM 2020)*, IEEE, 2020, pp. 996–1001. DOI: 10.1109/ICDM50108.2020.00115.

[40] J. Dressel and H. Farid, “The accuracy, fairness, and limits of predicting recidivism,” *Science Advances*, vol. 4, no. 1, 2018, AAAS. DOI: 10.1126/sciadv.aao5580.

[41] D. Dua and C. Graff, *UCI machine learning repository*, archive.ics.uci.edu/ml, University of California, Irvine, School of Information and Computer Sciences, 2017.

[42] M. Dudík, S. Phillips, and R. E. Schapire, “Correcting sample selection bias in maximum entropy density estimation,” in *Proceedings of the 18th International Conference on Neural Information Processing Systems (NIPS ’05)*, vol. 18, MIT Press, 2005, pp. 323–330.

[43] H. Duncanson, “Identification and mitigation of selection bias: Improvements, extensions and experiments,” Bachelor of Science (Hons) Dissertation, University of Auckland, New Zealand, 2021.

[44] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, “Fairness through awareness,” in *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS ’12)*, ACM, 2012, pp. 214–226. DOI: 10.1145/2090236.2090255.

[45] J. E. van Engelen and H. H. Hoos, “A survey on semi-supervised learning,” *Machine Learning*, vol. 109, no. 2, pp. 373–440, 2020, Springer USA. DOI: 10.1007/s10994-019-05855-6.

[46] A. Farahani, S. Voghoei, K. Rasheed, and H. R. Arabnia, “A brief review of domain adaptation,” preprint, arXiv, 2020. DOI: 10.48550/ARXIV.2010.03978.

[47] T. Fawcett and P. A. Flach, “A response to Webb and Ting’s on the application of roc analysis to predict classification performance under varying class distributions,” *Machine Learning*, vol. 58, pp. 33–38, 2005, Springer USA. DOI: 10.1007/s10994-005-5256-4.

[48] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian, “Certifying and Removing Disparate Impact,” in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ’15)*, ACM, 2015, pp. 259–268. DOI: 10.1145/2783258.2783311.

[49] S. A. Friedler, C. Scheidegger, and S. Venkatasubramanian, “The (im)possibility of fairness: Different value systems require different mechanisms for fair decision making,” *Communications of the ACM*, vol. 64, no. 4, pp. 136–143, 2021, ACM. DOI: 10.1145/3433949.

[50] S. A. Friedler, C. Scheidegger, S. Venkatasubramanian, S. Choudhary, E. P. Hamilton, and D. Roth, “A comparative study of fairness-enhancing interventions in machine learning,” in *Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT* ’19)*, ACM, 2019, pp. 329–338. DOI: 10.1145/3287560.3287589.

[51] G. P. C. Fung, J. Yu, H. Lu, and P. Yu, “Text classification without negative examples revisit,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 1, pp. 6–20, 2006, IEEE. DOI: 10.1109/TKDE.2006.16.

[52] P. Gajane and M. Pechenizkiy, “On formalizing fairness in prediction with machine learning,” preprint, arXiv, 2017. DOI: 10.48550/ARXIV.1710.03184.

[53] S. Galhotra, Y. Brun, and A. Meliou, “Fairness testing: Testing software for discrimination,” in *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering (ESEC/FSE ’17)*, ACM, 2017, pp. 498–510. DOI: 10.1145/3106237.3106277.

[54] T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. Wallach, H. D. III, and K. Crawford, “Datasheets for datasets,” *Communications of the ACM*, vol. 64, no. 12, pp. 86–92, 2021, ACM. DOI: 10.1145/3458723.

[55] R. Gladysz, F. M. Dos Santos, W. Langenaeker, G. Thijs, K. Augustyns, and H. De Winter, "Spectrophores as one-dimensional descriptors calculated from three-dimensional atomic properties: Applications ranging from scaffold hopping to multi-target virtual screening," *Journal of Cheminformatics*, vol. 10, no. 9, 2018, Springer International. DOI: 10.1186/s13321-018-0268-9.

[56] N. Goel, M. Yaghini, and B. Faltings, "Non-discriminatory machine learning through convex fairness criteria," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, AAAI, 2018. DOI: 10.1609/aaai.v32i1.11662.

[57] O. Granichin, Z. Volkovich, and D. Toledano-Kitai, "Cluster validation," in *Randomized Algorithms in Automatic Control and Data Mining*, ser. Intelligent Systems Reference Library. Springer Berlin Heidelberg, 2015, vol. 67, pp. 163–228. DOI: 10.1007/978-3-642-54786-7_7.

[58] E. Gregori-Puigjané and J. Mestres, "Coverage and bias in chemical library design," *Current Opinion in Chemical Biology*, vol. 12, no. 3, pp. 359–365, 2008, Elsevier. DOI: 10.1016/j.cbpa.2008.03.015.

[59] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, "Covariate Shift by Kernel Mean Matching," in *Dataset Shift in Machine Learning*, MIT Press, 2013, pp. 131–160. DOI: 10.7551/mitpress/9780262170055.003.0008.

[60] A. Habib Polash, T. Nakano, C. Rakers, S. Takeda, and J. Brown, "Active learning efficiently converges on rational limits of toxicity prediction and identifies patterns for molecule design," *Computational Toxicology*, vol. 15, 2020, Elsevier. DOI: 10.1016/j.comtox.2020.100129.

[61] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*, 3rd ed. Elsevier, 2012.

[62] E. Hargittai and G. Karaoglu, "Biases of online political polls: Who participates?" *Socius*, vol. 4, 2018, ASA. DOI: 10.1177/2378023118791080.

[63] B. Hassani, "Societal bias reinforcement through machine learning: A credit scoring perspective," *AI and Ethics*, vol. 1, pp. 1–9, 2020, Springer Nature. DOI: 10.1007/s43681-020-00026-z.

[64] S. Herbold, "Autorank: A python package for automated ranking of classifiers," *Journal of Open Source Software*, vol. 5, no. 48, p. 2173, 2020, The Open Journal. DOI: 10.21105/joss.02173.

[65] J. Hert, J. J. Irwin, C. Laggner, M. J. Keiser, and B. K. Shoichet, "Quantifying biogenic bias in screening libraries," *Nature Chemical Biology*, vol. 5, pp. 479–483, 2009, Springer Nature. DOI: 10.1038/nchembio.180.

[66] W. Hoeffding and H. Robbins, "The central limit theorem for dependent random variables," *Duke Mathematical Journal*, vol. 15, no. 3, pp. 773–780, 1948, Duke University Press. DOI: 10.1215/S0012-7094-48-01568-3.

[67] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Schölkopf, "Correcting sample selection bias by unlabeled data," in *Advances in Neural Information Processing Systems 19 (NIPS '06)*, vol. 19, MIT Press, 2007, pp. 601–608.

[68] R. Huang, M. Xia, D.-T. Nguyen, T. Zhao, S. Sakamuru, J. Zhao, S. A. Shahane, A. Rossoshek, and A. Simeonov, "Tox21challenge to build predictive models of nuclear receptor and stress response pathways as mediated by exposure to environmental chemicals and drugs," *Frontiers in Environmental Science*, vol. 3, 2016, Frontiers Media S.A. DOI: 10.3389/fenvs.2015.00085.

[69] Z. Huang, H. Wang, E. P. Xing, and D. Huang, "Self-challenging improves cross-domain generalization," in *Computer Vision (ECCV 2020)*, ser. Lecture Notes in Computer Science, vol. 12347, Springer Cham, 2020, pp. 124–140. DOI: 10.1007/978-3-030-58536-5_8.

[70] A. Hyvärinen and E. Oja, "Independent component analysis: Algorithms and applications," *Neural Networks*, vol. 13, no. 4-5, pp. 411–430, 2000, Elsevier. DOI: 10.1016/S0893-6080(00)00026-5.

[71] G. Idakwo, S. Thangapandian, J. Luttrell, Y. Li, N. Wang, Z. Zhou, H. Hong, B. Yang, C. Zhang, and P. Gong, "Structure–activity relationship-based chemical classification of highly imbalanced Tox21 datasets," *Journal of Cheminformatics*, vol. 12, no. 66, pp. 1–19, 2020, Springer International. DOI: 10.1186/s13321-020-00468-x.

[72] S. Jaeger, S. Fulle, and S. Turk, "Mol2vec: Unsupervised machine learning approach with chemical intuition," *Journal of Chemical Information and Modeling*, vol. 58, no. 1, pp. 27–35, 2018, ACS. DOI: 10.1021/acs.jcim.7b00616.

[73] X. Jia, A. Lynch, Y. Huang, M. Danielson, I. Lang'at, A. Milder, A. E. Ruby, H. Wang, S. A. Friedler, A. J. Norquist, and J. Schrier, "Anthropogenic biases in chemical reaction data hinder exploratory inorganic synthesis," *Nature*, vol. 573, no. 7773, pp. 251–255, 2019, Springer USA. DOI: 10.1038/s41586-019-1540-5.

[74] I. T. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical transactions. Series A: Mathematical, physical, and engineering sciences*, vol. 374, no. 2065, 2016, Royal Society. DOI: 10.1098/rsta.2015.0202.

[75] F. Kamiran and T. Calders, "Data preprocessing techniques for classification without discrimination," vol. 33, pp. 1–33, 2012, Springer Berlin Heidelberg. DOI: 10.1007/s10115-011-0463-8.

[76] S. G. Kang, J. A. Morrone, J. K. Weber, and W. D. Cornell, "Analysis of Training and Seed Bias in Small Molecules Generated with a Conditional Graph-Based Variational Autoencoder – Insights for Practical AI-Driven Molecule Generation," *Journal of Chemical Information and Modeling*, vol. 62, no. 4, pp. 801–816, 2022, ACS. DOI: 10.1021/acs.jcim.1c01545.

[77] H. Kaur, H. S. Pannu, and A. K. Malhi, "A systematic review on imbalanced data challenges in machine learning: Applications and solutions," *ACM Computing Surveys*, vol. 52, no. 4, 2019, ACM. DOI: 10.1145/3343440.

[78] M. Kearns, S. Neel, A. Roth, and Z. S. Wu, “Preventing fairness gerrymandering: Auditing and learning for subgroup fairness,” in *Proceedings of the 35th International Conference on Machine Learning (ICML ’35)*, J. Dy and A. Krause, Eds., vol. 6, IMLS, 2018, pp. 4008–4016.

[79] ——, “An empirical study of rich subgroup fairness for machine learning,” in *Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT* ’19)*, ACM, 2019, pp. 100–109. DOI: 10.1145/3287560.3287592.

[80] A. Kerstjens and H. De Winter, “LEADD: Lamarckian evolutionary algorithm for de novo drug design,” *Journal of Cheminformatics*, vol. 14, no. 3, pp. 1–20, 2022, Springer International. DOI: 10.1186/s13321-022-00582-y.

[81] S. Kim, J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B. A. Shoemaker, P. A. Thiessen, B. Yu, L. Zaslavsky, J. Zhang, and E. E. Bolton, “PubChem in 2021: new data content and improved web interfaces,” *Nucleic Acids Research*, vol. 49, no. D1, pp. D1388–D1395, 2020, Oxford University Press. DOI: 10.1093/nar/gkaa971.

[82] W. M. Kouw and M. Loog, “A review of domain adaptation without target labels,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 3, pp. 766–785, 2021, IEEE. DOI: 10.1109/TPAMI.2019.2945942.

[83] M. J. Kusner, J. Loftus, C. Russell, and R. Silva, “Counterfactual fairness,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc., 2017.

[84] H. Kuwahara and X. Gao, “Analysis of the effects of related fingerprints on molecular similarity using an eigenvalue entropy approach,” *Journal of Cheminformatics*, vol. 13, no. 27, pp. 1–12, 2021, Springer International. DOI: 10.1186/s13321-021-00506-2.

[85] Y. Kwon and J. Lee, “MolFinder: an evolutionary algorithm for the global optimization of molecular properties and the extensive exploration of chemical space using SMILES,” *Journal of Cheminformatics*, vol. 13, no. 24, 2021, Springer International. DOI: 10.1186/s13321-021-00501-7.

[86] R. La, Y. Zhang, K. Dost, and J. Wicker, “Quantifying reliability using adversarial regions,” Summer Project Report, University of Auckland, New Zealand, 2021.

[87] D. Latino, J. Wicker, M. Gütlein, E. Schmid, S. Kramer, and K. Fenner, “Eawag-soil in envipath: A new resource for exploring regulatory pesticide soil biodegradation pathways and half-life data,” *Environmental Science: Process & Impact*, vol. 19, no. 3, pp. 449–464, 2017, The Royal Society of Chemistry. DOI: 10.1039/C6EM00697C.

[88] A. Lavalle, A. Maté, and J. Trujillo, “An approach to automatically detect and visualize bias in data analytics,” in *CEUR Workshop Proceedings*, ser. 22nd International Workshop On Design, Optimization, Languages and Analytical Processing of Big Data, vol. 2572, CEUR, 2020.

[89] Y. Lin, Y. Lee, and G. Wahba, “Support vector machines for classification in nonstandard situations,” *Machine Learning*, vol. 46, no. 1–3, pp. 191–202, 2002, Kluwer Academic Publishers. DOI: 10.1023/A:1012406528296.

[90] Z. C. Lipton, Y. Wang, and A. J. Smola, “Detecting and correcting for label shift with black box predictors,” in *Proceedings of the 35th International Conference on Machine Learning (ICML ’18)*, ser. Proceedings of Machine Learning Research, vol. 80, PMLR, 2018, pp. 3128–3136.

[91] R. J. A. Little and D. B. Rubin, *Statistical analysis with missing data.* Wiley, 2019. DOI: 10.1002/9781119482260.

[92] A. Liu and B. D. Ziebart, “Robust classification under sample selection bias,” in *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS ’14)*, Curran Associates Inc., 2014, pp. 37–45.

[93] M. Long, Z. Cao, J. Wang, and P. S. Yu, “Learning multiple tasks with multilinear relationship networks,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS ’17)*, Curran Associates Inc., 2017, pp. 1593–1602.

[94] A. Lyon, “Why are Normal Distributions Normal?” *British Journal for the Philosophy of Science*, vol. 65, no. 3, pp. 621–649, 2014, BJPS. DOI: 10.1093/bjps/axs046.

[95] A. Mayr, G. Klambauer, T. Unterthiner, and S. Hochreiter, “Deeptox: Toxicity prediction using deep learning,” *Frontiers in Environmental Science*, vol. 3, 2016, Frontiers Media S.A. DOI: 10.3389/fenvs.2015.00080.

[96] G. McGaughey, W. Walters, and B. Goldman, “Understanding covariate shift in model performance,” *F1000Research*, vol. 5, p. 597, 2016, Taylor & Francis. DOI: 10.12688/f1000research.8317.1.

[97] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, “A Survey on Bias and Fairness in Machine Learning,” *ACM Computing Surveys*, vol. 54, no. 6, 2021, ACM. DOI: 10.1145/3457607. eprint: 1908.09635.

[98] Y.-Q. Miao, A. K. Farahat, and M. S. Kamel, “Ensemble kernel mean matching,” in *2015 IEEE International Conference on Data Mining (ICDM ’15)*, IEEE, 2015, pp. 330–338. DOI: 10.1109/ICDM.2015.127.

[99] J. J. Moré, “The Levenberg-Marquardt Algorithm: Implementation and Theory,” in *Numerical Analysis*, Springer Berlin Heidelberg, 1978, pp. 105–116. DOI: 10.1007/bfb0067700.

[100] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera, “A unifying view on dataset shift in classification,” *Pattern Recognition*, vol. 45, no. 1, pp. 521–530, 2012, Elsevier. DOI: 10.1016/j.patcog.2011.06.019.

[101] V. D. Mouchlis, A. Afantitis, A. Serra, M. Fratello, A. G. Papadiamantis, V. Aidinis, I. Lynch, D. Greco, and G. Melagraki, "Advances in de novo drug design: From conventional to machine learning methods," *International Journal of Molecular Sciences*, vol. 22, no. 4, 2021, MDPI. DOI: 10.3390/ijms22041676.

[102] K. Muandet, D. Balduzzi, and B. Schölkopf, "Domain generalization via invariant feature representation," *Proceedings of the 30th International Conference on Machine Learning (ICML '13)*, vol. 28, pp. 10–18, 2013, JMLR.

[103] H. Al-Mubaid and S. Umair, "A new text categorization technique using distributional clustering and learning logic," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 9, pp. 1156–1165, 2006, IEEE. DOI: 10.1109/TKDE.2006.135.

[104] S. Niu, Y. Liu, J. Wang, and H. Song, "A Decade Survey of Transfer Learning (2010–2020)," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 2, pp. 151–166, 2021, IEEE. DOI: 10.1109/tai.2021.3054609.

[105] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. V. Dillon, B. Lakshminarayanan, and J. Snoek, "Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NIPS '19)*, Curran Associates Inc., 2019.

[106] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010, IEEE. DOI: 10.1109/TKDE.2009.191.

[107] T. Panch, H. Mattie, and R. Atun, "Artificial intelligence and algorithmic bias: Implications for health systems," *Journal of Global Health*, vol. 9, 2019. DOI: 10.7189/jogh.09.020318.

[108] S. Panigrahi, A. Nanda, and T. Swarnkar, "A Survey on Transfer Learning," *Smart Innovation, Systems and Technologies*, vol. 194, no. 10, pp. 781–789, 2021, IEEE. DOI: 10.1007/978-981-15-5971-6_83.

[109] Y. Pathak, P. Shukla, A. Tiwari, S. Stalin, and S. Singh, "Deep transfer learning based classification model for covid-19 disease," *IRBM*, vol. 43, no. 2, pp. 87–92, 2022, Elsevier. DOI: 10.1016/j.irbm.2020.05.003.

[110] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, and et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011, JMLR.

[111] T. Pereira, M. Abbasi, B. Ribeiro, and J. P. Arrais, "Diversity oriented Deep Reinforcement Learning for targeted molecule generation," *Journal of Cheminformatics*, vol. 13, no. 21, pp. 1–17, 2021, Springer International. DOI: 10.1186/s13321-021-00498-z.

[112] D. N. Perkins and G. Salomon, "Transfer of learning," in *International Encyclopedia of Education*, 2nd ed., Pergamon, 1992, pp. 6452–6457.

[113] P. E. Pfeiffer and D. A. Schum, *Introduction to applied probability.* Academic Press, 1973. DOI: 10.1016/C2013-0-11306-2.

[114] J. Poulos and R. Valle, "Missing data imputation for supervised learning," *Applied Artificial Intelligence*, vol. 32, no. 2, pp. 186–196, 2016, Taylor & Francis. DOI: 10.1080/08839514.2018.1448143.

[115] A. Pyae, *Fish market dataset*, kaggle.com/datasets/aungpyaeap/fish-market, pre-processed and exported from SAS OnDemand for Academics, kaggle, 2019, accessed: 2022-11-26.

[116] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning.* MIT Press, 2009.

[117] S. Rabanser, S. Günnemann, and Z. Lipton, "Failing loudly: An empirical study of methods for detecting dataset shift," in *Advances in Neural Information Processing Systems 32 (NIPS '19)*, Curran Associates Inc., 2019, pp. 1396–1408.

[118] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proceedings of the 24th International Conference on Machine Learning (ICML '07)*, ACM, 2007, pp. 759–766. DOI: 10.1145/1273496.1273592.

[119] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Machine Language*, vol. 85, no. 3, pp. 333–359, 2011. DOI: 10.1007/s10994-011-5256-5.

[120] D. Reker and G. Schneider, "Active-learning strategies in computer-assisted drug discovery," en, *Drug Discovery Today*, vol. 20, no. 4, pp. 458–465, 2015, Elsevier. DOI: 10.1016/j.drudis.2014.12.004.

[121] D. Reker, P. Schneider, G. Schneider, and J. Brown, "Active learning for computational chemogenomics," *Future Medicinal Chemistry*, vol. 9, no. 4, pp. 381–402, 2017, Future Science. DOI: 10.4155/fmc-2016-0197.

[122] F. Sahigara, D. Ballabio, R. Todeschini, and V. Consonni, "Defining a novel k-nearest neighbours approach to assess the applicability domain of a qsar model for reliable predictions," *Journal of Cheminformatics*, vol. 5, no. 27, p. 27, 2013, Springer International. DOI: 10.1186/1758-2946-5-27.

[123] P. Saleiro, B. Kuester, L. Hinkson, J. London, A. Stevens, A. Anisfeld, K. T. Rodolfa, and R. Ghani, *Aequitas: A bias and fairness audit toolkit*, preprint, 2018. DOI: 10.48550/ARXIV.1811.05577.

[124] K. Sarinnapakorn and M. Kubat, "Combining subclassifiers in text categorization: A dst-based solution and a case study," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 12, pp. 1638–1651, 2007, IEEE. DOI: 10.1109/TKDE.2007.190663.

[125] G. Schneider and D. E. Clark, "Automated de novo drug design: Are we nearly there yet?" *Angewandte Chemie International Edition*, vol. 58, no. 32, pp. 10 792–10 803, 2019, Wiley. DOI: https://doi.org/10.1002/anie.201814681.

[126] P. Schneider and G. Schneider, "De novo design at the edge of chaos," *Journal of Medicinal Chemistry*, vol. 59, no. 9, pp. 4077–4086, 2016, ACS. DOI: 10.1021/acs.jmedchem.5b01849.

[127] B. Settles, "Active learning," in *Synthesis Lectures on Artificial Intelligence and Machine Learning (SLAIML)*, 1, vol. 6, Springer, 2012, pp. 1–114. DOI: 10.2200/S00429ED1V01Y201207AIM018.

[128] H. Shimodaira, “Improving predictive inference under covariate shift by weighting the log-likelihood function,” *Journal of Statistical Planning and Inference*, vol. 90, no. 2, pp. 227–244, 2000, Elsevier. DOI: 10.1016/S0378-3758(00)00115-4.

[129] J. Sieg, F. Flachsenberg, and M. Rarey, “In Need of Bias Control: Evaluating Chemical Data for Machine Learning in Structure-Based Virtual Screening,” *Journal of Chemical Information and Modeling*, vol. 59, no. 3, pp. 947–961, 2019, ACS. DOI: 10.1021/acs.jcim.8b00712.

[130] B. Silverman, *Density estimation: For statistics and data analysis*, 1st ed., ser. Monographs on Statistics and Applied Probability 26. Chapman & Hall/CRC Press, 1998, pp. 1–175. DOI: 10.1201/9781315140919.

[131] A. T. Smith and C. Elkan, “Making generative classifiers robust to selection bias,” in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ’07)*, ACM, 2007, pp. 657–666. DOI: 10.1145/1281192.1281263.

[132] J. S. Smith, B. Nebgen, N. Lubbers, O. Isayev, and A. E. Roitberg, “Less is more: Sampling chemical space with active learning,” *The Journal of Chemical Physics*, vol. 148, no. 24, p. 241 733, 2018, AIP. DOI: 10.1063/1.5023802.

[133] L. Song, C. Wang, L. Zhang, B. Du, Q. Zhang, C. Huang, and X. Wang, “Unsupervised domain adaptive re-identification: Theory and practice,” *Pattern Recognition*, vol. 102, p. 107 173, 2020, Elsevier. DOI: 10.1016/j.patcog.2019.107173.

[134] R. Srinivasan and A. Chander, “Biases in AI systems,” *Communications of the ACM*, vol. 64, no. 8, pp. 44–49, 2021, ACM. DOI: 10.1145/3464903.

[135] T. Stepišnik, B. Škrlj, J. Wicker, and D. Kocev, “A comprehensive comparison of molecular feature representations for use in predictive modeling,” *Computers in Biology and Medicine*, vol. 130, p. 104 197, 2021, Elsevier. DOI: 10.1016/j.compbiomed.2020.104197.

[136] P. Stojanov, M. Gong, J. Carbonell, and K. Zhang, "Low-dimensional density ratio estimation for covariate shift correction," *Proceedings of Machine Learning Research*, vol. 89, pp. 3449–3458, 2019, PMLR.

[137] A. Storkey, "When training and test sets are different: Characterizing learning transfer," in *Dataset Shift in Machine Learning*, MIT Press, 2013, pp. 3–28. DOI: 10.7551/mitpress/9780262170055.003.0001.

[138] B. Strack, J. Deshazo, C. Gennings, J. L. Olmo Ortiz, S. Ventura, K. J. Cios, and J. N. Clore, "Impact of hba1c measurement on hospital readmission rates: Analysis of 70,000 clinical database patient records," *BioMed Research International*, vol. 2014, p. 781 670, 2014, Hindawi. DOI: 10.1155/2014/781670.

[139] M. Sugiyama and K.-R. Müller, "Input-dependent estimation of generalization error under covariate shift," *Statistics & Decisions*, vol. 23, no. 4, pp. 249–279, 2005, De Gruyter. DOI: 10.1524/stnd.2005.23.4.249.

[140] M. Sugiyama and N. Rubens, "A batch ensemble approach to active learning with model selection," en, *Neural Networks*, vol. 21, no. 9, pp. 1278–1286, 2008, Elsevier. DOI: 10.1016/j.neunet.2008.06.004.

[141] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. Von Bünau, and M. Kawanabe, "Direct importance estimation for covariate shift adaptation," *Annals of the Institute of Statistical Mathematics*, vol. 60, no. 4, pp. 699–746, 2008, Springer Science & Business. DOI: 10.1007/s10463-008-0197-x.

[142] J. Tam, T. Lorsbach, S. Schmidt, and J. Wicker, "Holistic evaluation of biodegradation pathway prediction: Assessing multi-step reactions and intermediate products," *Journal of Cheminformatics*, vol. 13, no. 63, 2021, Springer International. DOI: 10.1186/s13321-021-00543-x.

[143] F. Tramèr, V. Atlidakis, R. Geambasu, D. Hsu, J. P. Hubaux, M. Humbert, A. Juels, and H. Lin, "FairTest: Discovering Unwarranted Associations in Data-Driven Applications," *Proceedings of the 2nd IEEE European Symposium on Security and Privacy (EuroS&P '17)*, pp. 401–416, 2017. DOI: 10.1109/EuroSP.2017.29.

[144] V. T. Tran and A. Aussem, "Correcting a class of complete selection bias with external data based on importance weight estimation," in *Neural Information Processing (ICONIP '15)*, ser. Lecture Notes in Computer Science, vol. 9491, Springer Cham, 2015, pp. 111–118. DOI: 10.1007/978-3-319-26555-1_13.

[145] N. Tripuraneni, B. Adlam, and J. Pennington, "Overparameterization improves robustness to covariate shift in high dimensions," *Advances in Neural Information Processing Systems 34 (NIPS '21)*, vol. 34, pp. 13 883–13 897, 2021.

[146] B. Wang, J. A. Mendez, M. B. Cai, and E. Eaton, "Transfer learning via minimizing the performance gap between domains," in *Advances in Neural Information Processing Systems 32 (NIPS '19)*, vol. 32, Curran Associates Inc., 2019, pp. 10 644–10 654.

[147] H. Wang, Z. He, Z. C. Lipton, and E. P. Xing, "Learning robust representations by projecting superficial statistics out," preprint, arXiv, 2019. DOI: 10.48550/ARXIV.1903.06256.

[148] X. Wang, L. Li, W. Ye, M. Long, and J. Wang, "Transferable attention for domain adaptation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, AAAI, 2019, pp. 5345–5352. DOI: 10.1609/aaai.v33i01.33015345.

[149] L. Wen, "An analytic technique to prove borel's strong law of large numbers," *The American Mathematical Monthly*, vol. 98, no. 2, pp. 146–148, 1991, Taylor & Francis. DOI: 10.2307/2323947.

[150] J. Wicker, K. Fenner, L. Ellis, L. Wackett, and S. Kramer, "Predicting biodegradation products and pathways: A hybrid knowledge- and machine learning-based approach," *Bioinformatics*, vol. 26, no. 6, pp. 814–821, 2010, Oxford University Press. DOI: 10.1093/bioinformatics/btq024.

[151] J. Wicker, K. Fenner, and S. Kramer, "A hybrid machine learning and knowledge based approach to limit combinatorial explosion in biodegradation prediction," in *Computational Sustainability*, Springer International, 2016, pp. 75–97. DOI: 10.1007/978-3-319-31858-5_5.

[152] J. Wicker, T. Lorsbach, M. Gütlein, E. Schmid, D. Latino, S. Kramer, and K. Fenner, "Envipath - the environmental contaminant biotransformation pathway resource," *Nucleic Acid Research*, vol. 44, no. D1, pp. D502–D508, 2016, Oxford University Press. DOI: 10.1093/nar/gkv1229.

[153] R. Winter, F. Montanari, F. Noé, and D.-A. Clevert, "Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations," *Chemical Science*, vol. 10, pp. 1692–1701, 6 2019, The Royal Society of Chemistry. DOI: 10.1039/C8SC04175J.

[154] R. Xia, Z. Pan, and F. Xu, "Instance weighting for domain adaptation via trading off sample selection bias and variance," in *27th International Joint Conference on Artificial Intelligence (IJCAI '18)*, ACM, 2018, pp. 4489–4495.

[155] C. W. Yap, "Padel-descriptor: An open source software to calculate molecular descriptors and fingerprints," *Journal of Computational Chemistry*, vol. 32, no. 7, pp. 1466–1474, 2011, Wiley. DOI: 10.1002/jcc.21707.

[156] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" In *27th International Conference on Neural Information Processing Systems (NIPS '14)*, MIT Press, 2014, pp. 3320–3328.

[157] Y. L. Yu and C. Szepesvári, "Analysis of kernel mean matching under covariate shift," in *Proceedings of the 29th International Conference on Machine Learning (ICML '12)*, vol. 1, Omnipress, 2012, pp. 607–614. eprint: 1206.4650.

[158] B. Zadrozny, “Learning and evaluating classifiers under sample selection bias,” in *Twenty-first international conference on Machine learning (ICML ’04)*, ACM, 2004, p. 114. DOI: 10.1145/1015330.1015425.

[159] B. Zadrozny and C. Elkan, “Learning and making decisions when costs and probabilities are both unknown,” in *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ’01)*, ACM, 2001, pp. 204–213. DOI: 10.1145/502512.502540.

[160] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork, “Learning fair representations,” in *30th International Conference on Machine Learning (ICML ’13)*, vol. 28, JMLR, 2013, pp. 1362–1370.

[161] G. Zhang, B. Bai, J. Liang, K. Bai, S. Chang, M. Yu, C. Zhu, and T. Zhao, “Selection bias explorations and debias methods for natural language sentence matching datasets,” in *57th Annual Meeting of the Association for Computational Linguistics (ACL ’19)*, Association for Computational Linguistics, 2019, pp. 4418–4429. DOI: 10.18653/v1/p19-1435.

[162] S. Zhong, D. R. Lambeth, T. K. Igou, and Y. Chen, “Enlarging applicability domain of quantitative structure–activity relationship models through uncertainty-based active learning,” *ACS ES&T Engineering*, vol. 2, no. 7, pp. 1211–1220, 2022, American Chemical Society. DOI: 10.1021/acsestengg.1c00434.

[163] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy, “Domain generalization: A survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–20, 2022, IEEE. DOI: 10.1109/TPAMI.2022.3195549.

[164] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, “A comprehensive survey on transfer learning,” *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2021, IEEE. DOI: 10.1109/JPROC.2020.3004555.

[165] D. Zwillinger, *CRC Standard Mathematical Tables and Formulae*, ser. Discrete Mathematics and Its Applications. Baton Rouge: CRC Press, 2012.