Visual Integration of Model and Data Spaces in Classification Problems

Dissertation zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften (Dr. rer. nat.)

vorgelegt von

Bruno Schneider

an der

Universität Konstanz

Mathematisch-Naturwissenschaftliche Sektion
Fachbereich Informatik und Informationswissenschaft

Konstanz, 2023

Tag der mündlichen Prüfung: 27. April 2023

1. Referent: Prof. Dr. Daniel A. Keim
2. Referent: Prof. Dr. Tobias Schreck

To my beloved ones Patricia, Gael, and Teo.

Abstract

In supervised learning, a sub-area of the ubiquitous machine learning (ML) techniques, building models based on the ground truth extracted from a set of training examples defines an implicit dependence between models and data. Among the supervised learning techniques, this thesis focuses on classification problems. Given a set of known categories (classes), *classification* is the process of identifying to which class a new observation belongs.

The iterative nature of ML processes poses challenges in tracking, for instance, to which extent a classification model still works when the data change over time. It also makes non-trivial the task of following how data-classification changes during the model-building process, given the multitude of possible model parameterizations or even the combination of models in ensembles. In both examples, data visualization and interaction have the power to complement what numerical methods in isolation offer to model developers and ML practitioners.

In this doctoral thesis, the connecting point aggregating all the content is the visual integration of model and data spaces in classification problems. This thesis proposes categorizing model-data (M:N) relationships based on the number of models and data subsets at each side of that relationship. The proposed model-data relationships support the analysis of particular application scenarios.

The thesis has two main parts. The first part is about visual model comparison and visual model building of classifier ensembles. These application cases fit in the M:1 relationship, in which there are several model candidates (M) in the model space and one data subset in the data space. In visual model building, the research goal was to investigate how to integrate data and model space to enable visual analysis of classification results in terms of errors in ensemble learning. In visual model comparison, the customization of a data projection algorithm enabled a specialist's involvement in facilitating the comparison of classification landscapes produced by distinct models through anchor-points selection in data.

The second part is about visualizing the dataset shift problem in classification. This application case fits the 1:N relationship, in which there is one single model to classify several (N) data subsets. Statistics on data change, in isolation, are not enough to foresee the impacts of those changes in model performance, while data labels are not available yet. Inspired by *Anscombe's quartet*, in which visualization reveals very different data distributions of four sets with almost identical descriptive statistics, two experiments with linearly separable data are presented. For similar data changing stats, visualization reveals opposite impacts on the model's ability to classify data correctly.

Lastly, this thesis highlights the gaps in dataset-shift research from the experiments and related work. It proposes how to accomplish the persistent visual monitoring of model and data compatibility in supervised learning.

Zusammenfassung

Beim überwachtes Lernen, einem Teilbereich der allgegenwärtigen Techniken des maschinellen Lernens (ML), definiert das Erstellen von Modellen auf der Grundlage der Grundwahrheit, die aus einer Reihe von Trainingsbeispielen extrahiert wurde, eine implizite Abhängigkeit zwischen Modellen und Daten. Unter den überwachten Lerntechniken konzentriert sich diese Arbeit auf Klassifikationsprobleme. Bei einer gegebenen Menge bekannter Kategorien (Klassen) ist die Klassifikation der Prozess der Identifizierung, zu welcher Klasse eine neue Beobachtung gehört.

Der iterative Charakter von ML-Prozessen stellt eine Herausforderung dar, zu verfolgen, inwieweit ein Klassifizierungsmodell noch funktioniert, wenn sich die Daten im Laufe der Zeit ändern. Es macht auch die Aufgabe, zu verfolgen, wie sich die Datenklassifizierung während des Modellbildungsprozesses ändert, angesichts der Vielzahl möglicher Modellparametrisierungen oder sogar der Kombination von Modellen in Ensembles nicht trivial. In beiden Beispielen können Datenvisualisierung und -interaktion das ergänzen, was numerische Methoden allein Modellentwicklern und ML-Praktikern bieten.

Das Konzept der visuellen Integration von Modell- und Datenräumen in Klassifikationsproblemen fasst alle Inhalte dieser Dissertation zusammen. Diese Arbeit schlägt vor, Modell-Daten-Beziehungen (M:N) basierend auf der Anzahl von Modellen und Datenteilmengen auf jeder Seite dieser Beziehung zu kategorisieren. Die vorgeschlagenen Modell-Daten-Beziehungen unterstützen die Analyse spezifischer Anwendungsszenarien.

Der erste Teil der Dissertation befasst sich mit dem visuellen Modellvergleich und der visuellen Modellbildung von Klassifikator-Ensembles. Diese Anwendungsfälle passen in die M:1-Beziehung, in der es mehrere Modellkandidaten (M) im Modellraum und eine Datenteilmenge im Datenraum gibt. Bei der visuellen Modellbildung war das Forschungsziel zu untersuchen, wie Daten und Modellraum integriert werden können, um eine visuelle Analyse von Klassifikationsergebnissen im Hinblick auf Fehler beim Ensemble-Lernen zu ermöglichen. Beim visuellen Modellvergleich ermöglichte die Anpassung eines Datenprojektionsalgorithmus die Beteiligung eines Spezialisten an der Erleichterung des Vergleichs von Klassifikationslandschaften, die von unterschiedlichen Modellen durch die Auswahl von Ankerpunkten in Daten erzeugt wurden.

Der zweite Teil der Arbeit geht es um die Visualisierung des *Dataset-Shift*-Problems bei der Klassifikation. Dieser Anwendungsfall entspricht der 1:N-Beziehung, bei der es ein einziges Modell gibt, um mehrere (N) Teildatensätze zu klassifizieren. Statistiken zu Datenänderungen allein reichen nicht aus, um die Auswirkungen dieser Änderungen auf die Modellleistung vorherzusagen, während Datenkennzeichnungen noch nicht verfügbar sind. Inspiriert von *Anscombe-Quartett*, bei dem die Visualisierung sehr unterschiedliche Datenverteilungen von vier Sätzen mit nahezu identischer deskriptiver Statistik offenbart, werden zwei Experimente mit linear trennbaren Daten vorgestellt. Bei ähnlichen Datenänderungsstatistiken zeigt die Visualisierung gegensätzliche Auswirkungen auf die Fähigkeit des Modells, Daten korrekt zu klassifizieren.

Schließlich hebt diese Arbeit die Lücken in der *Dataset-Shift*-forschung aus den Experimenten und verwandten Arbeiten hervor. Es schlägt vor, wie die dauerhafte visuelle Überwachung der Modell- und Datenkompatibilität beim überwachten Lernen erreicht werden kann.

Acknowledgments

I am grateful for the opportunity that Prof. Dr. Daniel A. Keim, my first advisor, opened for me to work in one of the top *information visualization* and *visual analytics* research groups worldwide, his DBVIS group at the University of Konstanz. I also want to thank senior members of the group who provided me additional guidance and mentorship during my stay in Germany. Namely, Sebastian Mittelstädt, Johannes Fuchs, Michael Behrisch, Menna El-Assady, and Dominik Jäckle. In particular, Menna contributed decisively to the information theory standpoint on the dataset-shit problem in Chapter 5.

Regarding collaboration in writing research papers, I thank Dirk Streeb for the invitation to coauthor one of his published works in a first-tier journal of our field. I thank Michael Blumenschein, Florian Stoffel, and Alexandra Diehl for being coauthors in my research papers, together with the senior members I mentioned before. I am proud of my partnership with Jan Laun-Cieslik, a BA student I had the chance to supervise. I am also very grateful to Prof. Dr. Tobias Schreck, my second advisor.

Lastly, from outside the research environment, I cannot forget to mention the importance that Stefan Bühler, Dorthe, Christiane Schmidt-Neubig, Stefan Neubig, Ulrika, and Isabela had during the time I lived with my wife in Konstanz. I am from Rio de Janeiro, Brazil. Together with my supervisor and colleagues at the DBVIS group, they gave me the chance to experience a unique and unforgettable time in Germany.

Contents

1
Introduction

Contents

In *supervised learning*, a sub-area of the ubiquitous *machine learning* (ML) techniques, the process of building models based on the ground truth extracted from a set of training examples defines an implicit dependence between models and data. Due to machine learning processes' dynamic and iterative nature, it is not trivial to follow how data and models change over time or track the relationships among them. The inability to track model-data relationships may result, for instance, in the incapacity to decide if a model is still usable when data change. In my research, I visualize data and models to support model developers and domain experts in supervised learning applications. The visualization of data changes is in the scope of this thesis. Visualizations work as a layer that facilitates manipulating those objects and understanding complex phenomena that numeric scores cannot convey in isolation.

Another issue in supervised learning that the visual approach helps mitigate is how to explore the explosive number of model types and corresponding parameter settings that may fit the same problem. On the one hand, automated approaches [10] build and put several models in competition to keep the best one, given one or more performance scores of preference. On the other hand, it is also true that depending on the data domain, the application goals, and characteristics, it is not desirable to leave behind several model candidates based solely on one metric. For instance, analyzing in the data space which regions are more expensive to experiment with errors may generate competing alternatives that justify human intervention. Also, a deep understanding of how each model works cannot be substituted by a low-code tool, for instance, that automatically trains hundreds of models and automatically selects one based on scores. Machine learning is iterative, and what happens in one execution with the available data may not occur again in the following rounds with incoming data.

My research aims to support model selection by offering developers and data specialists an interface in which they can interactively experiment with several model combinations and build classifier ensembles. I do not go against or compete with the automated approaches. Instead, I build on top of them by first running algorithms for automatic model selection in ensemble building. I let the human operate on top of it and decide to fine-tune or accept the automatic configuration. The power of the visual approach is that it sets a live environment that one can explore in which directions it is beneficial to keep searching for the best possible model, guiding the development efforts in a complex space of possibilities. Also, it supports the decision among candidates when there is no single winner for the problem at hand.

Regarding terminology, *Artificial intelligence*, *Machine Learning*, and *Deep Learning* are frequent terms in scientific and non-scientific literature nowadays.

However, they do not mean the same thing.

"Artificial intelligence is an umbrella term for machines that are capable of mimicking human intelligence and behavior. Machine learning, on the other hand, is a process used to achieve artificial intelligence. It involves designing algorithms that can learn from data to become more accurate and effective over time. Deep learning is a sub-field of machine learning that draws inspiration from how the brain works. It involves using layers of artificial neural networks that create a human-like logic structure." [18]

From a broader perspective, this research is directly influenced by the *Visual Analytics* [51] research field mantra that advocates in favor of *keeping the human in the loop*, instead of letting automatic processes run by themselves. The generic goal is designing the right combination that profits from human analytic capabilities and machine-heavy processing power, taking the virtues of both sides and combining in interactive systems backed by visual interfaces. In machine learning, the adoption of visual interfaces also talks with a strong initiative in both industry and academia called *explainable AI*. As the adoption of ML increases worldwide and exerts a significant impact on our daily lives, society looks for the government regulation of ML systems [6]. Therefore, the explanations that bring transparency to ML directly profit from the long-tradition in visualization research of using visual representations to explore and explain complex phenomena.

Finally, I focus on *classification* problems, one of the supervised learning techniques in ML. However, part of my contributions are extensible to regression problems, the other important technique in which one is interested in predicting exact numerical values instead of class membership after training a model. Deep learning intrinsic characteristics, a hot topic nowadays, is out of our scope in this thesis.

1.1 Motivations

Machine learning (ML) represents a new paradigm in computer science. One does not explicitly write a sequence of instructions to a computer but instead builds more complex artifacts (models) that derive rules from data. However, it is challenging to understand why a particular model produces a particular output depending on the learning strategy. The inner complexity of ML brings an opportunity for *information visualization* and visual analytics research fields. Visualization can serve as a layer to access and manipulate those objects, increasing the understandability and providing greater control over the whole process.

Additionally, supervised learning has a dynamic and iterative nature. By design, there is never an end to the learning process because when new data appear, the models can be retrained, incrementally or by restarting the whole process. Therefore, textbooks' simple train/test setup does not reflect the reality behind the countless potential iterations. Visualization can also play an important role, namely the visual comparison of the model outputs (e.g., classified data in classification) across these iterations. Visual representations potentially increase the understandability and control over the learning process, especially in significant data changes over time. In the following, I present two particular motivations for using visualization to support ML, which resulted in two chapters of this thesis.

1.1.1 Model and Data Space Visual Exploration

Machine learning models offer the chance to fine-tune hyperparameters, seeking the best possible performance. On the one hand, there are indeed existing approaches that perform automatic model parameterization. On the other hand, it is also true that for a given change in model parameters or even the model type, it is difficult to track the impacts on the data space and how the new configuration affects different regions of that space, i.e., data classes in classification. In my research, I propose the visual integration of data and model spaces in classification by using visual analytics to explore and manipulate those spaces.

Visualization and interaction can play the role of offering an environment in which intensive experimentation with data, models, and their relationship is possible, aiming to increase the control of model developers and data specialists over the model building and maintenance processes. Therefore, independent of the convenience offered by automatic approaches, the control layer supported by visual methods and the interactive capabilities can work hand-in-hand with automation, serving as a playground for experimentation and guiding machine learning practitioners for greater control over the processes.

Still, the model building process is closer to handcraft creations than an artifact that one can automatically obtain. The specialist knowledge of model behavior and characteristics is crucial for a proper selection. After the feature selection process from data to train a model, which is out of the scope of this thesis, the model keeps changing when data change to maintain its usefulness. Therefore, the manipulation of these complex objects over time is far from trivial. Using the metaphor of model and data spaces, if we consider that there is a multiplication of objects on both sides as time goes, we can understand the potential benefits of building visual representations of these spaces to enable the identification of visual patterns on both sides. In Chapter 4, I show in greater detail how these representations work. By representing models as data, they also become data points. Consequently, they inherit all the capabilities that visualization brings to identify correlations, clusters, and other patterns in the model space.

1.1.2 A Latency Problem in Supervised Learning

Depending on the data domain, when one uses a model to classify unlabeled data, there is a time gap to obtain the actual data labels. This time gap can be huge. In image classification, for example, human judgment is necessary to confirm classes and build an initial training set. Despite the strategies that exist to make data labeling more efficient [61], the process of building training samples from an initial set of human-annotated data is expensive.

So, questions that arise are: what to do in the meantime? How can I track model performance while the data labels are not available yet? These questions bring a second motivation for using data visualization. Inspired by *Anscombe's quartet* (Figure 1.1.1, in which visualization reveals very different data distributions of four sets with almost identical descriptive statistics), I use visualization to display data shifts in unlabeled data based on data-class movements. For similar data-changing stats, visual patterns anticipate very different impacts on model performance. By tracking how classes expand or contract in unlabeled data, compared to what the model learned at training time, it is possible to foresee if a model is still usable

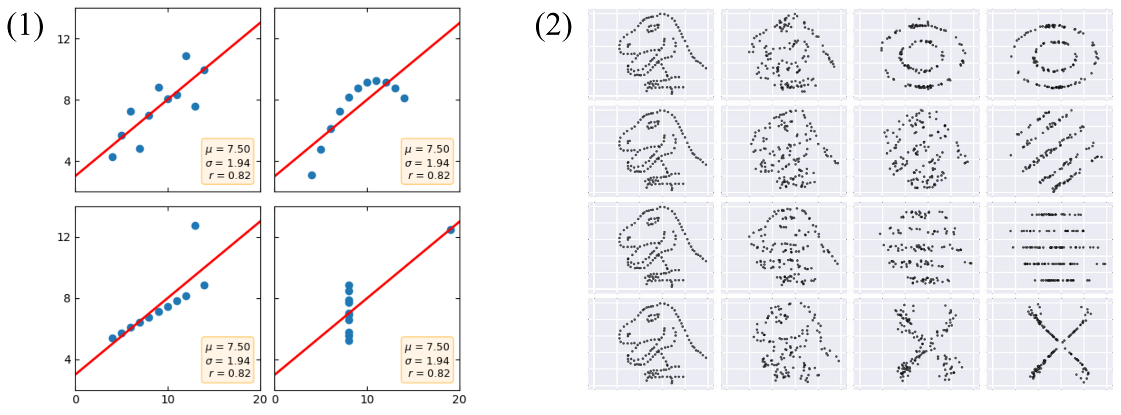


Figure 1.1: On the left (1), we see the *Anscombe's quartet* [5], in which scatterplot visualizations show completely different data distributions in four datasets with almost identical standard statistics. On the right (2), Matejka and Fitzmaurice [69] go further and present a generative model that builds datasets with even more different distributions but with the same stats. The immediate conclusion is that statistical measures cannot discriminate these datasets without data visualization.

even in the presence of latency to obtain all data labels (see the experiments in Chapter 7).

1.2 Research Trajectory

I started my research decided to focus on classification problems, a supervised learning technique that, in its standard form, automatically assigns a category to unlabeled new observations after building a model based on predefined categories found in training data. This starting point has to do with the research project I presented to obtain my funding sources. Classification was the technique I selected to propose the visual building of predictive models with movie industry data. Before I moved to Germany to start my Ph.D., I worked as an analyst in a government agency in Rio de Janeiro, Brazil, dedicated to the audio-visual industry.

From the beginning of my research, it was clear that the generalization of the research problem instead of working only with movie data was the direction to follow. The broader applicability of the research findings and the possibility to publish in the main information visualization and visual analytics research conferences, instead of collaborating mainly with experts for a particular domain, justified that decision.

I started working on the classifier model-comparison topic in collaboration with Sebastian Mittelstädt, a senior member at that time of my research group. Besides the visual approach, we worked on adapting a dimensionality-reduction algorithm to allow anchor-point selection when projecting data in two-dimensional space. This way, we facilitate the comparison among distinct projections of classification model outputs (in Chapter 3).

Right after, I focused on the visual model-building topic. The first reference that guided me through the beginning of my research was the work of Olah [74], in which this author uses the concept of the *space of representations* to connect different neural network models trained with distinct architectures to corresponding data outputs. In his interactive prototype, one can select one model configuration at each time and inspect the corresponding outputs. Models appear on the left-hand side in the prototype (model space) and data on the right-hand side (data space). From this reference, I started to build on top around the idea of allowing the combination of models in the model space to create classifier ensembles [28]. These ensembles are a type of model built from the combination of individual learners. Each output comes from aggregation criteria of the individual decisions to produce the ensemble decision (e.g., majority voting [14]).

Colah's work made me think about the power of having model space representations, very compact ones but in a way that models become the data. In this sense, they inherit all the properties of standard visual representations of any data points (i.e., the capacity of identifying clusters as previously mentioned). My first published full-paper work was on model and data space visual integration. In my work with classifier ensembles, one can interactively combine several pre-trained models on a left-hand side model space representation and see the impact on data classification on the right-hand side. If one chooses one single model at each time instead of combining them, this is what Colah has done, and from where I got inspiration at the beginning.

After working with evolving model configurations and the impacts on data classification, a natural path to follow was to examine the persistence/stability of model performance when data change, i.e., the performance across different (incoming) data subsets. This movement guided me to the *dataset shift* problem in machine learning (see Chapter 5), a well-studied problem by the ML research community. However, to my surprise, information visualization and visual analytics research communities still under-explored this problem to the best of my knowledge. These communities worked on issues like feature engineering, interactive model building, and model explainability, to name a few. However, the evolving nature of data and its impacts on supervised learning still offered room for research.

Throughout the development of my research, I realized that the visualization of model and data space representations and their connection guided me all the time. What changed at each application was the possible combinations of the number of models (one single model or M models) and the number of data subsets (one single dataset or N datasets). Therefore, from this research trajectory, I devised a categorization of the possible four model-data relationship combinations. This construction helped organize my thesis and constituted a proposal of an entry-point to analyze ML processes and contribute to its development. It also helped show what I explored and what is still open for future work in the field.

1.3 Thesis Related Work

In this section, I present, centralized and aggregated, the related work to the main topics of my thesis. The anticipation of this discussion connects to limitations and future work presented in Chapter 8. A detailed discussion of the related work appears in the following Chapters, connected to each corresponding part of this work.

As previously presented in my research trajectory (Section 1.2), this thesis starts with visual model-building methods and advances into analyzing model generalization with unseen data during the training phase. Into each of the two main topics (model building and model generalization), I worked respectively with ensembles of classifiers and with the analysis of the dataset-shift problem. Ensembles are a specific model-type that combines individual learners and their results in a single model, aiming to profit from the virtues of distinct learning philosophies. Figure 1.2 sums up part of the bibliography of this thesis connecting it to each main topic,

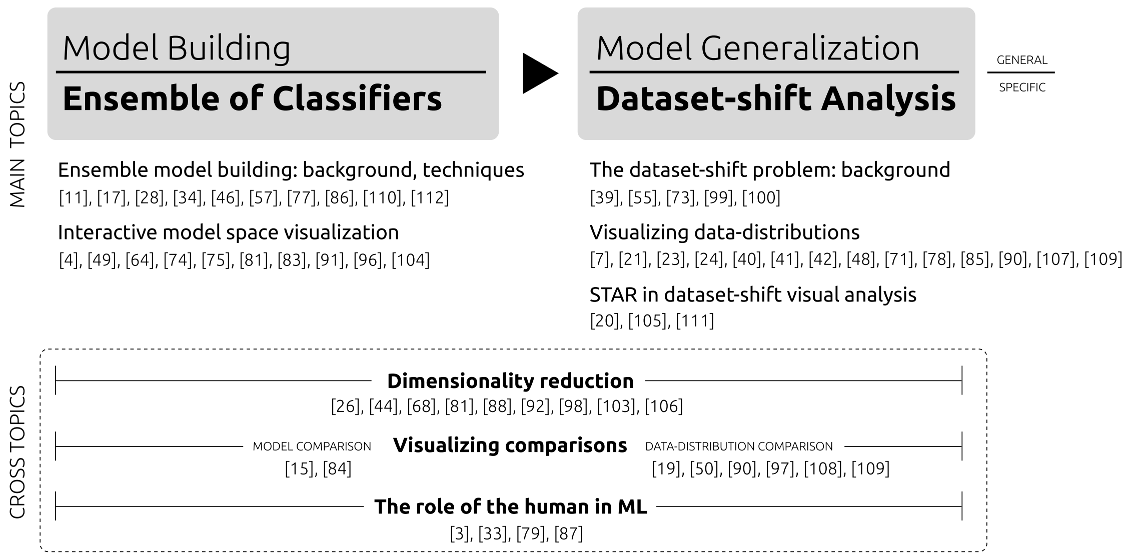


Figure 1.2: My work in this thesis starts with visual model-building methods and goes in the direction of model generalization visual analysis, focusing on dataset-shift problems. In this figure, I summarize the main and transversal cross topics. In each case, I follow with the references that I cite throughout the dissertation, giving a quantitative overview of the most researched subtopics that anchor my research.

which allows us to have an overview of the most explored themes by counting the number of references linked to each. The figure also brings three transversal topics to Parts I and II that build up my work: dimensionality reduction techniques, the role of the human in machine learning, and visualizing comparisons.

In quantitative terms, in the first Part of this thesis – which deals with visual model building – the subtopic I most researched was the foundations and techniques on building ensembles of classifiers before any visualization takes place. There is a vast amount of research on this subtopic. It offers rich opportunities to apply the visual model-building approach and work on the combination of individual learners by interactively incorporating analysts' specialized knowledge into the process. The second most prominent subtopic in the 1st Part of my work was the group of techniques for the interactive visualization of classifier model spaces. There is also a solid body of research on this subtopic, which brought me to my contributions in this Part in connection to the foundations of ensemble model building.

In Part II of the thesis, which deals with model generalization, I started with the foundations of the dataset-shift problem, where extensive research exists. On the other hand, despite a few STAR papers that apply visualization and visual analytics disciplines to data-changes analyses over time, I found a gap in this intersection compared to what exists around the topic out of visualization research communities. So far, extensive research exists on the visualization and visual comparison of data distributions, which facilitates the identification of certain types of dataset shifts, but only partially addresses some of its possible variations.

Regarding the cross-topics of the thesis, the use of visualization for making comparisons appear in both Parts (I and II). In the first part, I develop a specific application for comparing candidate models (Chapter 3), a task connected to model-building. In Part II, visualization played a fundamental role in comparing data distributions of the same feature in different data subsets and comparing training and unseen datasets (all features considered). In these developments, related work on the possible visualization techniques and visual layouts supported it throughout this work.

The second cross-topic was the role of humans in machine learning. It is a ubiquitous background, considering that I conducted my work in the visual analytics research community, where a fundamental goal is exploring how to conjugate human/analyst capabilities with the machine processing power integrated into the same pipeline. In Part I, this approach helped to guide the implementations of which input types an analyst can give to the classifier model-building process, connecting it to the model steering possibilities on top of automated methods. In Part II, I implemented techniques for including the human in the dataset-shift analysis loop and also proposed additional ones as future work in Section 7.7 (*Towards a General-purpose Visual-based Monitoring of Data and Model Compatibility in Classification Problems*).

The dimensionality reduction techniques (e.g., PCA, MDS) are the last important cross-topic I emphasize in Figure 1.2. I use these techniques in both Parts (I and II) for visually representing in 2D the model and data spaces after reducing a more extensive set of collected features in both cases to the two-dimensional space. Regarding the visualization techniques, I use scatterplots to represent the reduced data after projection. Due to their simplicity and low-learning curve, they are the most common form of visualization for this data type [93]. Scatterplots also allowed me to offer direct manipulation of models and data in a one-to-one representation of those objects. In conjunction with heatmap techniques, I also dealt with data scalability issues, managing overplotting in this visualization type.

Alternatively to the projections, I also work with filtering to select data features of interest for direct representation, binned per data class. However, dimensionality reduction plays its role when it comes to scale for datasets with any number of features without visualizations subsets of them at each time. Despite its importance, dimensionality reduction has its cons. Section 8.1 highlights the limitations associated with projection methods, like the data-loss problem. Then, Section 8.2 presents alternatives to mitigate those limitations in an application context.

1.4 Thesis Outline and Contributions

In my thesis, the connecting point that aggregates all the content is the visual integration of model and data spaces in classification problems, which is not by coincidence the title of this work. Regarding the organization of this content, I divided it into three parts. The first part contains two chapters. One is about the visual comparison of classifier models, and the other is about the visual model-building of classifier ensembles. The second part is about visualizing the dataset shift problem in classification and has three chapters. Chapter 5 introduces the dataset shift problem. Then, in Chapter 6 I present a general-purpose technique to compare multidimensional data distributions that also support dataset-shift identification and understanding. Chapter 7 concludes the second part of this thesis by presenting two experiments and the corresponding implementations that build evidence on the utility of using data visualization to detect and understand data changes. Still, this chapter also builds on a proposal of full visual support for monitoring data and model compatibility in classification problems. Finally, the third part concludes.

After this Introduction chapter, I present in the following my categorization of model-data relationships in the next chapter. This conceptualization describes four possible relationships. In my research, I instantiated two of them. I present both the model comparison and the ensemble building application based on the

M:1 relationship in the first part. There are diverse models to compare/manipulate in the model space and just one dataset to classify. Then, in the second part, I instantiate the 1:N relationship to visually explore data changes among distinct data subsets, keeping the same model in the model space. The last part concludes with open research gaps devised directly from the non-explored models-data relationships.

My main contributions are:

1. My work on the visualization of model spaces (Chapter 4) showed the benefits of looking for models as data points by inheriting the benefits and insight generation that came by default from the visual exploration of any standard data object. One possible output of a paper is influencing other works. My publication in *2017 IEEE Visualization in Data Science* (VDS, honored as the best paper [91]) played this role, and papers published in first-tier journals cited it in the following years.

2. A general-purpose new technique for visual comparison of multidimensional distributions (Chapter 6).

3. The implementation of novel techniques in two experiments that allow the visual identification of dataset shifts which result in no impact on performance before having the data labels, in the case of linearly separable data (Chapter 7).

4. The identification of research gaps and a corresponding proposal toward the full visual support for monitoring data and model compatibility in classification problems (Chapter 7).

2 Model-data (M:N) Relationships

Contents

2.1 Model-data relationships in supervised learning

This chapter proposes a categorization of model-data (M:N) relationships that offers an alternative entry-point to analyze supervised learning from a set of four possible M:N cases. The proposed model-data relationships help to categorize and analyze particular application scenarios.

Existing conceptual representations of supervised learning (SL) include the pipeline view (Figure 2.1.1) and the inputs/outputs view (Figure 2.1.2). In SL, the pipeline view helps to organize the sequence of processes that need to run, no matter their number, from raw data collection to the final outputs that predictive models produce at the end. The inputs/outputs view explicitly shows what comes in and what comes out at each step. Both views complement each other. The inputs/outputs view can appear as an additional layer of information attached to the pipeline view.

I follow a complementary approach. I take a particular supervised learning application and consider the number of models involved (1 or M) and the number of datasets (1 or N). I use the notation M:N to refer to the four possible model-data relationships. M and N are equal or greater than 2 (two). My conceptual representation does not substitute the pipeline view or the inputs/outputs view. Instead, it offers a lens to analyze supervised learning using both models and data as the protagonists to derive the possible relationships between them, depending on the number of models and data subsets for any particular application. It offers the advantage of highlighting the protagonists, models, and data. The other mentioned views excel at showing the big picture but do not serve to explain the M:N relationship I decided to give focus.

Besides giving a view of possible M:N relationships, my categorization serves as a conceptual representation that directly supports planning interactive applications in the SL context. By design, the model-data relationship is inherent to the inner working of supervised learning methods. Consequently, the relevance of that relationship, which I take as a foundational component of my thesis, comes directly from how those methods work. The datasets (N) consist of the initial training dataset plus the incoming data subsets not available at training time. It is out of the scope of my work both model transferability (when the dataset changes and one tries to use a pre-trained model) and incremental learning methods with streaming data.

2.2 The four M:N cases

In Figure 2.2, I introduce the four possible model-data relationships.

In the following, I describe each of them.

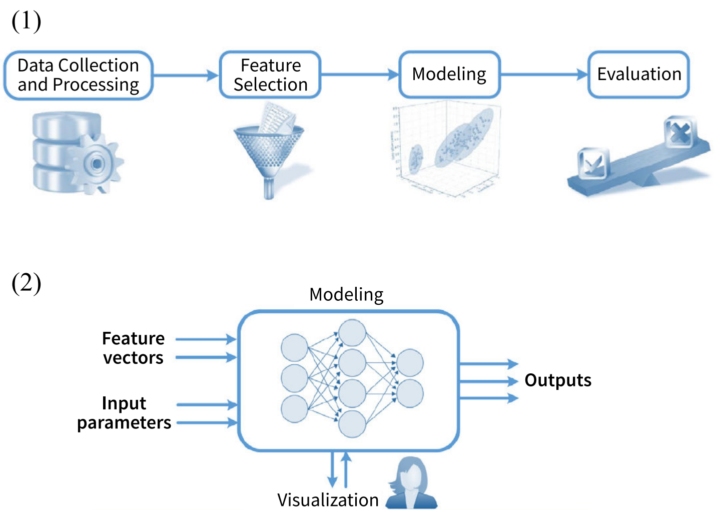


Figure 2.1: The ubiquitous pipeline view (1) and inputs/outputs view (2) are possible visual artifacts to explain supervised learning processes. They can also complement each other when the second representation works, for instance, as a detailed view for each step of the first. In my research, I look for a representation that does not compete or substitute both but offers a complementary entry-point for categorization, identification of research gaps, and most importantly, a direct metaphor to design visual analytics applications. (adapted from [64]).

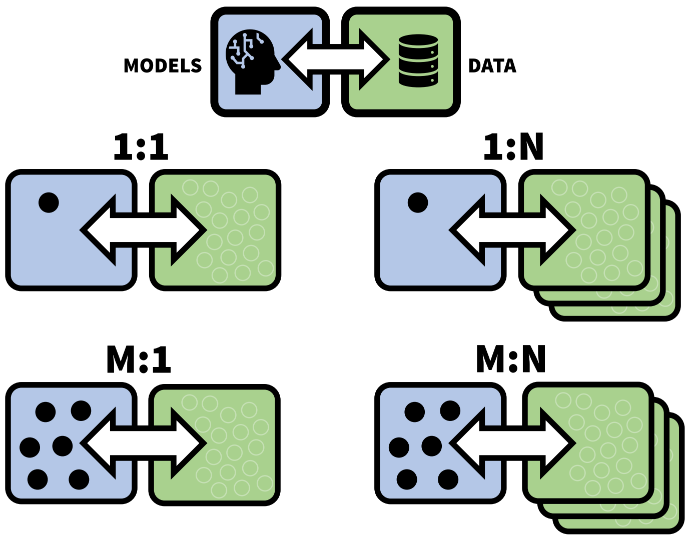


Figure 2.2: I propose a categorization of model-data (M:N) relationships in supervised learning divided into four cases based on the number of models (M) and data subsets (N) in model and data spaces, respectively. In the visual representation above, the model space appears blue on the left side, and the data space is green on the right side for each case. They are always connected by an arrow representing both, enabled from an application perspective by visualization and interaction.

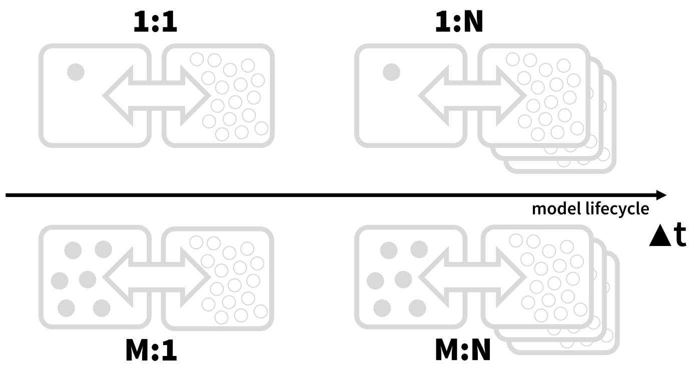


Figure 2.3: We can also look at the four M:N model-data relationships over time. From this perspective, the two cases with one element in the data space (1:1 and M:1) correspond to the beginning of the model lifecycle that will evolve to more data subsets as time goes.

- Case 1 (**1:1**): this is the most classical and straightforward setup, in which there is one single model and one dataset. For my visual integrative approach of model and data spaces, this is the less interesting case. There are no distinct models to experiment with within the model space nor multiple data subsets to compare in the data space. Also, the presence of just one single dataset does not allow visual comparison of how the model classified these data (in classification), or the inspection of data shifts, for instance. Under Case 1, standard methods like k-fold cross-validation or the ones that estimate model bias and variance (see Figure 4.12 in the next chapter) can return numerical scores to support model generalization analysis.
- Case 2 (**M:1**): here, there are two or more models in the model space and one single dataset. The diversity of models can come from different model types (e.g., decision tree model, nearest neighbors, neural networks), the same model type with varying parameter settings, or both things together. From an application perspective, several models in the model space give rich opportunities to explore them by interactively selecting one model at each time and verifying the corresponding outputs in the data space. Additionally, another application is combining the models in ensembles. Chapter 4 presents one application of this case for ensemble model building, and in Chapter 3, I also offer an M:1 case, but for model comparison.
- Case 3 (**1:N**): in this third possible relationship, there is one model in the

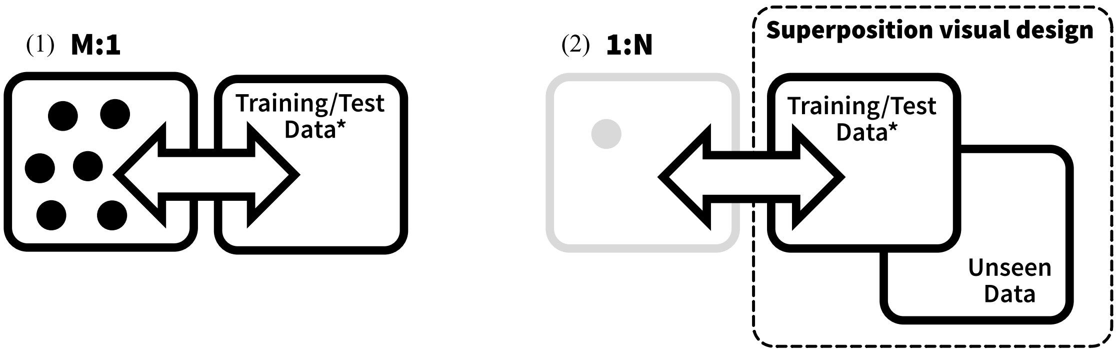


Figure 2.4: I separate two of the four possible M:N relationships presented in Figure 2.2. The M:1 case appears in the applications developed in Chapters 3 and 4 of this thesis, with several models in the model space for interactive ensemble model building. Then, in Chapters 6 and 7, I develop 1:N applications. When M is 1, I do the visual integration of N data spaces in the applications. In this case, the existence of the model space is implicit. Therefore, it does not appear anymore in the visualizations as there is no particular motivation to make explicit this single model.

dataspace and at least two or more data subsets. We can see this case (1:N) as an extension of the first one (1:1). With the possibility of visually comparing multiple data subsets in the data space, both the exploration and visual analysis of the dataset shift problem or just the inspection of how the model generalizes to new data are supported. In Chapters 6 and 7, I present applications of this case on the dataset shift problem.

- Case 4 (**M:N**): in this last case, we have a diversity of both model and data subsets. This case is an extension of the second one (M:1). From an application perspective, this extension enables merging the interactive capabilities of ensemble model building with the visual analysis of how they generalize to incoming data subsets and the support to the dataset shift issue. I present research opportunities under this case in the last chapter.

The utility of the four possible model-data relationships presented abstractly becomes more evident when I describe the applications based on them in the following chapters. These applications also provide concrete examples of how I visually represent models and data in the respective spaces and implement the interaction capabilities to explore their relationship.

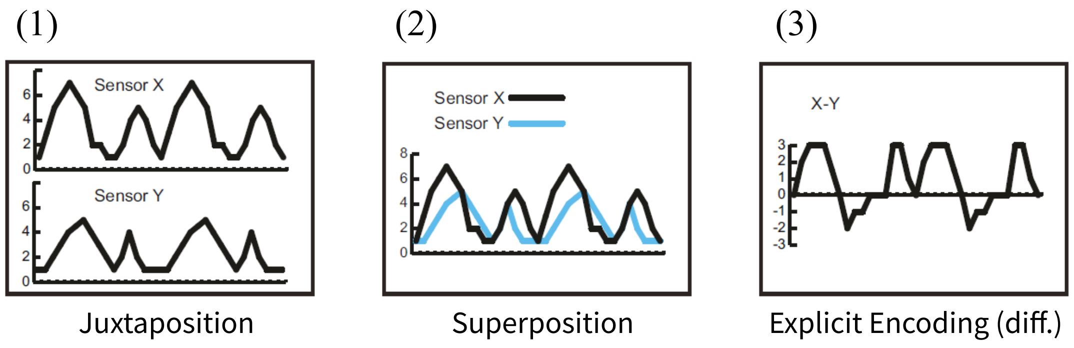


Figure 2.5: Visualization has the power to present in a compact form vast amounts of data. An application is the visual comparison of data. There are different ways (layouts) to display visual comparisons. Above, I selected three of them (1-3, adapted from [36]). In the third case, one single visualization directly depicts the difference between two datasets.

In Chapter 4, I deeply explored how the data space reacts to changes in the model space through interaction. In the other direction, I also studied how the model space reacts to selections in the data space. For instance, I implemented the reorganization of the model representations to show the best-performing ones accordingly to preferences of specific regions in the data space, providing a complete *tour* in both directions. The rich interactive capabilities of model-data relationships visual exploration support hypothesis forming that a series of static visualizations cannot convey.

Finally, in Chapters 6 and 7, I visually compare N data subsets to support the visual analysis of data changes for the same model type. The visual comparison of N subsets occurs by overlapping them in the same visualization and computing data densities to reveal only the relevant differences (like in Figure 2.5.3).

Part I

Visual Model Comparison and Ensemble Model Building (M:1)

3
When Individual Data Points Matter: Interactively Analysing Classification Landscapes

Contents

Parts of this chapter appear in the following publication:

- B. Schneider, S. Mittelstädt, and D. Keim. When individual data points matter: interactively analysing classification landscapes.*Eurographics Conference on Visualization (Poster paper)*, (2016): 013-015.

3.1 Introduction and Related Work

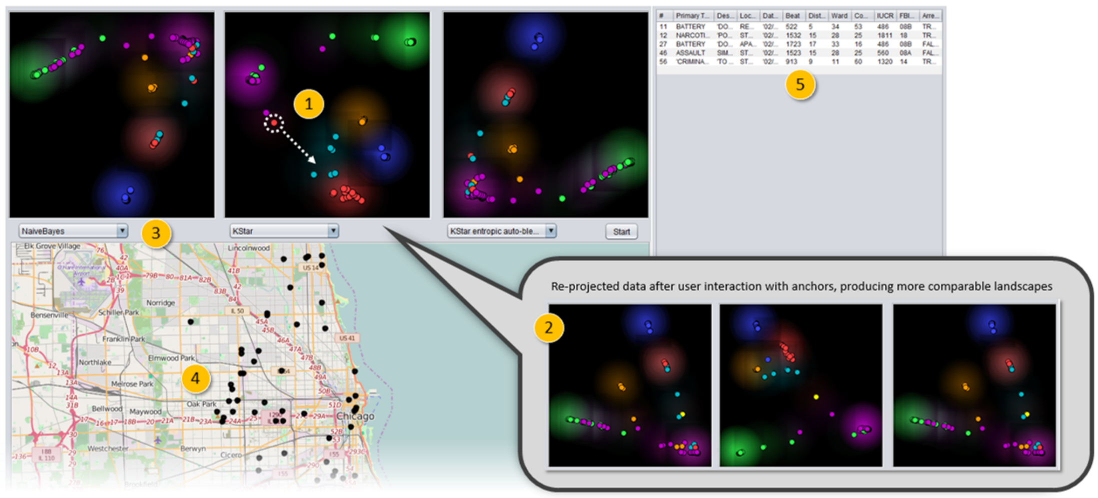
In some application scenarios of data analysis, the cost of misclassified or misinterpreted data instances is crucial. In crime intelligence analysis, for example, wrong classified data records could influence the life and reputation of persons critically. We argue that the real accuracy of a model cannot be estimated without domain and expert knowledge. The decision making processes require an understanding of model and data at the same time. Automatic methods oppose dangers in the decision making process since every dimensionality reduction (classification and projection) or aggregation hides important information for detailed analysis.

Interactive machine learning provides methods to build accurate models by integrating expert knowledge. Typically, performance measures evaluate the model while interactive refinement focuses on optimizing these measures or classification borders. Therefore, it is possible to build and refine several models with appropriate accuracy based on *global* measures. The emergent question is: which model to select if there are multiple models with similar accuracy?

In our target application scenario, however, *local* patterns matter because decisions are based on individual records which are not captured in *global* measures and classification borders. Therefore, we contribute with a classification landscape visualization based on multidimensional scaling (MDS) that allows *global* model comparison and analysis of *local* patterns. We extended an existing MDS algorithm [37] by interactive anchor-points selection that aims to stabilize different projection results. The anchored projection thereby allows to experience changes of models and parameters and, thus, supports experts in understanding and comparing models.

Related Work Visualization plays an important role in supporting model comparison during the process of model building [15]. After model construction, several works also propose visual techniques to evaluate classification results [29, 52, 83]. In particular, Alsallakh et al. [2] visually analyze class probabilities estimates. Kapoor et al. [49] apply confusion matrices for interactive optimization of the model.

The works mentioned above focus on the aggregates of probabilities or global performance measures and do not allow local pattern analysis. Differently, Migut



#	Primary T...	Des...	Loc...	Dat...	Beat	Dist...	Ward	Co...	IUCR	FBI...	Arre...
11	BATTERY	'DO...	RE...	'02/...	522	5	34	53	486	08B	TR...
12	NARCOTI...	'PO...	ST...	'02/...	1532	15	28	25	1811	18	TR...
27	BATTERY	'DO...	APA...	'02/...	1723	17	33	16	486	08B	FAL...
46	ASSAULT	SIM...	ST...	'02/...	1523	15	28	25	560	08A	FAL...
56	'CRIMINA...	'TO ...	ST...	'02/...	913	9	11	60	1320	14	TR...

1
2
3
4
5
NaiveBayes
KStar
KStar entropic auto-ble...
Start
Re-projected data after user interaction with anchors, producing more comparable landscapes
Chicago

Figure 3.1: Our system provides visual model comparison by (1) showing side-by-side the classification landscapes of different models in linked panels after applying MDS projection on the probabilities estimates for each class of each data instance, (2) enabling interactive anchor-points selection, dragging and data reprojection for producing more comparable landscapes, (3) allowing the selection of other previously evaluated models, (4) letting the user collaborate with domain knowledge through direct selection of georeferenced events on a map, and also (5) giving detailed information of each data instance in a text table.

et al. [72] apply scatter plots on non-aggregated data. However, the attributes of the data are directly plotted, preventing the user from estimating global classification borders and local classifications patterns. In these attribute visualizations, the user has to choose which pair to see at each time throughout several distinct visualizations. In our approach, we use all attributes in data projections of the classification landscape and visually organize similar classified records close to each other to facilitate visual comparison among model candidates.

3.2 Interactive Anchor-points Selection in Classification Landscapes

We developed a prototype for the comparison of classification landscapes produced by different machine learning classifiers, and after the process of model building. The visual components and our workflow were designed to support analysis tasks in which it is important to compare classification borders between different models with similar global accuracy, while preserving individual data instances. Also, in our system the user can load up to three previously built classification models for visual exploration of the outputs.

The classification results are shown in scatter plots, one for each model, after applying MDS projection on the input data. Each class is represented by a different color and high densities of records of the same class reveal position of classes in the landscape. Regarding the data projection, we applied multidimensional scaling on the probabilities estimates of each data instance for each predicted class, instead of projecting all the original data attributes. By projecting the data with fewer dimensions, we minimized the problems that arise with the curse of dimensionality.

All visualization panels are interconnected by linking and brushing, enabling top-down analysis from global classification borders to local crime patterns in their geographic context. The crime events can also be analyzed in a geo-referenced context, together with a text table providing details for each crime event.

The global comparison of classification landscapes can be difficult since the class positions and borders may appear in different locations in different scatter plots. Therefore, we implemented interactive anchor-points selection, in which the user can select single records, drag, and anchor them in the scatter plots and, thus, within the projection. Then, when the user drags a data point the MDS projection is recomputed without moving the anchored records in the stress minimization. Thus, the anchored records will appear at the same position of all plots. Additionally, all other points are moved accordingly in the projection, e.g., dragging all similar records towards anchors and thereby also class positions. Since all classification landscapes share the same anchor points, global comparison can be performed with less cognitive load.

3.3 Use Case: Visual Analysis of Crime Classification Data

We have chosen to work with a crime dataset from the city of Chicago, U.S. [27]. This choice fits our proposal focused on domains where it is important to keep track of individual events, due to the sensibility of related issues and resulting high costs of wrong predictions. The attributes selected for prediction were the most frequent six types of crimes from the dataset, in a way similar to the competition organized in [47].

We trained seven different classification models. In this group, we have one model based on Neural Networks, a Support Vector Machine model, a KStar instance-based classifier, Decision Trees and also a K-nearest neighbours model. Some of them gave us very good accuracy for our classification task (results ranging from 72 to 99% of accuracy).

Regarding findings, our tool achieved initial good results in a scenario where we had the same classification model with 3 different parameter settings. Then, by manipulating the data-points as shown in Figure 3.1, we generated much more comparable shapes among plots after user interaction and data manipulation than the initially generated automatic projections by the system. To choose which point to select as an anchor and drag it, we prioritized points that were classified differently by one single model and moved it into the same direction that this point was classified by the other models.

We see our work as an effort for better understanding of individually classified data instances, with an important extension for geo-referenced data. The proposed framework and the way we applied the data projections with interactive anchor-points selection could be incorporated into more extensive predictive modeling workflows. Also, the interactive anchor-points helped in providing more comparable classification landscapes, through direct user manipulation supported by contextual information.

Despite our efforts on bringing the most of contextual information as possible to the analysis of the results produced by different classifiers, this work can be extended towards the field of *Interactive Machine Learning* [79], in which the users can improve the performance of different classifiers through visual means.

4
Integrating Data and Model Space in Ensemble Learning

Contents

Parts of this chapter appear in the following publication:

- B. Schneider, D. Jäckle, F. Stoffel, A. Diehl, J. Fuchs, and D. Keim. Integrating data and model space in ensemble learning by visual analytics. *IEEE Transactions on Big Data*, 7.3 (2021): 483-496.

4.1 Introduction

Given a set of known categories (classes), *classification* is defined as the process of identifying to which category a new observation belongs [89]. In the context of machine learning, classification is performed on the basis of a training set that contains observations whose categories are known. A key challenge in classification is to improve the performance of the classifiers, hence new observations are correctly assigned to a category. Classification can be performed with a variety of different methods tailored to the data or the task at hand. Examples include, among others, decision trees, support vector machines, or neural networks.

Research proposes to improve the accuracy of classification using *Ensemble Learning* [28, 113], also known as *Multiple Classifier Systems* (MCS) [86]. Such systems suggest combining different classifiers, potentially expanding the space of representable functions by using distinct learning philosophies at the same time. Well-known approaches for building ensembles propose to either train the same model successively with different subsets of the data [11, 34], to combine different model types [57, 110], or to combine different strategies such as bagging [11] with random feature combinations in Random Forests [12]. Generally speaking, the application of ensembles increases the complexity of the classification process bringing in the inherent problem of decreasing comprehensibility. In particular, it is challenging to understand how and to what extent the models contribute to the classification, as well as which models produce a significant number of classification errors.

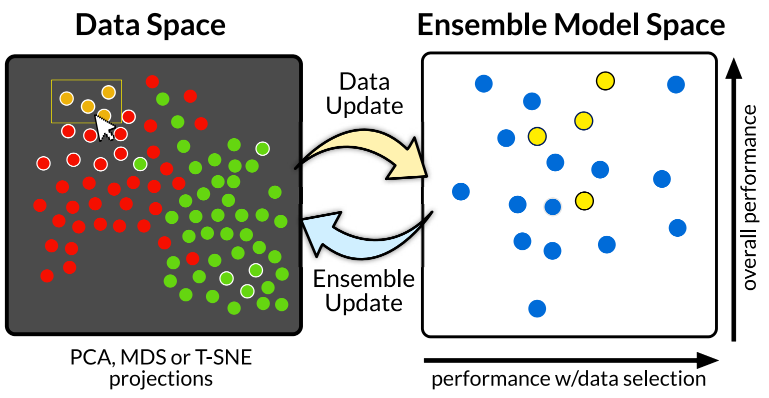


Figure 4.1: Visual integration of the data and ensemble model space. Left: The classification results are displayed in scatter plots. The user can decide between a linear (MDS, PCA) or non-linear (t-SNE) projection technique that transforms the results to a two-dimensional scatter plot. A manual selection in the data space triggers a *data selection update*. Right: The model space depicts every single model and allows to compare them by customizing the axes; herein, we contrast the overall performance with the performance w/ data selection. The interactions in the model space trigger an *ensemble update* with immediate impact on the data space.

Visual and automatic methods for the analysis of Classification outputs in Ensemble Learning do not provide a direct link from the data space back to classification model spaces with other candidates for experimenting with new ensemble configurations. Regarding the visual methods, they also do not scale properly to represent a greater number of classifiers in ensemble model spaces. For example, in [96] Silva and Ribeiro show how the models contribute individually, but the analysis is limited to inspect the ensemble after making the decision of which models will take part on it. In [101], Talbot et al. present a system in which is possible

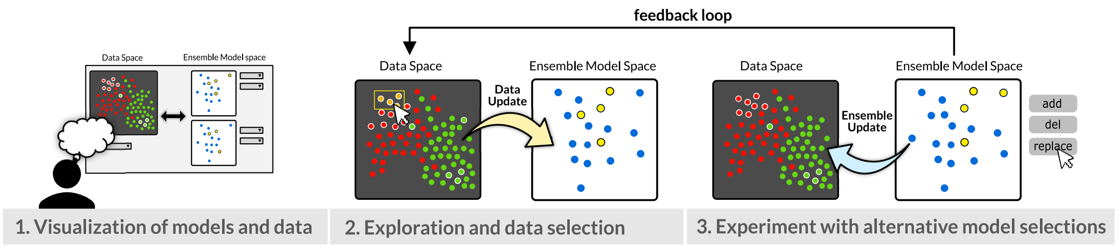


Figure 4.2: Our process for exploring ensembles of classifiers starts with the visualization of the classification outputs and the combination of models that produced the corresponding classification (1). Then, it allows the user to explore and select regions of interest in the data space (2), updates the model space to show how each model classifies the current data selection, and allow inclusion, replacement or removal of models from the ensemble (3). At each change in the ensemble configuration, the visualization of the classification outputs is updated accordingly, which introduces a feedback loop that can lead to new rounds of interaction with the system.

to interact and combine models and their classification outputs through *confusion matrices*, but with a limited set of model candidates. However, we can build classification data spaces and connect them with model spaces covering a wide range of the parameter space for the classification problem at hand. By linking model and data spaces, we foster an analysis process with a feedback loop that allows the effective exploration of these spaces driven by the user notion of importance. To the best of our knowledge, this workflow is not supported by any visual or automatic method for analyzing and exploring ensembles of classifiers.

In this work, we aim to address the research question: *How to integrate data and model space to enable visual analysis of classification results in terms of errors in Ensemble Learning?*

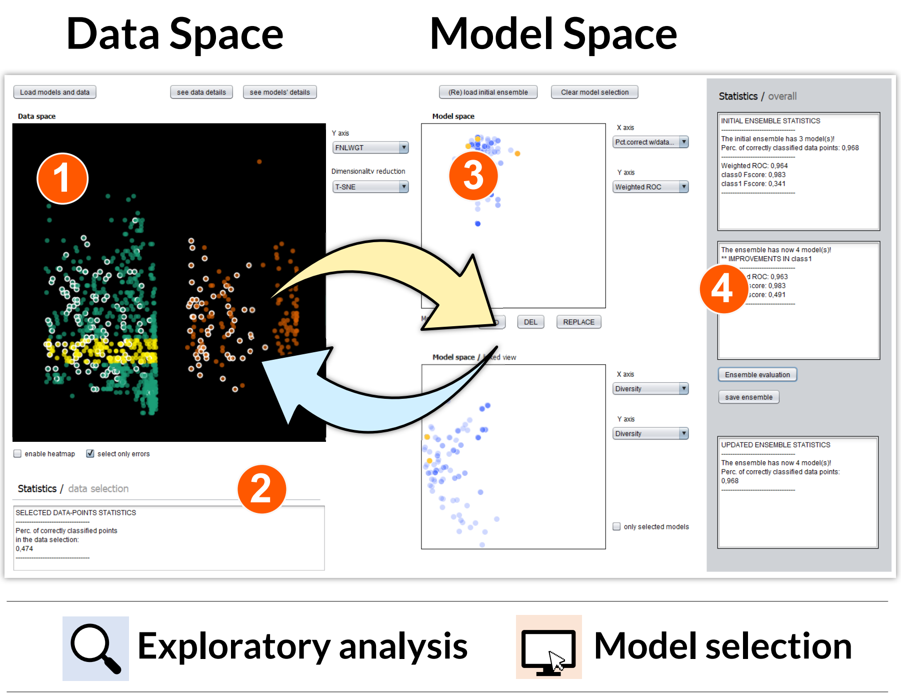
We propose an interactive visual approach for the exploration of classification results (data space) in close integration with the model space. Its main goal is to give direct access to models in classifier ensembles, thus enabling to experiment with alternative configurations and seek for local classification patterns that are not visible through aggregate measures. We visualize each classified data point and then provide direct access to each individual model that is part of the ensemble. Figure 4.1 depicts our approach. We use data and planar projections to reveal linear (PCA [44], MDS [26]) or non-linear patterns (t-SNE [68]) in the data points or models. Besides, we also offer binned visualizations of the data space to show characteristics of each class that the projections can occlude. The data points are binned per class label and data dimension (or data similarity, see details in Figure 4.4). This representation enables the identification in each class of local areas of classification errors and areas of high classification certainty or uncertainty, respectively.

In this work, we claim the following two-fold contribution towards enabling the visual analysis of classification results in Ensemble Learning: First, *the tight visual integration of the data and the model space.* Second, *a workflow that builds upon the visual integration and enables the effective exploration of models and classification outputs.* The visual integration allows to manipulate and explore the impact of each data object and model in a straight-forward manner. Key to our concept is the role of the user, who aims at forming hypotheses and gaining new insight based on the task at hand. Therefore, we relate our contributions to existing theoretical frameworks on human-centered machine learning and provide visual guidance to identify effective models not selected by the automatic search in first place. One can then experiment with alternative ensemble model selections and seek for local improvements based on the constraint that the overall performance is not impaired. The views update on the fly, enabling the user to retrace the impact on the classification outputs. We apply our approach to binary and multiclass classification problems.

Our target users are model developers that can benefit from the explorative capabilities of our approach, as well as domain experts in their classification problem of choice. These experts can express preferences concerning one class or region of the data space, and our tool takes care of finding a proper combination of models to fit these needs.

4.2 Related Work

Our work builds upon the idea of visually integrating the space of machine learning models and the data space, thus enabling the exploration of the impact of each data object and model. Following, we discuss related work from ensemble learning and interactive model space visualization. Our approach does not aim at retraining



SELECTED DATA-POINTS STATISTICS

Perc. of correctly classified points
in the data selection:
0,474

INITIAL ENSEMBLE STATISTICS

The initial ensemble has 3 model(s)!
Perc. of correctly classified data points: 0,968

Weighted ROC: 0,964
class0 Fscore: 0,983
class1 Fscore: 0,341

The ensemble has now 4 model(s)!
** IMPROVEMENTS IN class1
----
d ROC: 0,963
score: 0,983
score: 0,491

UPDATED ENSEMBLE STATISTICS

The ensemble has now 4 model(s)!
Perc. of correctly classified data points:
0,968

Figure 4.3: Overview of our tool. The user can select regions of the classification outputs (1), see how the models classified these areas (2), interact with a preloaded collection of models adding, replacing or removing them to update the ensemble (3), and track the performance while interacting with the models (4). Above, we also show the icons we use throughout this Chapter to identify the two tasks we support: *Exploratory analysis* of data and models spaces, and *Model Selection*.

the models but at finding effective model combinations that were not given by the automatic search.

4.2.1 Ensemble Learning

Classifier ensembles aim at combining the strengths of each classification model. To build ensembles, it is necessary to generate a variety of models and then to combine their results. The first step – generating the diversity of models – can be accomplished by making use of different strategies. Several ensemble learning philosophies [46] and methods for combining the classification outputs [77] exist. For example, the same model can be trained successively with different subsets of the data [11, 34], with different types of models [57, 110] (e.g. Decision trees, K-nearest neighbors), or with combinations of strategies such as the mixture of bagging [11] and random combinations of strategies in the Random Forests [12].

In our case, we follow the strategy of producing distinct types of model. With multiple types of classifiers, it is necessary to define which of these types will take part in the ensemble, and this model generation procedure and the multitude of possible combinations motivated the use of data visualization to support this task. Conversely, the model diversity produced by the other strategies is often given by the design of the respective algorithms. In these cases, it is only necessary to set a base classifier, and an automatic process generates all the other models in the background (e.g., the AdaBoost M1 method [34], in which usually Decision Stump trees are the base classifier to produce ensembles using a boosting strategy).

In particular, we worked with Multiple Classifier Systems, in which there is an overproduction phase and the generation of big model libraries (with hundreds or even thousands of models because the analyst typically does not know beforehand which model types will perform well together). Then, with the big model libraries, there are several search algorithms that were developed to look for the best possible combination of models automatically (e.g. GRASP [66, 112], evolutionary algorithms [1]), without experimenting with all the possible combinations due to the complexity of this combinatorial problem. In our work, we use a search selection algorithm developed by Caruana et al. [17]. However, using visualization and interaction we enable the user to update *on-the-fly* the ensemble selection and instantly see the changes in classification outputs. This workflow fits into the *interactive machine learning* concept presented by Amershi et al. in [3], in which the authors refer to the user updates as rapid and focused. Conversely, the fully automated selection method requires to restart the algorithm from the beginning if the user is not satisfied with the results, a time-consuming process.

4.2.2 Interactive Model Space Visualization

Following, we provide an overview of visualization techniques to represent the model space, also called the *model landscape.* Building upon the well-known visualization methods, we then discuss interactive approaches introduced to steer the performance of classifier ensembles.

Rieck et al. [84] used scatter plots for representing regression models and to perform a comparative analysis of competing models. In contrast, Olah [74] also shows groups of models using scatter plots, but for representing distinct Neural Network architectures to classify images of hand-written digits. Similarly, Padua et al. [75] represented collections of *Decision Trees* using several linked visualizations, in which the users can explore large portions of the parameter space of these models and assess the predictive quality of trees derived from several combinations of parameters. In all cases, a positive aspect of building compact visual representations of models is that one can then easily link them to the data, an essential step in better understanding machine learning models [8]. However, it is still missing the extension of these ideas to ensemble learning applications.

The analysis of classifier models through interactive visual interfaces is an active area of research [65]. Talbot et al. [101] present *EnsembleMatrix*, in which the user can interactively build and steer the performance of ensembles of classifiers. In [49], Kapoor et al. also present an interactive tool called *ManiMatrix*, in this case for the improvement of individual classifiers. In both cases, *confusion matrices* appear as a central component. In *ManiMatrix*, the users can express their preferences w.r.t. decision boundaries among classes using a confusion-matrix. In *EnsembleMatrix*, the matrices support the decision of which combination of classifiers works better when building ensembles.

However, despite the compact and efficient information about the class confusion that the matrix-based approaches convey, it is still aggregated data about performance that does not go until the bottom level of the errors with individual data points. To provide this level of access and better visualize where are the errors coming from, we worked with a representation of the data space that shows this level of detail. Also, due to scalability issues, the use of one confusion matrix for every classifier in *EnsembleMatrix* is not applicable to our case, in which we had libraries with hundreds of models for building MCS.

In *EnsembleMatrix*, the ensembles were built from a limited and small number of candidate models, and not in the same way that happens in our context of building MCS. Regarding giving access to the data instead of only showing aggregated information about model errors, Ren et al. [82] pointed recently this limitation of most current systems. They presented a solution for multiclass problems in which they visually compare different models with similar performance but with very distinct behavior w.r.t. to the classes and local regions of the data space. *ModelTracker* [4] also provided access to the data level for model performance analysis.

We go in the same direction of revealing errors that are not visible when aggregated but we do that in an ensemble learning context. While our interactive visual approach supports the overall improvement of classifier models, we mainly focus on the integration between classification results and models and propose a workflow for effective analysis. With our approach, we give direct access to any model or data point, thus enabling the direct manipulation of these objects and bringing the possibility of locally adjust the ensemble behavior accordingly to the user preferences, when several alternative model selections are possible.

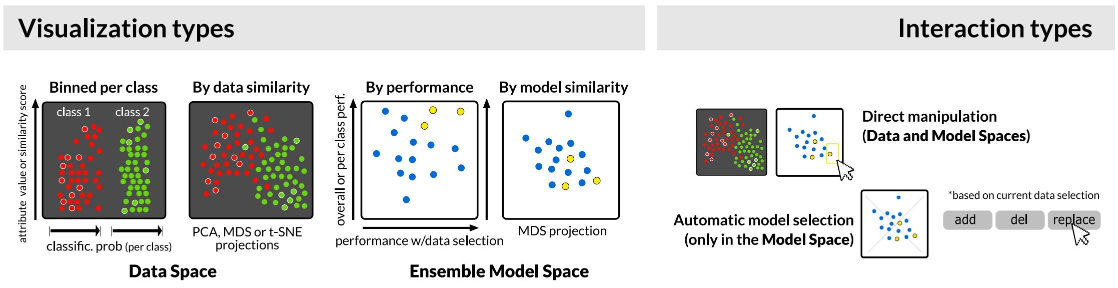

Visualization types
Interaction types
Binned per class
class 1
class 2
attribute value or similarity score
classific. prob (per class)
By data similarity
PCA, MDS or t-SNE projections
Data Space
By performance
overall or per class perf.
performance w/data selection
By model similarity
MDS projection
Ensemble Model Space
Direct manipulation (Data and Model Spaces)
Automatic model selection (only in the Model Space)
*based on current data selection
add
del
replace

Figure 4.4: Visualization and Interaction types. To visualize the data space, we provide two alternatives. The first one, *binned per class*, shows the data aligned by measures of model uncertainty and a one-dimensional similarity score or a data attribute value. The second way to visualize the classification outputs is by choosing data or planar projections. The model space also offers two visualization possibilities: the models organized by performance measures or data and planar projections of a model distance matrix. Regarding the interactions, the user can directly select models or data points in the visualization. With the models, there is the additional capability of automatically adding, removing or deleting them based on the current user selection of data, by clicking on the corresponding buttons in the interface.

4.3 Integrating Data and Classification Model Spaces

We propose a visual analytics approach for the exploration of model and data spaces in ensembles of classifiers. We work with Multiple Classifier Systems (MCS) and introduce a data-guided and user-centered process for interacting with data and models in this context (see Figure 4.2). In addition, the direct linkage of data and models is a central component of our workflow, because it allows the user to manipulate objects in any side and see the impacts on the other side instantly, by means of interaction and data visualization.

MCS are often generated from huge model libraries of several types of classifiers with different parameter settings. The process does not depend on previous knowledge about which models perform better for the data and classification task at hand. Several models are produced and an automatic search step looks for the best possible combination of models that deliver higher performance when combined in an ensemble of classifiers. In our workflow, we build a MCS using the standard automatic approach previously described. Then, we initialize our tool, in which we can visualize the models and the classification outputs produced by the initially automatically selected ensemble. Our starting point with our visual analytics approach is after the automatic construction of MCS.

In our tool, we have a visualization panel that represents the classification outputs (the data space, see Figure 4.3 (1)), and two linked other ones that show the classification models accordingly to selectable performance and diversity measures (the model space, Figure 4.3 (3)). Importantly, we show not only the models that were automatically selected and correspond to the initial ensemble configuration, but also show the whole model library that was used in the beginning of the construction. Therefore, it is possible to add, replace or remove models at any time in the ensemble. With our approach, the process of exploring the model space is driven by the user interest in particular regions of the data space. We present our model space exploration process and its feedback loop with greater detail in the next subsections.

4.3.1 Representing Models and Data

We aim at enabling the user to directly manipulate each data point and each model in our visualizations. With respect to scalability, we have to consider that the model libraries for building MCS can have hundreds of classifiers. Regarding the data space, we visualize a validation or test dataset with unseen data during the training phase of the models. In any case, models or data, we need a visualization that can accommodate these objects at scale. For this reason, we decided to use scatter plots to visualize both. In the model space, each dot corresponds to a classifier model and the color shows if the model is part of the current selected ensemble or not. Analogously, in the data space, each dot corresponds to one instance of the dataset, and the color indicates the actual label. Besides, a white outline represents mislabeled points.

In our tool (Figure 4.3), the left-side panel is the data space. The user can decide to start the exploratory data analysis and search for clusters of errors in the classification outputs by selecting to project the data dimensions to the two-dimensional space using data or planar projections.

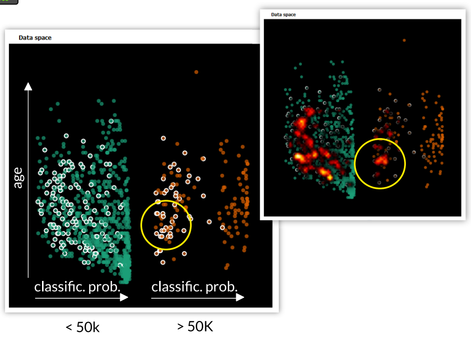
In addition to the projections, we support the exploration of classification outputs vertically aligned per data similarity or data dimension, and in both cases horizontally aligned by model uncertainty (the *Binned per class* visualization type; see details in Figure 4.4). For instance, the user can select one dimension at each time (e.g., *age*) and visualize how the ensemble classifies the data regarding this dimension (see Figure 4.5). The user can also decide to see the data points organized by a one-dimensional data similarity measure. The visualization that uses this similarity measure provides the advantage of producing one single plot for the classification outputs instead of having to alternate among plots for each data dimension while keeping the binned per class layout. To obtain the similarity score for each data point given a dataset for classification, we project all data dimensions using *PCA* and take the first component.

Still on the *Binned per class* visualization of the data space, besides using the vertical axis of the scatter plot to show the dimension or similarity score values, we also compute the classification probabilities of the predicted class for each data point and map to the horizontal axis. In addition, we display each class in a different region of the plot and side-by-side, to better distinguish the classification outputs per class (see Figure 4.5, in which we have a binary classification problem and the green and orange colors distinguish the data instances from both classes.). However, in scatter plots the overplotting can occur and make it difficult to better identify dense regions in the data. To overcome this problem, we implemented and included a heat map visualization in the data space. At any time, the user can switch between the standard scatter plot and the heat map to show the same data. With this heat map, the clusters with classification errors become more distinguishable.

Concerning the models, we represent them in our tool in the right-side scatter plots (Figure 4.3). We precompute measures of performance and diversity for each model and let the user decide which one should be assigned to each axis. We have two linked panels for the models because this layout gives more flexibility to the user. It allows the simultaneous visualization of the model library from different perspectives by assigning different measures to each of the panels. For overall performance, we compute the weighted *Area Under ROC* and weighted *F-Measure* scores. The user can also choose the *F-Measure* score per class. Regarding the model diversity, we use the *Q-statistics* [58], a pair-wise measure that compares the classification of each data point between two classifiers and captures if the models similarly classify the data or not. Then, we build a distance matrix using the *Q-statistics* measure and project this matrix to the two-dimensional space using MDS. In the end, we let the user visualize the outputs of this projection. We can read the model diversity scatter plot in the following way: models more close to each other classify the data in a similar way. Models far away from each other classify regions of the data space differently, despite the fact that they can have similar overall performance.

In the context of ensemble learning, model diversity is an important aspect. Very often, we want to find models that classify distinct regions of the data space not in the same way, because then we can combine the model strengths in a good ensemble. There is research about the role of diversity in ensembles [14] and it is not guaranteed that we can always use this measure to get the best model combination. However, it is still a relevant metric to consider when comparing and visualizing classifiers in ensemble model spaces.

Exploratory analysis
Hypothesis forming



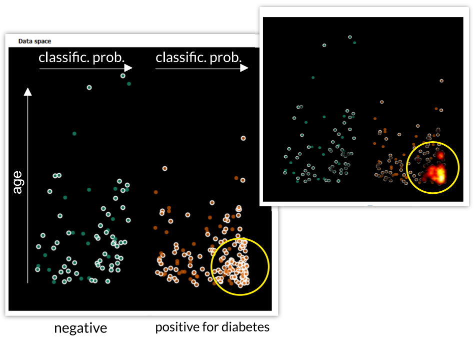


Figure 4.5: On the left pair of images, we see the classification outputs for a dataset of individuals that earned less or more than 50K per year. There is a concentration of mislabeled data points in the *>50k* class (circulated in yellow). In these cases, the ensemble assigned low classification probability, which makes these errors more easy to fix. The smaller image shows the same data with the addition of a heat map. On the right pair of images, we see the classification outputs for a dataset of individuals that tested negative or positive for diabetes. There is a cluster circulated in yellow of mislabeled data points in the *positive* class (dots with a white outline) that corresponds to individuals with lower age. In these cases, the ensemble assigned high classification probability, which makes these errors more difficult to fix.

4.3.2 Interacting with Data and Model Representations

The previously mentioned model-data linkage is a central component of our approach. With this link, we can have the visualization of the data space as an entry point for the user to find regions of interest in the data and the corresponding performance of the models for these regions. This behavior is backed, naturally, by a series of interactions that we implemented in both model and data panel visualizations. We have, at the end, a process that contains a feedback loop (Figure 4.2), in which for any data selection we have corresponding model candidates, and them for any model selection the classification outputs in the data space change again, potentially allowing new rounds of interaction.

When the user interacts with the data space and selects items, we compute the performance of each individual model for the current data selection (percentage of correctly classified data items). Then, the user can decide to use in one axis of the model scatter plots this local performance, and the model space will update accordingly to the selection. In addition, it is also possible to activate a filter that selects only the points that were wrongly classified by the ensemble. This functionality makes it easier to take care of the errors in separate, by facilitating a fast selection of these data instances. The data space always represents the classification outputs (actual class, model uncertainty, and misclassified items) for the current ensemble selection in the model space visualization.

The user can also interact with the models. This can occur in two ways: by direct manipulation, in which the user directly select/deselect models in the scatter plots, based on their position and corresponding metrics assigned to the axes. We also provide the *add*, *del* and *replace* buttons in the interface, which automatically update the ensemble selection accordingly to a previously selected region of interest in the data space (see Figure 4.4, *Interaction types*). This functionality facilitates the interaction because if we have several models close to each other in the scatter plots, the direct manipulation is not always a convenient way to precisely pick one of them in particular. To give an example of using the buttons, the user can select a cluster of errors in the data space, and then press the model *replace* button, for instance. This action triggers a method that accesses the performance of all available classifiers in the model space and returns the best and the worst performing model for the currently selected data points. Then, it removes this worst model from the ensemble and includes the best. If the best model for the current data selection is already part of the ensemble, then it is considered for reinsertion, giving it more weight in this case. We add the constraint that we only pick the best model for the current data selection if it also lies into the last two deciles of overall classification performance. If this condition is not satisfied, our search looks for the next candidate that both performs well locally and globally. Independently of the interaction type of preference, whenever an interaction occurs with the model space and the ensemble changes, we compute the classification outputs for the new model selection and update the corresponding linked panels.

The accuracy of the ensemble with the points in the data space is updated at each interaction with the models and displayed in a text panel with the percentage of correctly classified instances. This computation is very fast to do because we already have precomputed the results for each available classifier in the model space, so it is only necessary to combine the results of the selected models at each time the selection changes. We use the arithmetic mean of probabilities method, which is a standard procedure to combine classification results in an ensemble (for more details and other possible methods, see [59]). To get more significant results about the ensemble performance, we also have a button that the user can press to perform a 5-fold cross-validation evaluation and get the overall and per class ensemble model performance.

Lastly, there is also another filter that makes only the current selected classifiers in the model space accessible, to allow the detailed analysis of the models that are part of the ensemble.

4.4 The Role of the User in the Visual Exploration of Classification Model and Data Spaces

In this section, we connect our workflow with theoretical frameworks that describe the role of the user in machine learning pipelines. We identify two main tasks supported by our approach: the *exploratory analysis* 🔍 of model and data spaces

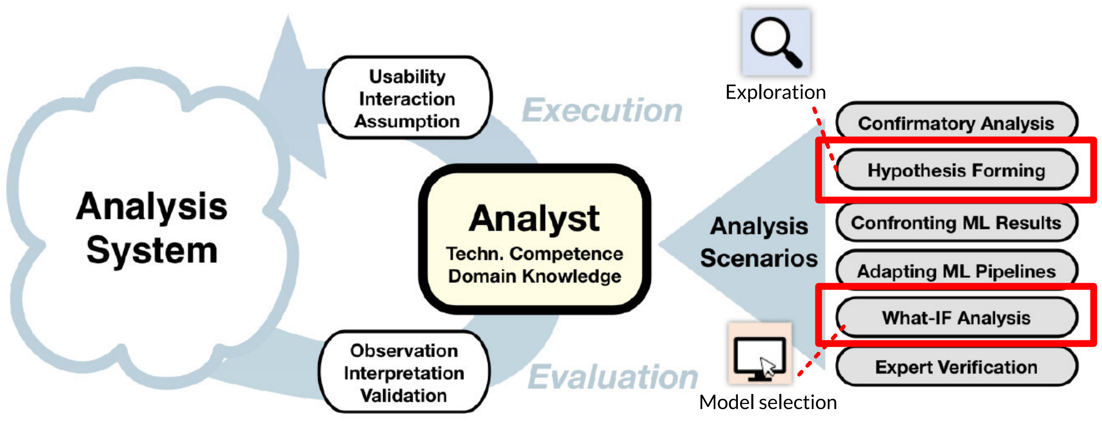


Figure 4.6: We show which analysis scenarios we support in a human-centered machine learning framework of reference. Namely, we support the *hypothesis forming* and the *WHAT-IF Analysis* scenarios (**adapted** from Sacha et al. [87]).

(*hypotheses forming*) and the ensemble *model selection*, in this case by experimenting with alternative ensemble configurations guided by the interplay between those spaces and how the data reacts to model changes (*WHAT-IF analysis*).

In [87], Sacha et al. use the expression *Human-Centered Machine Learning* to present a conceptual framework that describes human interactions with machine learning (ML) components. The authors focus on the combination of ML methods with human feedback through interactive visualization. The proposed conceptual framework fits any ML method besides classification and describes the role of the analyst in any step of a complete visual analytics/ML pipeline. Other authors also highlight opportunities to combine ML algorithms with human expert knowledge through interactive graphical interfaces. In [3], Amershi et al. discuss the role of humans in *Interactive Machine Learning*, in comparison with traditional machine learning workflows. Still in [87], the role of the analyst in a human-centered ML process loop is shown in more details and considers six analytic scenarios: *Confirmatory Analysis*, *Hypothesis Forming*, *Confronting ML Results*, *Adapting ML Pipelines*, *What-IF Analysis* and *Expert Verification*.

Two of the mentioned scenarios are within the scope of our work, namely the *Hypothesis Forming* and *What-IF Analysis*. The first one corresponds to the *Exploration* task in our categorization. The second, *What-IF Analysis*, corresponds to our *Model selection* task (see Figure 4.6).

4.4.1 Exploratory Analysis

Our workflow suggests that one can explore either the model space, the data space, or both spaces combined. This brings flexibility to the user, who can start the exploration by interacting with the data or classifier models. The exploratory analysis capabilities of our approach go hand in hand with the proposed scenario *Hypothesis Forming* introduced by Sacha et al. [87]. The main idea is to form new hypotheses without having specific knowledge about the data or the models. A means to form new hypotheses is to seek for patterns that reveal trends, clusters, outliers, or any other structure of interest. Such structures help to generate insight and provide helpful information towards understanding data and models. Following, we outline all three scenarios: the model space exploration, the data space exploration, and the combined exploration.

Model Space Exploration

The user *explores* 🔍 the model space by assigning different precomputed measures to the scatter plot axes. Going one step further, one can identify clusters of models that perform similarly by deciding to investigate their dissimilarities at a glance. To do so, the user can project the models in a two-dimensional space using MDS. In this particular case, MDS is the rational choice, because it is a linear projection technique that preserves the distances and provides a global view of the models. MDS is applied to the model distance matrix, which we derive using the pair-wise *Q-statistics* model diversity measure. This measure captures differences in the way classifiers label the data points. This way, the user can inspect the impact of the model diversity concerning a particular ensemble selection (see Figures 4.7 and 8). If all the ensemble selected models are clustered in the scatter plot representation, this indicates that this ensemble does not have a diverse set of classifiers, and performance played a major role. Such information can support the decision to not spend additional time training alternative model types that can result in new diversity score ranges. We applied our visual approach to analyze model spaces ranging from 100 to 1000 trained classifiers of different types and distinct parameter settings. One can combine these models in several ways to build ensembles.

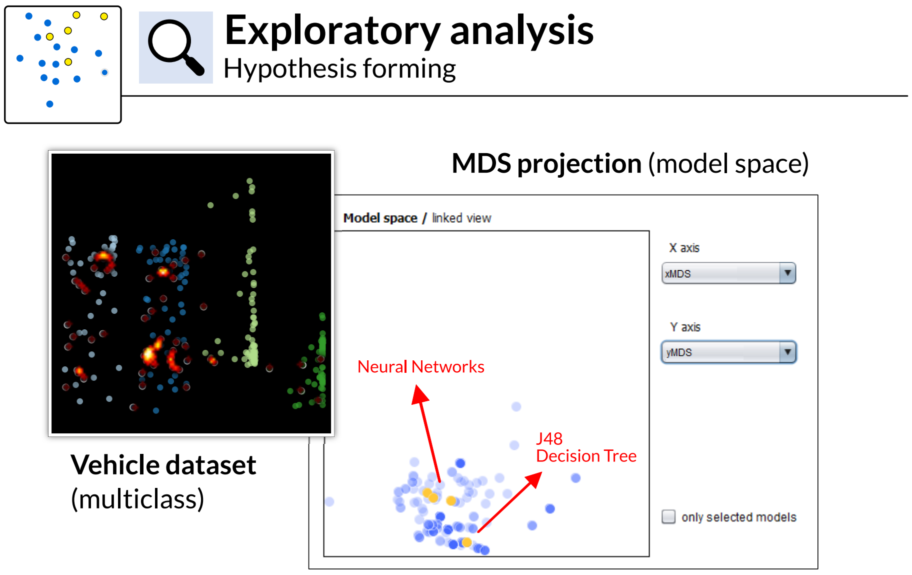


Figure 4.7: Model diversity in a multiclass problem. The right scatter plot shows an MDS projection of the models automatically selected to classify the *Vehicle* dataset in ensembles of classifiers, marked in yellow. In this case, we run the ensemble automatic selection [17] using a backward and forward search strategy. Even though the neural networks have the best overall accuracy individually and outperform decision trees, a J48 Decision Tree was also selected because in combination with the others it improves the classification globally. In our tool, the individual model performance is accessible in auxiliary text panels and linked visualizations.

Data Space Exploration

In the second scenario, and analogous to model space exploration, the user can decide to *explore* the data space and corresponding outputs of the current ensemble selection as depicted in Figure 4.5. The analysis of the classification outputs with the *binned per class* visualization type (see Figure 4.4) reveals the distribution of classification errors. At any point in time, the user can change the visual alignment of the data instances by selecting either a data dimension or the precomputed data similarity measure. While the data dimension supports the identification of classification errors that occur in a particular value range, the data similarity measure reveals errors that can be considered as similar based on the first component of

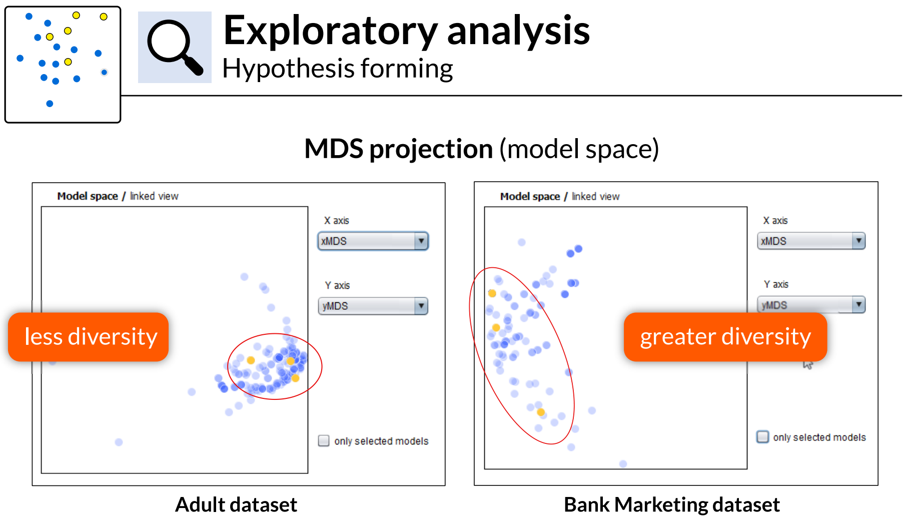


Figure 4.8: Model diversity in binary classification problems. In the above examples, we use two binary classification datasets, the *Adult* and the *Bank Marketing* datasets. Both scatter plots show MDS projections of models automatically selected to classify each dataset in ensembles of classifiers, marked in yellow. In comparison, the right case shows greater model diversity, represented by the greater distance among models in the projection. The *Bank Marketing* dataset is highly unbalanced, and most of the models with better overall performance show poor performance with the smaller class. Then, a very diverse model appears in the automatic ensemble selection because of its excellent performance with the class with fewer instances, while not hurting the ensemble overall performance.

the PCA. We choose PCA over other projection techniques because it is a linear technique, which captures global patterns of the data based on the pair-wise co-variances (a measure of the joint variability). It also tends to retain in the first component more information about the data variance than other methods. This choice provides us with the necessary means to identify similar data points, as well as outliers on a global scale, which were misclassified. Digging into the characteristics of similarly misclassified data points can also give an idea of why they are mislabeled. For example, a single model could be responsible for the misclassification of a point cluster, which is not visible in another view. Note that the application of a planar projection technique has no impact on the classification results at all. The projection does not determine the classification results but facilitates to draw conclusions based on the data characteristics. In both cases, classification errors next to the decision boundaries are typically easier to correct by experimenting with alternative ensemble selections compared with errors where the model assigned a high probability to the predicted class. To change the classification of mislabeled data instances with a high probability, more significant changes in the ensemble selection are necessary. These changes typically propagate across all classification outputs.

Besides, the user can also visualize the data and find clusters of errors by exploring two-dimensional data or planar projections. This alternative has the advantage of using both the vertical and horizontal axes of the scatter plots to represent data similarity, thus preserving more information about how similar or dissimilar are the data. In the end, we tackle a combinatorial problem with multiple candidate solutions, which are combinations of models in ensembles of classifiers. Therefore, we offer distinct views on the data, giving more possibilities to the user in identifying regions of interest in the data space that lead to that alternative model combinations.

Exploration of the Interplay Between Model and Data Space

Besides the possibility of exploring both, models and data separately, the full functionality of our approach comes to light when exploring both spaces combined (see the *feedback loop* in Figure 4.2). The user can *explore* 🔍 the reaction of the data space to the model space, and vice versa. For example, the user can investigate how the ensemble selected models perform with the most prominent clusters of errors in the data space. By choosing a region of interest in the data space, the user can adjust the model space to show the performance of all models regarding the data selection. The user can then focus on the ensemble selected models and see if the models classify clusters of errors similarly or if there are significant differences between models. This information is crucial to get an initial idea of the reason for the misclassification. Either all the ensemble models' perform poorly with the data selection or only a subset of them. In the latter case, it is more likely that an alternative ensemble selection help to fix mislabeled items in the data space.

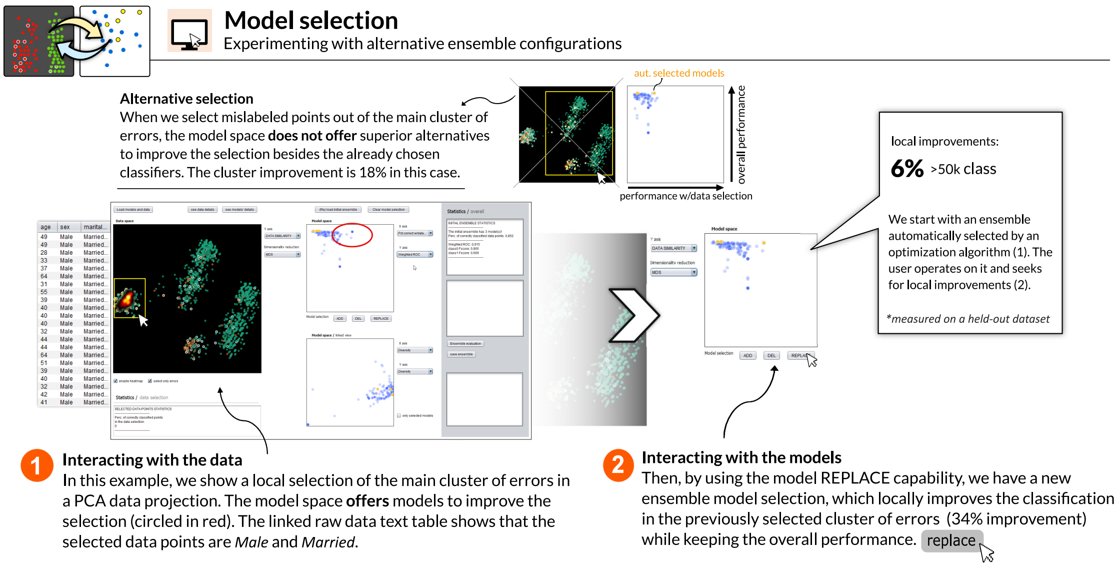
4.4.2 Model Selection in Ensemble Learning

Apart from the *Exploration* task, our workflow also supports *Model selection* . This capability brings power to our tool because it enables the user to construct ensemble models interactively. The user experiments with alternative ensemble configurations and introduces changes in the ensemble model selection, which affect the respective classification outputs. The experimentation typically occurs right after a previous exploration phase. However, here the user ends up with a different ensemble of classifiers compared with the state of the system at the beginning.

We correlate the model selection capabilities of our approach to the *What-IF Analysis* scenario described in [87]. Applying this type of exploration, the analyst can interact with the ML pipeline and observe its reaction to changes. According to our workflow for model selection in ensemble learning, the changes correspond to the modifications that are introduced by the user in the selection of classifiers in the model space.

As mentioned in Section 4.2.1, the automatic ensemble selection procedure is a huge combinatorial problem and, therefore, does not experiment with all possible model combinations. On that score, it is natural to expect alternative ensemble selections to be in favor of a given class and also preserve the overall classification performance at the same time. In a multiclass problem, the user has a greater chance to improve the performance of the model for one or another class than for all classes together.

The user notion of importance helps to prioritize and decide on trade-offs. Often, it is not possible to improve all regions of the data space at the same time. The chance of identifying clusters of errors through several visualization types, select and inspect the raw data in linked text tables and experiment with alternative ensemble model combinations with *on-the-fly* feedback about model performance is central in our approach. Thus, the user has comprehensive information at hand to decide which region of the data space should be prioritized, taking into account that frequently there is no chance to fix all points together. We refer to this process of prioritizing regions in the data space, as well as looking for alternative ensemble selections, as a task of setting up constraints in the data. Constraints, thereby, refer to areas that the user wants to protect from poor classification performance.


Model selection
Experimenting with alternative ensemble configurations
aut. selected models
overall performance
performance w/data selection
local improvements:
6% >50k class
*measured on a held-out dataset
replace

Alternative selection
When we select mislabeled points out of the main cluster of errors, the model space **does not offer** superior alternatives to improve the selection besides the already chosen classifiers. The cluster improvement is 18% in this case.

We start with an ensemble automatically selected by an optimization algorithm (1). The user operates on it and seeks for local improvements (2).

1 **Interacting with the data**
In this example, we show a local selection of the main cluster of errors in a PCA data projection. The model space **offers** models to improve the selection (circled in red). The linked raw data text table shows that the selected data points are *Male* and *Married*.

2 **Interacting with the models**
Then, by using the model REPLACE capability, we have a new ensemble model selection, which locally improves the classification in the previously selected cluster of errors (34% improvement) while keeping the overall performance.

Figure 4.9: Binary classification with the *Adult* dataset. The user interacts with the data and models to find a model selection that improves the *>50k* class without hurting the overall performance. The automatically selected ensemble (1) has a 0.91 overall performance ROC score, a 0.90 F-Measure score for the *<50k* class, and a 0.67 F-Measure for the *>50k* class. After user interaction (2), the corresponding scores with the new ensemble selection are 0.91, 0.9 (0.897) and **0.71**.

A full example and walk-through of this type of exploration appear in the following section.

4.5 Visual Analysis of Classification Results in Binary and Multiclass problems

In this section we showcase the full capabilities of our proposed workflow by building Multiple Classifier Systems (MCS) for binary and multiclass application problems. We use state of the art benchmark datasets from the UCI machine learning repository [62], and a collection of existing classification models [45] to build MCS for binary and multiclass problems. For the binary classification problem, we work with two set-ups: a model library with 1,045 classifiers, and a stratified sample with 200 models, for fast training. In both cases, we have 13 different model types and varying parameter settings. With the multiclass experiment, we work with 200 classifiers sampled from a library with 986 models. This library has eleven different model types with varying parameter settings.

We use the ensemble selection algorithm of Caruana et al. [17] to perform initial automatic ensemble selections. At the beginning of each experiment, our tool initializes with the classifier libraries, the automatic ensemble selection performed by Caruana's algorithm, and the initial classification outputs. Users can freely experiment with alternative ensemble selections in the model space, and see how the data space reacts to these changes.

In Section 4.5.1, we group and describe all we have done to prevent overfitting, a common issue that comes with the usage of ensembles. In subsections 4.5.2 and 4.5.3, we describe two experiments following the workflow presented in Figure 4.2, contemplating the interactions with models and data described in Section 4.4.

4.5.1 Overfitting and Model Generalization

Ensembles can overcome the performance of individual classifiers, but the benefit comes with the higher risk of overfitting [16]. This problem happens, for instance, when a model captures some peculiarities of the training data, such as those caused by noise in collecting the learning examples [113]. Then, these are wrongly recognized by the learner as the underlying truth. An ensemble can be complex enough to fit the training data perfectly, but too much model complexity leads to a poor generalization with new data.

To prevent the problem mentioned above, the main functionality we introduce is the chance of replacing existing models in an ensemble. By keeping it compact, we avoid excessive model complexity, which would have been the case if we employ only an additive approach. However, besides preventing the overfitting and aiming to keep an optimum global model performance with new data, we allow the user to favor one particular class as Kapoor et al. did in [49]. In this work, the authors refer to existing classification problems with distinct mislabeling costs.

Regarding the evaluation of the ensemble selection obtained after user interaction, we use two methods to assess model generalization. In the binary classification problem (Section 4.5.2), we split the data into train, validate, and test sets. The first split we use to train the individual classifiers. The second, validate data, is the one the user interacts with to adjust the ensemble and get instant updates on the model performance metrics. Then, we compare the performance of the automatically selected ensemble and the user-adjusted ensemble using a held-out test dataset and report the results in the following subsection. With the multiclass problem (Section 4.5.3), we use only the train/test data splits because the dataset is not big enough for more splits. The user interacts with the test data and modifies the ensemble. Then, we update the performance metrics using a 5-fold cross-validation evaluation on all the data.

Finally, we use bias/variance decomposition and report these measures to collect additional evidence that the user does not go in the overfitting direction after adjusting the ensemble selection. This method also allows inspecting if the efforts to avoid overfitting (variance) do not fall into the opposite error of underfitting (bias) [32].

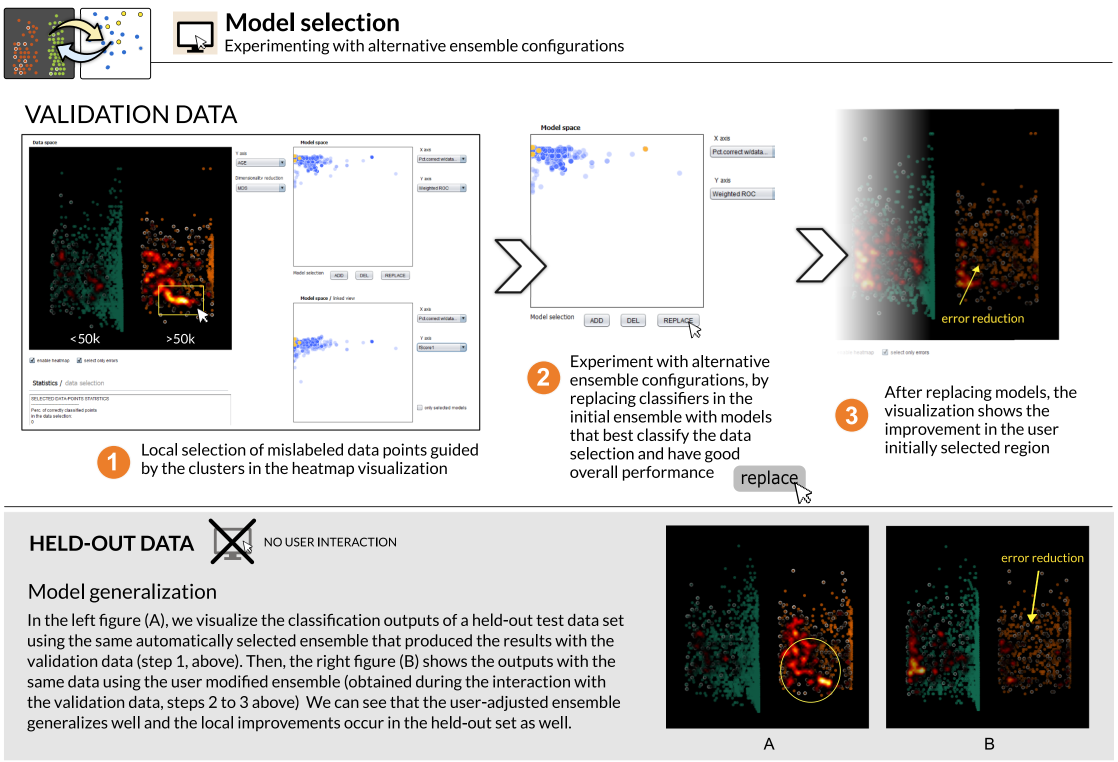
4.5.2 Ensemble Model Selection in a Binary Classification Problem

In the following experiments, we showcase how we support ensemble *model selection* in a binary classification problem, using model libraries of 200 and 1045 classifiers.

Adult dataset with a 200-models library

We use in this example the binary classification *Adult* dataset, in which the task is to predict whether the income of an individual exceeds 50k US dollars per year based on census data. We work with train, validate, and test (held-out) datasets, with 5000, 2500 and 2500 instances, respectively. To assess model generalization, we use the test (held-out) dataset to compute the performance metrics of the model obtained after user interaction with the validate set.

We start with a model library of 200 classifiers, and the automatic selection procedure gives an ensemble with three models at the beginning (a Bayes Net, a Decision Stump base model using AdaBoost and a REPTree base model with bagging). We experiment with different visualizations of the classification outputs, and the PCA projection enables us to identify clusters of errors in the data space easily. We try selecting the most prominent cluster and improve the classification



1 Local selection of mislabeled data points guided by the clusters in the heatmap visualization

2 Experiment with alternative ensemble configurations, by replacing classifiers in the initial ensemble with models that best classify the data selection and have good overall performance

3 After replacing models, the visualization shows the improvement in the user initially selected region

Model generalization

In the left figure (A), we visualize the classification outputs of a held-out test data set using the same automatically selected ensemble that produced the results with the validation data (step 1, above). Then, the right figure (B) shows the outputs with the same data using the user modified ensemble (obtained during the interaction with the validation data, steps 2 to 3 above) We can see that the user-adjusted ensemble generalizes well and the local improvements occur in the held-out set as well.

Figure 4.10: Visualizing clusters of errors and searching for alternative ensemble configurations based on user-defined regions of interest in the data space. The automatically selected ensemble (1) has a 0.91 overall performance ROC score, a 0.91 F-Measure score for the *<50k* class, and a 0.67 F-Measure for the *>50k* class. After user interaction (2), the corresponding scores with the new ensemble selection are 0.91, 0.90 and **0.72** (7% class improvement). The improvement in the selected cluster of errors is 21% in this case. Besides reporting the performances, we visualize how the interactively chosen ensemble performs with unseen data (in *HELD-OUT DATA*, A and B).

in this region (see Figure 4.9.1). After selecting the biggest cluster, we adjust the model space to show the best models for the current data selection. Next, we press the model *replace* button once (Figure 4.9.2). This interaction removes the worst model for the current data selection and searches for the best one for inclusion. In this case, the Bayes Net was replaced by another model of the same type with different parameters settings, available in the model library but not automatically selected at the beginning. These changes improve the classification of the *>50k* class by six percent while keeping the overall performance (0.91 using the *ROC score*).

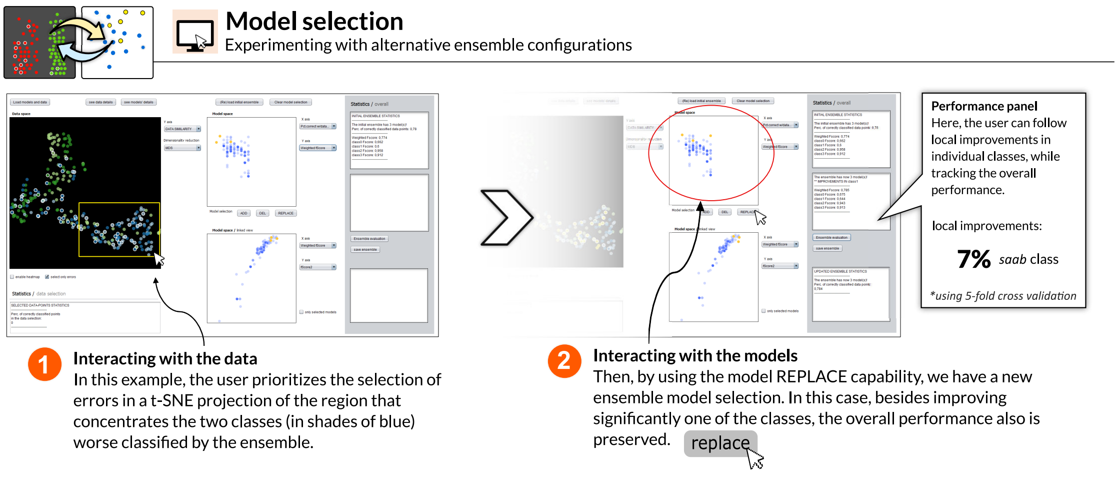

Model selection
Experimenting with alternative ensemble configurations
Performance panel
Here, the user can follow local improvements in individual classes, while tracking the overall performance.
local improvements:
7% saab class
*using 5-fold cross validation
1
Interacting with the data
In this example, the user prioritizes the selection of errors in a t-SNE projection of the region that concentrates the two classes (in shades of blue) worse classified by the ensemble.
2
Interacting with the models
Then, by using the model REPLACE capability, we have a new ensemble model selection. In this case, besides improving significantly one of the classes, the overall performance also is preserved.
replace

Figure 4.11: Multiclass classification. In this example with the four-classes *Vehicle* dataset, we describe above how the user interacts with the data and models (1) to find a better ensemble selection that improves one class without hurting the overall performance (2). We start with a model selection produced by an automatic search algorithm and explore alternative combinations using our tool.

The *Adult* dataset is unbalanced, with more instances in the first class (<50k/year). Most of the classifiers in the model library perform better with this class, which gives room for improvement with the other one (>50k/year). To assess how well the new ensemble selection generalizes after user interaction, we estimate the bias and variance for the new ensemble selection (see the table in Figure 4.12). The bias is the same, which shows that the new model selection is as accurate as the initial one. The variance almost does not change, which gives us the information that the new selection potentially generalizes well to unseen data and overfitting is not a major risk.

Adult Dataset with a 1045-models Library

In the second example with the *Adult* dataset, we use a different model setup. In this case, we use a model library with 1045 classifiers, instead of 200. The automatic ensemble selection picks seven decision tree models from the initial collection: two base-classifier trees with AdaBoostM1, three with bagging and two standard J48 decision tree models. Regarding the preparation of the datasets, we use the same setup of the previous experiment (train, validate, and test (held-out) datasets, with 5000, 2500 and 2500 instances, respectively).

We start the interactive exploration of the data space by selecting the *Age* data dimension on the vertical axis of the data space plot with the aim of finding clusters of mislabeled items by this dimension. In the visualization of the data (see Figure 4.10.1), we have the two classes positioned horizontally in opposite areas of the scatter plot. The classification probability, indicated by the horizontal position of each data point, allows the analysis of the error distribution per class. So far, we identified areas that are likely to contain more errors in the classification than others. Although, the visualization suffers from overplotting, which hinders us from perceiving the actual data point density in the corresponding regions of the scatter plot. To overcome this limitation, we enriched the point-based scatter plot visualization with an inverse distance weighting-based point density visualization [95]. The actual density of data points can be perceived efficiently using the resulting heat map. In areas with overplotting caused by clusters of mislabeled data, the heat map helps to distinguish significant variations in the quantity of these errors.

Then, we choose the class that worsens the ensemble performance (*>50k class*, Figure 4.10.1) and select the densest regions using the heat map to identify areas with clusters of mislabeled data points. To do so, we adjust the model space scatter plots to show both the performance of the ensemble with the current data selection and the overall performance. In the top-right area of the models scatter plot, the region that shows classifiers with good local and overall performance, we discover models that were not included by the automatic selection procedure (Figure 4.10.2). The automatic ensemble selection [17] does not take all possible combinations of models into account. In consequence, it is always possible to find competitive new ensemble configurations.

We start the interaction with the models by replacing the worst performing model for the current data selection. Next, we repeat this procedure by pressing the *replace* button again, backed by the performance statistics panel and the data space visualization updates. Last, we make the ensemble even more compact to reduce the chances of overfitting and press the *del* button once. This action removes from the current ensemble the worst model for the initially selected data points. We see in the data space that we reduced the errors in the *>50k class* significantly, with the cost of increasing the errors in the *<50k* class by a fraction (see details in Figure 4.10). The final ensemble selection contains a new Bayesian Network model inserted twice, together with decision trees we had at the beginning.

4.5.3 Ensemble Model Selection in a Multiclass Problem

In this example, we showcase how we support ensemble *model selection* in a multiclass problem using the *Vehicle* dataset. The classification task is to label a given silhouette as one of four types of vehicle, using a set of features extracted from the silhouette. We divide the data into 564 training and 282 test instances. The user interacts with the test data. When the model selection changes, we update *on-the-fly* the performance statistics using 5-fold cross-validation on this data. We initialize our tool with an automatically selected ensemble with three neural networks models. We use the F-Measure score throughout this experiment, both to compute the weighted overall and per class performances.

We start the exploration of the data space by trying to improve the classification in the region with most of the points of the worse performing classes (see Figure 4.11). The visualizations that use dimensionality reduction allow to inspect the mentioned region. We select among them the t-SNE projection because it provides the best class separation in this case. Next, we select the points in this region with most worse classified instances (*opel* and *saab* classes). We use the linked data text panel to confirm the data labels. We try replacing the model with the worst performance for the current data selection with the best one. We press once the button *replace* in the model space for that. Automatically, our tool removes one of the neural networks from the ensemble selection and insert one learner based on a multinomial logistic function. We still have three models after the interaction, but not all of them are neural networks anymore.

We evaluate the performance of the new ensemble selection by performing cross-validation and verify it using the statistics panel in our tool. The results are satisfactory, because one of the classes show now a significantly better classification accuracy, with 7 percent improvement (class *saab*). Meanwhile, the overall performance is still as good as the initial one, showing even a small increase (0.77 initially, and 0.78 after user interaction).

Ensemble model generalization analysis

BEFORE INTERACTION

Datasets	ensemble size	bias	variance
Adult	3 models	0.13	0.03
Vehicle	3 models	0.18	0.15
Adult (w/ a 1045-models library)	7 models	0.12	0.07

AFTER USER INTERACTION

Datasets	ensemble size	bias	variance
Adult	3 models	0.13	0.04
Vehicle	3 models	0.17	0.15
Adult (w/ a 1045-models library)	6 models	0.13	0.08

Figure 4.12: We estimate the model bias and variance before and after user interaction, with different data sets and corresponding experiments. When we compare in each case (dataset) the values before and after interaction, they almost do not change. This decomposition of the error using both measures shows that the new ensembles obtained by the user do not move substantially in the direction of overfitting (variance) nor underfitting (bias).

The *Vehicle* dataset we use in this example is balanced, with almost the same number of instances for all the classes. However, the performance of the initial automatically selected ensemble is way better with two of these classes (*bus* and *van*, 0.96 and 0.91 respectively) than the other ones (*opel* and *saab*, 0.66 and 0.60 respectively).

So, there was room for improvement in the worst classified classes. We do that by experimenting with alternative configurations using our approach, and we achieve this goal without hurting the overall performance. After user interaction, the new ensemble selection shows a performance of 0.94 and 0.91 with the *bus* and *van* classes. With the *opel* and *saab*, the new performance is 0.67 and 0.64, respectively. We observe that the benefit of improving the *saab* class comes with a small loss in performance in the *bus* class.

Very importantly, we estimate the bias and variance of the initial ensemble (before interaction) and the final model selection obtained after the user interaction (see table in Figure 4.12). The bias is almost the same, which confirms that the new ensemble keeps the overall predictive performance of the automatic model selection. Additionally, the variance is the same, which tells us that the new ensemble has the same capacity to deal with variations in unseen data distributions than the initial one. The fact that we use the *replace* feature instead of an additive approach is crucial to avoid data overfitting while bringing clear benefits to regions of the data space initially worse classified.

4.6 Discussion and Future Work

In this section, we emphasize the main strengths of our work and indicate directions for future research on the integration of classification model and data spaces.

Data vs. Feature-Space Many related works that use visualization for the inspection, attribute selection (also called *feature selection*), or in general the improvement of a part of the classification problem exist [13, 54, 70]. Those techniques stay at attribute level and allow the exploration of the attribute space with respect of the importance or the added value of an attribute. Typically, the goal is to adjust the attribute set, and in consequence the re-training of classification models, which is fundamentally different to the model selection task we are aiming at. We provide the complete data space in two-dimensional scatter plots, as illustrated in Figure 4.1, with focus on the exploration of classification errors, as these are the starting point for further adoptions of the ensemble. Compared to existing work, this effectively reduces the abstraction between the data input and the classification problem, as we omit the attribute extraction and the corresponding data transformation. Instead, we allow the user to directly work with the data that is subject to the classification, which fosters reasoning and enables findings on data record level, which is the key feature of our work.

Interactive Exploration A core part of our contribution is the manual selection of regions of the data space, which is a task that can be automated, for example, by utilizing an interestingness measure. However, the search space enumeration is a very costly operation, in terms of computing time and the required computing power, as there are many different sets of data points to enumerate. To overcome this problem, we present the user scatter plot visualizations, where visual patterns created by the point positions as well as their visual mapping, guide the user to interesting areas. Additionally, the user can bring in expert or domain knowledge about the data to make informed guesses of interesting local regions as a starting point for further examination and exploration. When it comes to interactive model selection, i.e. the adaption of the classifier ensemble, we protect the performance of the ensemble by providing a linked text panel that shows if the global performance does not get worse when the model selection changes. By doing that, we do not require the user to understand all model differences at all. Still, the user can decide to add, remove or replace existing models in the ensemble, in correspondence with the selection of a region in the data space. In the current implementation, when the user presses the model *add* or *replace* button more than once, our algorithm favors the reinsertion of the same best model for the current data selection. Future work could extend this mechanism to support optimization methods that perform a neighbor search, and offer alternatives to reinsertion.

Generalization Our approach is clearly suited for wider application beyond the use that we illustrated in Section 4.5. Formally, our proposed workflow is comprised of the classification problem, the input data and a collection of classifiers. The classification problem can be a binary or multiclass problem, which makes the workflow applicable to all kinds of classification problems. The collection of classifiers is not restricted to Multiple Classifier Systems. Random forests, or more general, any other hybrid information system, is also suitable for our approach. Additionally, we also support varying parameter spaces, varying model families and arbitrary combinations of them. In consequence, our workflow is not only suitable for the selection of model families, as we demonstrated in this Chapter, but also for parameter space exploration. An issue of classic feature-based visualizations of classification problems is scalability. Visually, we already introduced density-based heat maps as a counter-measure to be able to scale to large data sets or model spaces. Therefore, limitations are imposed only by the available computing power, and in consequence the ability to support interactions in the model and data space.

Visualization As described in Section 4.3.1, we visualize the data and model spaces using scatter plot visualizations, where each point represents a data record, or a classification model from the classifier model library, respectively. To get an overview of the classification outputs, the user can generate the two-dimensional data space scatter plots by applying state of the art dimensionality reduction, namely projection techniques such as PCA, MDS, or t-SNE. The choice of the projection technique depends on the data characteristics the user aims to consider during analysis. Therefore, we have to differentiate between linear (PCA, MDS) and non-linear (t-SNE) projection techniques. While linear techniques provide a global view on dissimilarities, non-linear techniques look at local characteristics. Then, using a linear technique always provides an overview of how all data records are connected to each other. The visual proximity between data records has a meaning, which is defined by the similarity measure. For example, MDS is based on the pair-wise distances and PCA is based on the pair-wise co-variances. In contrast, non-linear techniques, such as t-SNE, look at local dependencies, where visual distances have no specific meaning other than a separation of dissimilar data records.

We also provide *binned per class* representations as depicted, for example, in Figure 4.10. We designed our data exploration process taking into account the possibility of filtering the data space by one dimension at each time, which potentially allows the user to do meaningful selections and identify particular regions of major interest. This enables the user to deeply explore the classification outputs, understand their relationship with the selected data dimension and identify clusters of errors that could not be distinguishable only with overview visualizations (like the ones obtained with the data and planar projections). The classification results binned per class provide an additional way of finding local patterns. The bins for each class have limitations regarding their scalability, yet support a wide range of the existing classification problems (see the survey presented in [82]).

Scatter plots are prone for overplotting, which could result in potentially wrong impressions of the data distribution, as it is nearly impossible to perceive the number of overplotted data points correctly. To cope with this issue, we integrated a heat map overlay, which displays the density of data points using a continuous interpolation, mapping point density to colors ranging from black over red and yellow to white. Alternatives to this approach, such as scatter plot matrices (SPLOMs), are available, although, they do not support the idea of an integral data space visualization. Instead, they display pairwise attribute combinations, which are subsets of the data space. Similarly, small multiples or glyph-like settings are possible, but still, it has to be decided what information is shown by the visualization, as well as how to order them meaningfully. Because the two-dimensional position of the points in the scatter plots indicates their position in the data space, we use in the heat map the color of the data points to indicate errors, as we are especially interested in data points that are classified wrongly. Future work is necessary to assess how well our data and models visual integrative approach fits other visualization techniques.

Overfitting In our experiments, we noticed that adding many models to the initial automatic ensemble selection can increase the performance with a validate data subset, but the new model selection often does not generalize well, and model overfitting is a problem. In these cases, we got substantial differences in the variance values, when comparing the distinct ensemble models *before* and *after interaction*, for the same dataset. So, we recommend using the model *replace* feature, which allows local improvements while adequately generalizing with unseen data. For future work, we plan to integrate additional visualizations that facilitate the analysis of how well the model obtained through interaction performs with new and unseen data, e.g., the *bull's eye* diagram to visualize model bias and variance [32].

Evaluation We performed a quantitative evaluation of our methods. We measure the gain in performance obtained through local classification improvement of regions of the data space. We also assess how well the user-adjusted ensemble generalizes to new data. However, research on the broader impacts of giving more roles to the human in the model building process is still an open field [67]. We plan to extend the scope of the evaluation we have done so far, and perform controlled user studies to assess both qualitative and quantitative aspects related to human participation in the construction of classifiers.

4.7 Conclusion

We foster the use of visual methods for exploring model and data spaces, thus enabling the experimentation with alternative models selection in ensemble learning. Our integrative approach enables a feedback loop that keeps the user always in control of any model selection change introduced in ensembles of classifiers. We use Multiple Classifier Systems to instantiate our ideas and explore those abstract spaces. However, we can generalize and extend our workflow to any type of classifier models, combined in ensembles or not, what gives plenty of opportunities for visualization research on correlated topics.

Part II

Dataset Shift Visual Exploration (1:N)

5 Background: the Dataset Shift Problem

Contents

Parts of this chapter appear in the following publication:

- B. Schneider, M. El-Assady, and D. Keim. *DataShiftExplorer*: visualizing and comparing change in multidimensional data for supervised learning. In *15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2020* (pp. 141-148).

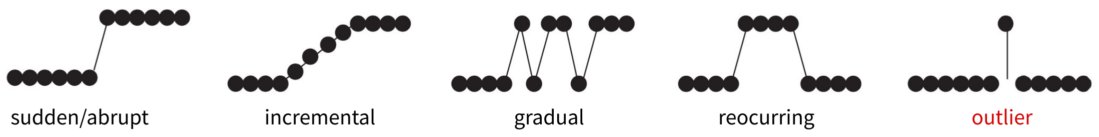


Figure 5.1: From left to right, graphical representation of possible dataset shift patterns. In particular, the last pattern (*oulier*) is not a case of data shift (**adapted** from Gama et al. [35]).

5.1 Introduction

Many machine learning algorithms follow the assumption that the training data is governed by the same distribution which the model will later be exposed [9]. However, in real-life settings, the data a system faces after deployment does not necessarily have the same properties of the data available during development. This issue is called *dataset shift* [73] (see the graphical representation of possible dataset shift patterns in Figure 5.1). It is a well-investigated real-world problem, by machine-learning practitioners and researchers [39],[99],[55].

Regarding the terminology, most of the dataset shift literature uses *training*, *validation* and *test* sets when they refer to data subsets. The first two ones are different splits of the labeled data one has at hand to build a model, by training and evaluating it. Then, the *test* set corresponds to the unlabeled data for classification after model building. The shift, when it occurs, happens between *training* and *test* sets. We prefer to use the term *unseen*, instead of *test*. We want to make a distinction because the *test* data is also used to refer to one more data split at the time of model building, in which the labels are omitted temporarily. On the other hand, our definition of *unseen* encompasses data that come after the process of model building, and are not available during its construction.

The most basic form of the dataset shift is the simple *covariate shift*, and it occurs when only the distribution of input variables change from training to unseen data [39],[99]. A typical example of covariate shift occurs in assessing the risk of future events given current scenarios, e.g., the chance of developing a disease in five years considering the patient habits in the last years to the present [100]. In contrast, the *prior probability shift* happens when only the distributions of the target output variable change [55]. A typical prior probability shift example is the classification of e-mails in *spam* or *no-spam*. When developing such a system, the proportion of messages labeled as spam for its development may differ significantly from the quantity of the spam messages that will arrive in mailboxes when the system goes into daily use. A third type of dataset shift is the *concept drift* or *concept shift* [39]. It is a more complex type and the relationships between input and target variables in the underlying classification problem change over time. An example could be the customer purchasing behavior, in which the influencing factors (input variables) on the prediction may change accordingly to the economic landscape.

The two most common causes of dataset shift are *non-stationary environments* and the *sample selection bias* [39, 73]. A non-stationary environment refers to the shift in data due to a temporal or spatial change [73]. In real-world applications, it is often the case that the data are not time nor space-stationary. An example is network intrusion detection. The second most common cause for the data shift is the sample selection bias [39],[73]. As the name implies, it happens when the selection method is biased. An example is a survey that estimates population statistics by considering a small sample of the population, which is not representative [100].

Identifying the different types of dataset-shifts is a challenging problem, and state-of-the-art approaches face issues in high-dimensional spaces [80], [43]. A common method is to use histogram plots [80] and statistical measures like the *Hellinger* distance or *Kolmogorov-Smirnov* (KS) statistics [22] to analyze training and unseen data similarities. However, with high-dimensional data, the analysis of data distribution changes per data feature may become impractical. In novelty detection, density-based algorithms estimate the likelihood that the unseen data belong to the training distribution, but without informing where the shift originated in the data feature space [80].

When one identifies the data shift, there are existing methods which try to correct or alleviate its adverse effects before having the unseen data actual classes. *Importance reweighting* [39],[80],[99] and *representation mapping* [80],[39] are the most common approaches. While the reweighting methods up-weights training data such that they look similar to unseen instances, representation mapping works on training and unseen data representations to make them appear more similar [80].

Besides all the efforts to minimize the potential drop in model accuracy, the best solution is only available when one can retrain the model [80]. Unfortunately, the latency can be huge to obtain the actual labels. Even when the labels are available, the shift in the data may be a result of sensor error, for instance, and it is not the case to incorporate it into the model. We argue that the data shift problem is still under-explored by visual analytics and information visualization researchers. Visualization can support the users while the labels are not available yet in making an informed decision about keeping the model in use. In practical situations, visualization can also help to decide when to retrain a model from scratch, especially when this is not cost-free.

5.2 An Information-Theory Standpoint

Besides previously summarizing the related work on the dataset shift problem, we anchor it through the lens of *information theory* in this section. As an integral part of data science, information theory can help us understand the relation between datasets, as well as quantify the transferability and suitability of models to given data. We can further analyze inequalities in data distribution and processing. As noted by Cover and Joy, "[..] *the data processing inequality can be used to show that no clever manipulation of the data can improve the inferences that can be made from the data.*" [25]

Given the data-shift problem, as observed in this Chapter, we can model both the training data and the unseen data distributions as alphabets. We denote $a_{training}$ for a particular set of observed points by a model and a_{unseen} for a particular set of unseen data points (that might be shifted). Both $a_{training}$ and a_{unseen} are subsets of the complete data space S, i.e., $\{a_{training}, a_{unseen}\} \subseteq S$. The degree of overlap of the two alphabets reflects the amount of *seen* data when applying the model.

To measure the distance between these two alphabets we can rely on the *relative entropy* $= (a_{training}, a_{unseen})$, also known as *Kullback–Leibler divergence* [56]. "*The relative entropy* [..] *is a measure of the inefficiency of assuming that the distribution is* [$a_{training}$] *when the true distribution is* [a_{unseen}]." [25] We can, hence, quantify the cost of assuming that a given training set $a_{training}$ and an unseen dataset a_{unseen} are interchangeable. Note that the use of $a_{training}$ refers to the data points that were considered by a model in its decision-making, and not to all input data.

Besides measuring the relative entropy, we can compute the *mutual information* [1] of the two distributions, as we assume that they are related through S. Mutual information is defined as *"the relative entropy between the joint distribution and the product distribution.* [..] *It is a measure of the amount of information that one random variable contains about another random variable. It is the reduction in the uncertainty of one random variable due to the knowledge of the other."* [25] Therefore, to quantify the uncertainty in the different cases presented in this work, we can model their mutual information as decisive factor, i.e., a dataset that has a high value of mutual information with $a_{training}$ has a lower likelihood to cause data-shift issues.

In particular, we can model the fitness of data to models following the (m:d) model to data mapping from the introduction. **(1)** For the trivial **(1:1)** case, mutual information can provide us with an indicative measure of expected performance for a classifier given on testing dataset a_{unseen}. **(2)** When choosing from multiple models for a fixed dataset a_{unseen}, in the **(m:1)** case, we can rely on the mutual information score to determine the most suitable model, as we can measure a pairwise score for the target data and all considered training instances $a_{training_j}, \forall j = (0, .., m)$. Such a score can not only help us assess the compatibility of different model to a dataset, but in particular, help in comparing different models through their shared mutual information scores. **(3)** For the case of one model being deployed on multiple datasets **(1:n)**, as discussed in Part II in detail, we can measure the mutual information of $a_{training}$ to all new datasets $a_{unseen_i}, \forall i = (0, .., n)$. Such a measure would not only be used as a pairwise fitness score, but also to determine the transferability of a model to new data. Intuitively, if a model is trained on central instances, covering S, we can assume that it would be less uncertain handling a new dataset a_{unseen_i} if its data points also cluster around the central instances observed by the model. We can formalize *model transferability* as follows: given $a_{training}$ a model is transferable if the mutual information of $a_{training}$ and a_{unseen} is not lower than a given ϵ, i.e., we

[1] `http://www.scholarpedia.org/article/Mutual_information`

want to guarantee that the two distributions $a_{training}$ and a_{unseen} are not independent from each other. Lastly, **(4)** for the case of changing models and changing data **(m:n)**, we can measure for every model measure the mutual information of $a_{training_j}, \forall j = (0, .., m)$ to all new datasets $a_{unseen_i}, \forall i = (0, .., n)$. Analogous to the introduced concept of *model transferability*, we can also define *data transferability* as another concept to help in assessing the compatibility of models and datasets.

Overall, information theory gives us powerful mechanisms to model the data-shift problem. In our future work, we want to employ the concepts introduced in this section on real data and models to study approaches that may enrich our dataset shift visual exploration pipeline.

6 Visualizing and Comparing Change in Multidimensional Data

Contents

Parts of this chapter appear in the following publication:

- B. Schneider, M. El-Assady, and D. Keim. *DataShiftExplorer*: visualizing and comparing change in multidimensional data for supervised learning. In *15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2020* (pp. 141-148).

6.1 Introduction

Supervised learning is ubiquitous in diverse application domains, ranging from image recognition to health-care and disease prevention. The success of its application depends on the data used to train a model. However, even when a classification or regression model achieve high accuracy during the model building phase, their performance might drop when applied on new data that has not been seen during training. This issue is known as the *dataset shift* problem in machine learning [73], presented in the previous chapter. Common causes for this problem are *non-stationary environments* (due to temporal or spatial change) and the *sample selection bias* [39]. Under these scenarios, it is particularly helpful if we can foresee and analyze the change in new data, especially when we do not have the new data labels or cannot track the model's performance. Numerical methods can produce the same statistics for data with entirely different properties [69]. A way to reveal and convey what statistics alone can not capture is through data visualization and analytics [19, 103].

This chapter spans the design space of visualizations for multidimensional comparative data analytics. Based on this space, we identify an under-explored problem, namely, the explicit encoding [50] of change in multidimensional data. In our work, we address the general problem of *how to capture and visualize changing data properties in multidimensional data distributions*. The application to the dataset shift problem in supervised learning guided our efforts in developing a visual analytics technique to analyse change between two multidimensional datasets, the

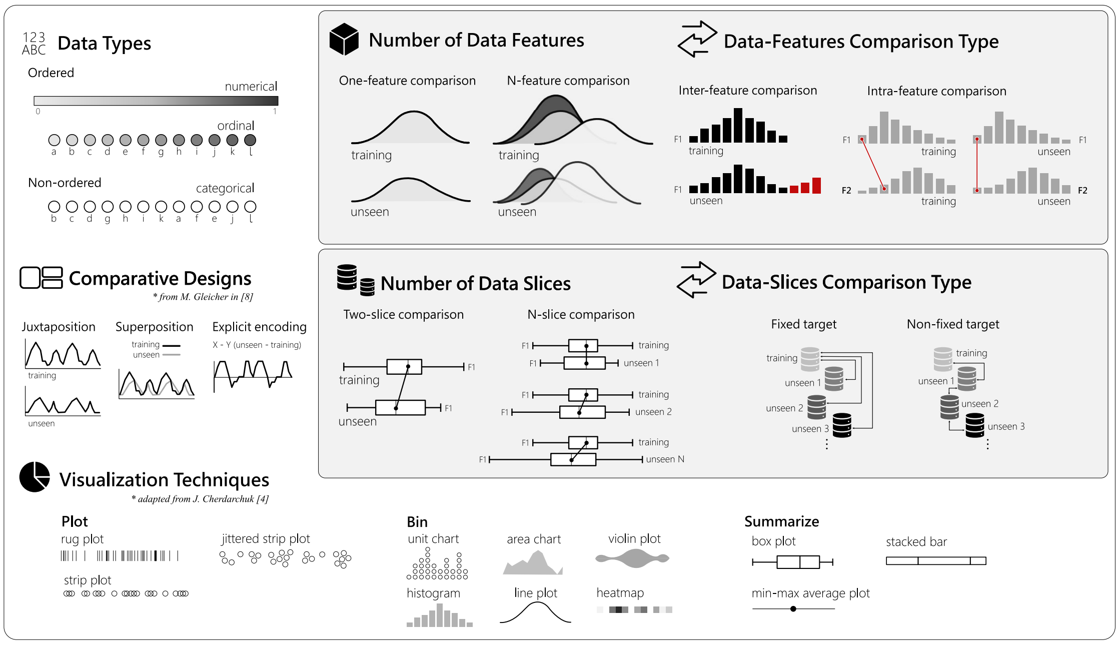


Figure 6.1: The design space for multidimensional comparative data analysis. This space spans seven dimensions to structure the design of comparative VA solutions. It provides a systematization of all aspects to be considered for the *visual comparison* of two or more data distributions. It also facilitates the identification of under-studied alternatives, opening gaps, and providing directions and opportunities for future research.

DataShiftExplorer.

We visually integrate different data representations and facilitate the comparison in a single view, instead of analyzing separate visualization components. Our approach also enables interactions, e.g., local data selections that reveal shift-patterns that conventional visualization techniques may not find. Regarding the data preparation, we simplify the representation of each multidimensional data record in order to further create a visual hierarchy that emphasizes the most recurring data structures. Hence, our tailored visual representations and data filtering enable the analysis and comparison of the changes in data distributions at different levels of detail, both, during the training and testing (with unseen data) phases of supervised learning.

In this chapter, we contribute with a **design space** for developing visual methods that explicitly encode the change in multidimensional data, for comparing two or more data subsets. Both, the visualization of data distributions, as well as the dataset shift are well-studied problems in the visual analytics and machine learning fields, respectively. However, in this work we attempt to connect both sides through an explicit visualization of data change. We present the *DataShiftExplorer*, an interactive **visual analytics technique** to identify, analyze, and compare the change in multidimensional data distributions, in general.

6.2 Related Work: Visualization of Data Distributions

The use of graphical methods for exploratory data analysis received a lot of attention over the last decades [19, 102, 103]. Among these methods, we focus on the visualizations of data distributions, alone or in comparative layouts. Approaches that enable the selection and visualization of data subsets [60] are out of our scope. We are interested in visualizing all the data. We use the categorization of data-distribution visualizations proposed by Cherdarchuk in [21]. The categories describe what to do with the data, *plot*, *bin* or *summarize* (see *Visualization Techniques* in Figure 6.1). The three categories complement each other and offer different perspectives on the same data. Individually, the techniques inside each category have limitations [24]. The author proposes a forth category, *rank*, that we consider an operation over the existing ones, and we do not use.

In the first category, *plot* the data, examples are the rug-plot [40] and the strip-plot [107]. The idea is directly representing each data point using the graphic element of choice. It provides direct information on the number of points as well as its distribution, but overplotting is an issue if the dataset is vast. Jittering[19] may alleviate that issue, by randomly changing the position of the graphical elements in the axis that does not encode the data value.

The second category of data-distribution visualizations proposed in [21] includes techniques based on binning and on the estimation of densities. The histogram [78] is a typical example of a visualization based on binning. The density-plots, another technique, produce a smoothed representation of the histogram. However, instead of directly representing the number of counts in each bin, they use more sophisticated estimation methods [97] to represent a probability density function. Variations of the density plot exist, like the violin-plot and the bean-plot [23, 41, 48, 71, 109]. They add a mirrored density plot, producing a symmetrical figure that helps in comparing certain types of distributions. Dot plots [90] also use binning but representing the counts with circles instead of rectangles. In [85], Rodrigues et al. present a technique to construct non-linear dot plots for a high dynamic range of data frequencies.

In the last category, *summarize*, box-plots [7, 109] are an example. In this visualization, the idea is representing the second and third quartiles, together with the median, to communicate where half of the data is in the distribution. Additionally, it can also show outliers, for instance, or minimum and maximum values, depending on the variations of box-plot used. Enhancements exist, like adjusting the box-plot to show skewed distributions [42]. There are plots which combine summarizing with density curves. The vase-plot [109] is an example, which shows summary statistics like the box-plot, together with the shape of the distribution for the middle-half of the data. We can also find examples in which violin-plots and bean-plots appear with additions of summary statistics on top of them.

6.3 Visual Comparisons

We propose a design-space for multidimensional comparative data analysis (details in Figure 6.1). It helps in the systematization of which are the main aspects that we should consider when approaching the problem of visualizing the change in multidimensional data, and what options are available for each of those aspects, or problem dimensions. This systematization also helps to identify options that are not explored yet in its full potential.

6.3.1 Design Space for Comparative Data Analysis

The task of comparing data distributions may sound not too complicated. However, if we consider our application context, one may need to analyze and compare an initial set of training data with a series of different incoming unseen data sets, for instance. In this analysis, it may also happen that the data has not only a few but dozens of features. So, the problem we investigate demands scalability in different directions, one or N-feature, and one or N-data slices in our particular case.

To cope with a dynamic scenario and the comparative needs our application brings, we work with seven problem dimensions. In the following, we describe one-by-one.

Data Types: the data are either *numerical*, *ordinal*, or *categorical*. The first two ones, numerical and ordinal, are ordered types. However, the ordinal data fall into categories, while the numerical data fall into a continuous or discrete numerical scale. Examples are one person's weight for numerical data or rating a hotel from one to five starts for the ordinal data type. The categorical type also falls into categories but does not have an implicit order. One example is the hair color of individuals (e.g., blonde, red, brown, and black). The type of data directly impacts which statistical methods and visualization techniques are available in each case. In [53], for instance, Kosara et al. present *Parallel sets*, a technique based on the parallel-coordinates plot, but tailored to deal with categorical data.

Number of Data Features: How many data *features* (also called data *attributes*, or *dimensions*) take part in the comparison. Each feature in each data subset (e.g., training or unseen) has its distribution. We want to compare them, and as the number of features increase, more difficult is finding a suitable visualization solution that does not overload the analyst and hinder the comparative analysis.

Data-Features Comparison Type: One can compare the distributions of the same feature in different data subsets (*inter-feature comparison*). Another possibility is comparing the relationships between two or more feature in one subset, versus the same relationships in another subset (*intra-feature comparison*). By relationships, we mean, for instance, analyzing if a given range of values in a numerical feature majorly connects to a particular range in another feature and if this behavior changes across different data subsets.

Number of Data Slices: In our application for supervised learning, typically there are two data slices, the training set, and another set of unseen data. However, an increasing number of unseen data subsets may appear in a supervised learning problem. We call this the N-slice scenario, which turns the comparison of a growing number of sets even more challenging.

Data-Slices Comparison Type: If we have more than two data slices for comparison, then one possibility is to have a fixed target, i.e., comparing the incoming sets always with the training data. Another option is comparing the next slice always with the previous one, and we call it the *non-fixed target* comparison type.

Comparative Designs: Regarding the data visualization and how to organize a layout for visual comparison, we use the categorization proposed by Gleicher in [36]. There are three possible arrangements: *juxtaposition* [108], in which different plots appear side-by-side, the *superposition*, in which two or more plots appear on top of each other, and the *explicit encoding*. In this last case, an example is the subtraction of values between two data series to obtain and visualize the change. *Explicit* in this case means that we directly represent and visualize the amount of change, instead of inferring it by looking to what is different between two side-by-side plots, for instance. The three arrangements do not exist only in isolation. One can combine them in the same visual comparison, supported by a set of visualizations.

Visualization Techniques: We consider three types from the categorization

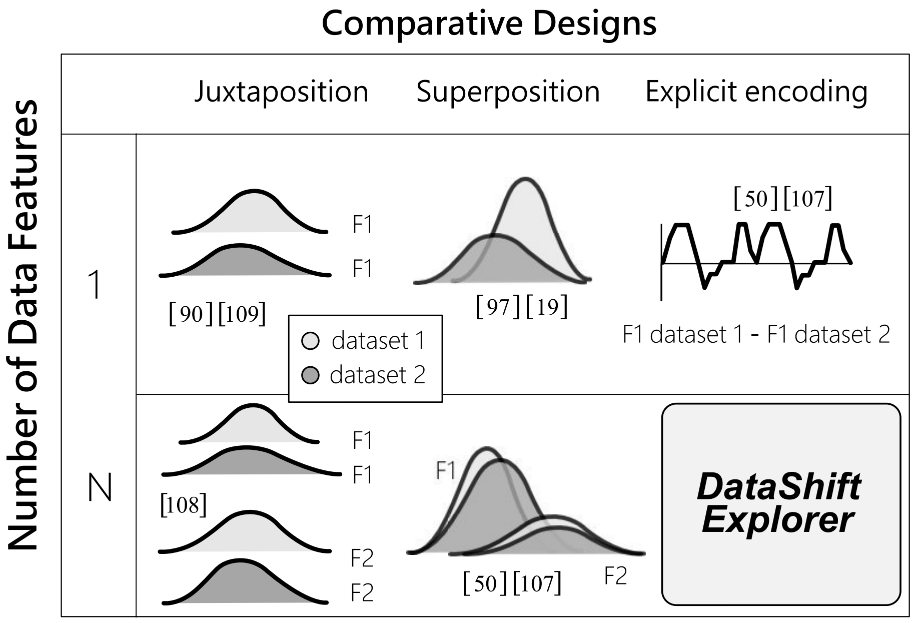


Figure 6.2: Using the terminology we propose in our design-space (Figure 6.1), we identify an under-served task in comparative data analysis, namely the explicit visualization of change in multidimensional datasets. We present the *DataShiftExplorer* to support this task.

of data-distribution visualizations proposed by Cherdarchuk in [21]: *plot*, *bin* or *summarize* the data, as we discuss in Section 6.2. They complement each other and may appear in combination, in superposed layouts, for instance. If one uses a box-plot, which *summarize* the data, the goal is not showing in detail value ranges are distributed, but a few descriptive statistics. On the other hand, to see data counts per value range, the *bin* type is an alternative, using a histogram. The first strategy, to directly *plot* the data, provides a very compact representation of the distributions, but overplotting is an issue.

6.3.2 Under-explored Comparative Approaches

The design-space we propose served to the purpose of systematizing how we approach the problem of visualizing the data shift. The problem-dimensions we present and the options inside each of those also accept combinations. For instance, the same application may contain more than one data type, combinations of comparative designs, and a set of different visualization techniques.

Among the alternatives we illustrate in the design-space figure, we want to draw attention to the intra-feature comparison type (in Figure 6.1, *Data-Features Comparison Type*). In this comparison, it is necessary to show not only the data distributions shapes in isolation but also how the values across different features and corresponding distributions connect. And not only how they are connected in a single data slice, but what are the main changes between these connections in two different data slices, for instance. That comparison is one of the goals we pursue with our *DataShiftExplorer*, which we present in the following section.

Finally, we identify, using the terminology from the design-space, a visual comparison type of multidimensional data that, to the best of our knowledge, is still under-explored (Figure 6.2), taking into account the existing techniques we revise in Section 6.2. This type is about visualizing the change in multidimensional datasets using a single visualization, instead of replicating series of visualizations for each feature and putting them side-by-side. Starting from this opportunity, we develop our approach to experiment with new ideas and solutions to support this comparison. The data preprocessing requirements and the visual-encoding decisions appear in detail in the following section.

6.4 Visualizing Data-changes at Local Level (per data feature)

We present the *DataShiftExplorer*, a visualization technique to identify and analyze the dataset shift in supervised learning. This technique is tailored to encode the change in data distributions explicitly, between the training data and the unseen set, for **N-feature comparison** in a single and compact visualization. It is also **model-agnostic** because it does not need the classification data labels, as well as it supports **binary and multiclass problems**.

6.4.1 Basic Concept of the *DataShiftExplorer*

The *DataShiftExplorer* main visualization has three components. The first one (Figure 6.4.1) is a bar chart that shows the magnitude of change between training and unseen data per feature. The second (Figure 6.4.2) is a series of superimposed

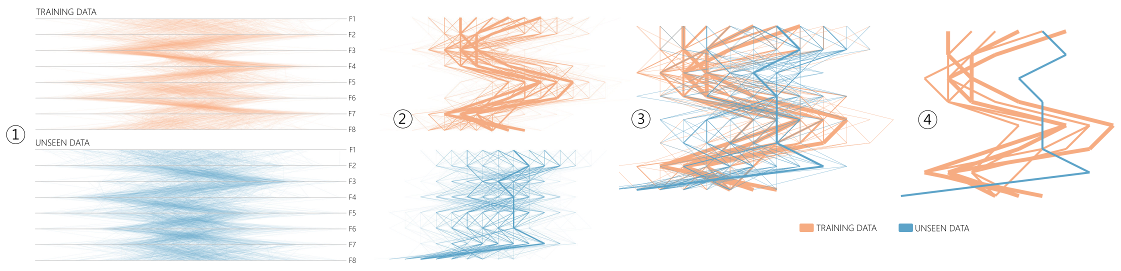


Figure 6.3: Data simplification for visual comparison. From left to right, we show how we simplify the visualization of the differences between data distributions in training and unseen data subsets. To illustrate our process, we start with two parallel-coordinates visualizations (1), side-by-side, each one with eight features. Then, we take each feature and divide them into equal bins (20 in this example), substituting the exact value per feature by the bin number. After binning, we plot the data again, setting the stroke-width to represent the number of counts of each data record (2). We use a non-linear exponential scale in this case, to visually emphasize the most recurring binned data instances. Next, we use just one visualization to over-plot both training and unseen sets (3). In the last step, we filter the data (4) to show only the data records above a threshold regarding the number of counts. All the time, we use distinct solid colors to represent both training and unseen data, with no opacity in this figure.

density plots to compare the shapes of the data distributions. Finally, the third component (Figure 6.4.3) is a tailored visualization to explore in detail the data shift patterns, composed of two layers of information. The first layer is a difference-plot, which shows the (normalized) difference of counts between data bins per feature, in unseen and training sets. In this layer, we use the *explicit encoding* comparative design. The second layer shows the connections among binned feature values in training and unseen sets using the *superposition* comparative design, for all the data instances in both sets.

In the **difference of counts** layer (Figure 6.4.3 again), we show in a very compact form, in a single plot, where are the more significant data changes between two datasets. We inform, based on a diverging color scale and dots of varying size, if new data ranges per feature appeared for the first time in the unseen data. Conversely, we show if data ranges appear in training but do not exist in unseen data. However, despite the compact form of a difference-plot, it cannot communicate the amounts of data which generated the differences. To overcome this limitation, we have an additional layer with the lines that show the connection among all the **binned feature values** for every multidimensional data record, like

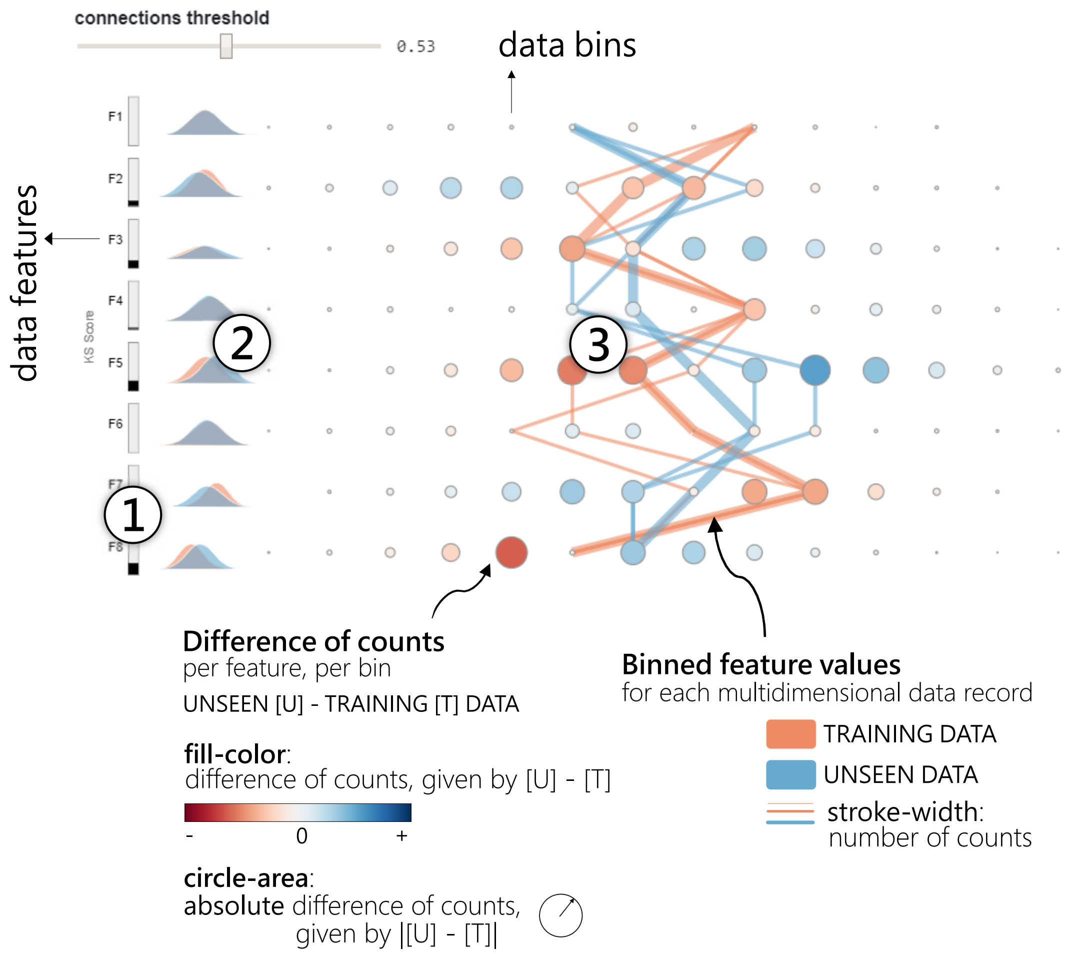


Figure 6.4: Identification and analysis of change in multidimensional data distributions. We present *DataShiftExplorer*, a visual analytics technique to analyze the data shift between two multidimensional distributions. It enables users to (1) *sort* the features by data change; (2) *compare* the shapes of the data distributions in superimposed density plots; and (3) *explore* in detail the data shift patterns in a tailored visualization, composed of a parallel-coordinates style plot and a difference-plot in a compact form. The target application domain is the comparison between training and unseen data for supervised learning.

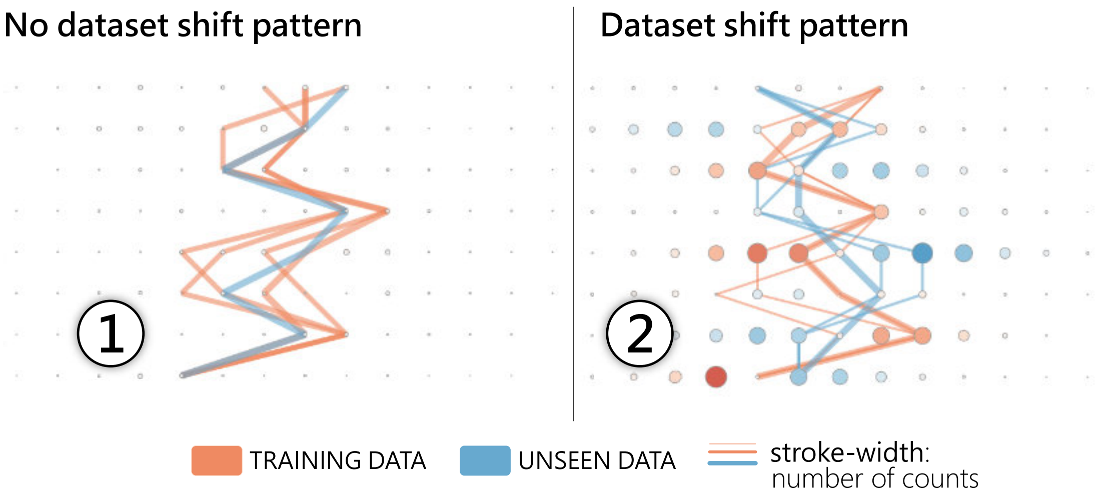


Figure 6.5: We use different datasets to show which patterns the *DataShiftExplorer* produces when there is a data change and when there is not, between training and unseen sets. In the left example (1), the lines follow a very similar path, which corresponds to the no data shift pattern. In the right (2), there are crossings between training and unseen data lines, and they follow opposite directions. We can also see blue and red circles in the difference-plot, which corresponds to the data shift pattern.

a parallel-coordinates plot. Therefore, these lines give an idea of the amount of data in each set that respond to the differences in counts.

To align the layers of information in the visualization and simplify the data representation, we use data binning in both cases (dots and lines). Figure 6.3 shows the visual impact of the data simplification process in the lines which represent feature values. Instead of using the exact value for each feature, we substitute it by to which bin the value belongs for every feature, in case of numerical data. With this new representation, we group the data by the same (transformed) value for every feature and count the number of occurrences of multidimensional data-records with that same representation. Lastly, we give more importance to the data-records with more counts, by increasing the stroke-width in the visualization using a non-linear exponential scale. This visual hierarchy has a significant impact on the simplification of the visualization, and helps the visual comparison of data in different sets. In contrast, directly plotting the data without preprocessing in bins result in overplotting.

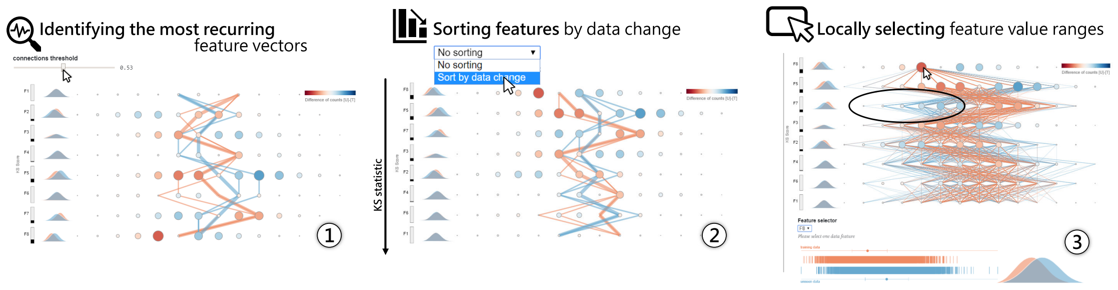

Identifying the most recurring feature vectors
connections threshold
Sorting features by data change
No sorting
No sorting
Sort by data change
KS statistic
Locally selecting feature value ranges
Feature selector
1
2
3

Figure 6.6: We use the *DataShiftExplorer* to compare training and unseen sets. We generate synthetic classification data with eight features and two classes. From left to right, we first filter the data to show only the most recurring data-instances in both training and unseen sets (1), which reduces the number of lines and the over-plotting in our main visualization. Then, we can identify the dataset shift pattern we describe in Figure 6.5. After, we sort the features to see the ones that most change next to each other in the visualization (2). We also support the task of *locally selecting feature value ranges* (3). This selection triggers a filter that shows only the data instances, both in training and unseen sets, in which the values for the data feature under analysis fit into the selection. Using this resource, we can investigate in which other features there is less overlap between training and unseen data lines in the visualization for a given selected range. Lastly, auxiliary visualizations reveal additional information on the distributions of one particular data feature.

6.4.2 Visual Encoding

We build our difference-plot (Figure 6.4.3) by first computing, for each bin (value range) in each feature, the normalized number of counts in both training (T) and unseen (U) data. Then, we build, in both cases, matrices of counts per bin per feature and compute [U] - [T], element-wise. We use a diverging color scale to map the result of the subtraction to the fill color of the dots in the difference-plot. Negative values appear in red, zero in white, and positive values in blue. Regarding the data preprocessing before the construction of our difference-plot, we normalize the number of counts using simple proportional scaling, taking into account differences of size between training and unseen data. Then, we compute [U] - [T]. However, before plotting the resulting matrix, we first look back individually in [U] and [T] counts per row. For each row, which corresponds to one feature, we take the maximum value we find. This number gives us the magnitude of change per feature, and we map this maximum (*maxPerFeature*) to the extremes of the diverging color scale before plotting each row, using [-*maxPerFeature*, *maxPerFea-*

ture]. The normalization per feature avoids that an outlier makes the changes in other features almost imperceptible.

We organize the layout of the difference-plot in the following way: each feature corresponds to one row in the visualization along the vertical axis, and each bin corresponds to one column in the horizontal axis. Then, each dot, which corresponds to one possible combination of feature and value range, has the fill color mapped to the result of the subtraction [U] - [T], normalized per feature as we explained before. Shades of blue mean that more data appear in unseen for the given feature and bin. Conversely, red corresponds to a negative value and means that, in training data, there are more counts for the respective dot than in unseen data. Also, the absolute value of unseen minus training, $\mid [U] - [T] \mid$, is mapped to the size (area) of the circles.

In the parallel-coordinates style plot (Figure 6.4.3 again) that show all feature value ranges for each data record, we use distinct solid colors to set the stroke color, salmon for training and blue for unseen data. We then use a non-linear exponential scale to map the number of counts to the stroke-width, because we want to emphasize the data-records with more counts and make the lines of the less frequent connections appear with much less visual importance. Regarding the data preprocessing for the parallel-coordinates style plot, we also use simple proportional scaling to normalize the data instances counts and consider differences of size between datasets.

6.4.3 Interaction, Data-filtering, and Details-on-demand

On top of the data preprocessing steps and visual encoding, the interaction plays an essential role in the *DataShiftExplorer*, facilitating our visual comparison task. Interactive components allow us to support three subtasks (see Figure 6.6). The first one, *identifying the most recurring feature vectors* (**T1**), works by reducing overplotting and keeping only the most recurring changes among feature values. Then, by *sorting features by data change* (**T2**), we organize the visualization layout and see significant changes first. The third subtask, *locally selecting feature value ranges* (**T3**), profits from the interaction capabilities to reveal local changes, besides the overall view.

To support the first task (**T1**), the interactive component we use is a data-filter that controls the number of lines to appear in our visualization, the ones that show the value ranges for all features in the training and unseen sets. This filter interactively updates the threshold of the minimum number of counts, so that lines representing multidimensional data instances with fewer counts than this threshold do not appear. There is a clear trade-off between seeing all data, and in this case with poor legibility for comparison, or filtering it and keeping only the most recurring structures for comparison in both sets, for better legibility.

Regarding the possibility to sort the features by data change (**T2**), we use the two-sample *Kolmogorov-Smirnov* (KS) statistic [22], which tests whether two samples are drawn from the same distribution. The KS statistics lies between 0 and 1 and works as a distance measure between data distributions. The smaller the KS statistics value is, more similar are the distributions. We precompute the KS for each feature between training and unseen sets and allow the interactive selection of this sorting criterion, which updates our visualization accordingly. As an auxiliary plot, we show next to the rectangle bars that encode the KS statistics for each feature a small superposition of two density plots, showing the shapes of the distributions in training and unseen sets. The density plots are efficient for comparing differences in shape between both distributions, as well as confirm the data shift indicated by the KS statistic.

For the third task (**T3**), we provide an interaction on mouse-click that shows, for a selected data range per feature in the difference plot, only the lines in training and unseen data that fits into this selection. It also works as a filter, but this time we are filtering by a range of values in one feature. The mouse-click on a particular circle in the difference-plot determines which feature and which value range to filter (example in Figure 6.6). Using this interaction, we can select one data range we know is new in unseen data by looking for the biggest blue circle in the difference-plot, for instance. Then, we inspect where the lines go, analyzing where there is no overlap between training and unseen data lines. This way, we identify, for a given selection, in which features and value ranges the dataset shift occurs.

Finally, we also work with three linked visualizations: one rug-plot, a simplified version of a box-plot, and one density-plot (Figure 6.6.3, on the bottom). The density plot is the same one that appear in a much smaller version next to the KS statistics rectangle bars. We use these visualizations to show the data-distribution of one feature at each time, both in training (in red color) and unseen sets (in blue color). They offer three different perspectives on the same data distributions. As we present in Section 6.3, there are the *plot*, *bin* and *summarize* data distribution visualization types, which correspond respectively to our auxiliary rug plot, density plot and simplified box-plot.

6.5 Future Work

After spanning the design-space of multidimensional comparative data analytics, we identify a potential research gap, and develop an interactive visualization prototype, the *DataShiftExplorer* [1].

However, our work has limitations. The main one is that user studies are missing. We plan to validate our prototype in controlled environments as future work. Regarding the prototype, one limitation is that we do not let the user change the number of data bins interactively. Binning is a necessary preprocessing step in our pipeline. However, a different number of bins may affect the visual outcome significantly. A necessary extension is letting the user interactively update this number.

Lastly, in the user studies, it will also be important to explore real datasets with our tool. So far, in the examples we provide in this Chapter, we generate the data using a synthetic classification data generator [2], to have complete control over the generation process.

[1] *http://datashiftexplorer.dbvis.de*

[2] *http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html*

7 Visual-based Monitoring of Data and Model Compatibility in Classification Problems

Contents

7.1 Introduction

Studying data changes in supervised learning has to do with the fact that the stability in data distribution from the training set to incoming data streams is not always the case.

"The fundamental and rather naïve assumption made by most computational intelligence approaches – that the training and testing data are sampled from the same fixed, albeit unknown, probability distribution – is simply not true." [30]

In this Chapter, I motivate the use of data visualization for the dataset shift problem through two experiments. These experiments aim to reveal a gap in visualization research applied to the dataset shift problem. Namely, I implement visual techniques with the capacity to distinguish data changes with high and with absolutely no impact on model performance and show how numerical methods in isolation are not enough for this task.

The second experiment is an evolution of the first experiment because I implement visual components that emphasize the direction of the movement of data class boundaries from training data to new data streams when it exists. However, even though my experiments make a point on how data visualization can add capacities to the numerical analysis of data changes, they have limitations. The main restriction is that they may not work with non-linearly separable data.

Therefore, I revise the state-of-the-art applications in the field and propose how to evolve towards a general-purpose *visual-based monitoring of data and model compatibility in classification problems*. The proposal comes from the gaps in STAR applications and the limitations I identify in my experiments.

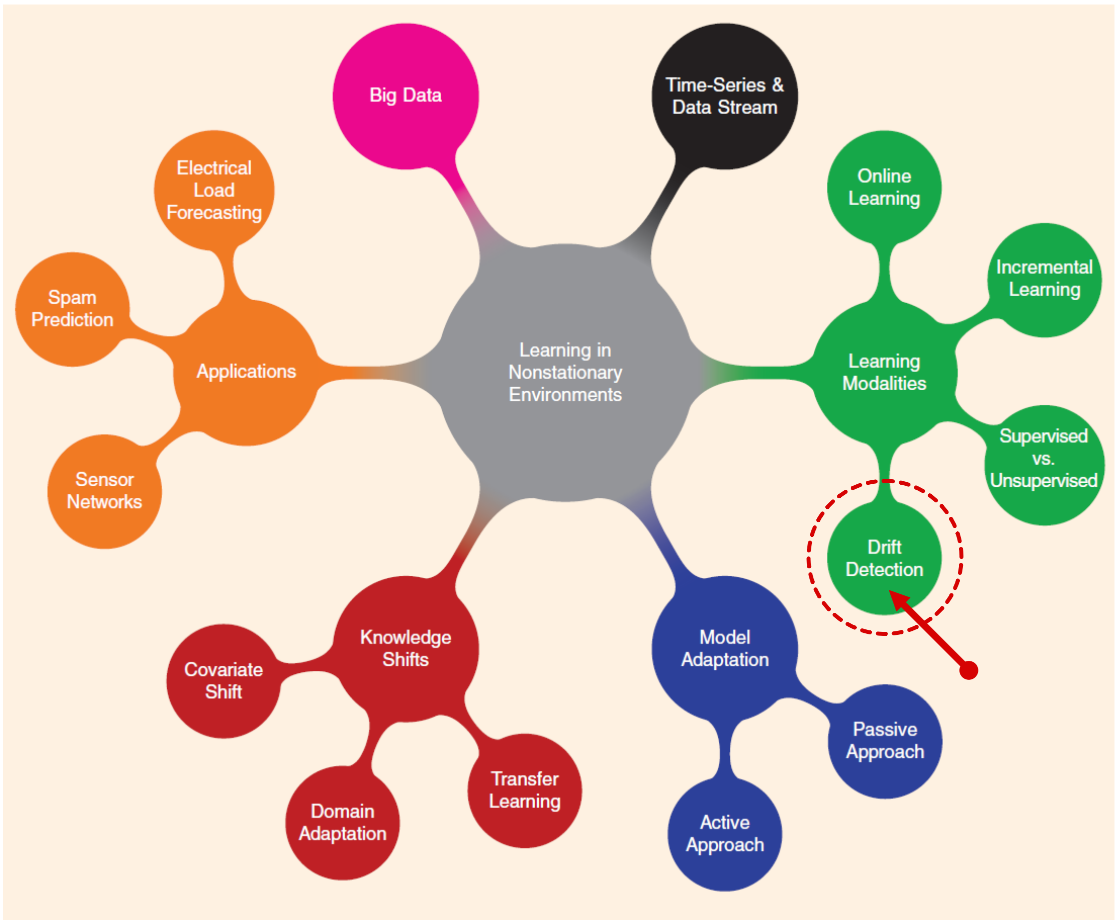


Figure 7.1: In this mindmap, which describes how the topic of *learning in nonstationary environments* connects with machine learning techniques and the application domain, I highlight (in red color) the most related subtopic in *Learning Modalities* I developed in this part of the thesis (**adapted** from Ditzler et al. [30]).

7.2 Why Use Visualization to Detect the Dataset Shift Problem?

Although automatic drift detection is a feature that machine-learning pipelines automate, the latency related to the time necessary to obtain incoming data labels keeps the door open for research on visual methods on this problem. The new labels can arrive in irregular time intervals, so the decision on when and how (e.g., with all data? portions of it?) to retrain a model still depends on human expert knowledge. As usual, visualization and interaction offer the chance to work as a

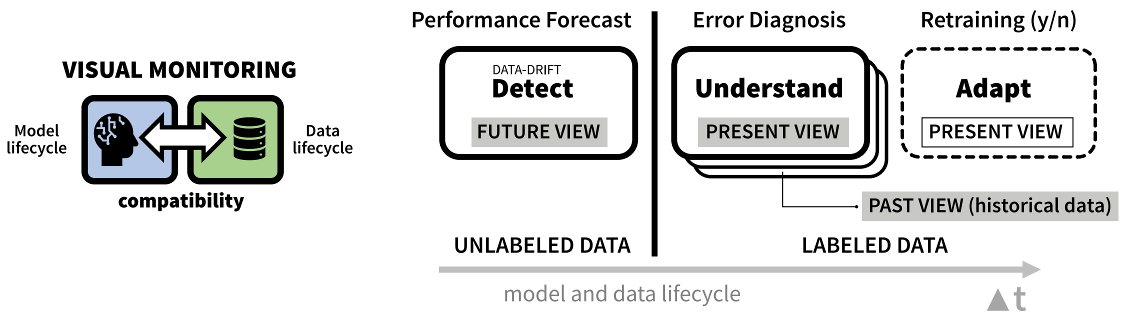


Figure 7.2: Visual-based Monitoring of Data and Model Compatibility in Classification Problems. The three tasks above (*detect*, *understand*, and *adapt*) correspond to the main macro-functionalities that a general-purpose data-model compatibility monitoring system should address. In the *future view* at the detection step, the system should be able to foresee drops in classifier model performance before the labels of incoming data are available.

layer between the specialist and models/data.

Regarding the scope, my proposal of visual-based monitoring of data and model compatibility extends the application initially targeted at the dataset shift problem in machine learning. It also considers the case of inspecting and visually comparing how different model candidates generalize to new data. Therefore, I prefer to use the term *model-data compatibility* instead of narrowing it only to the case where one expects the presence of the data drifts. An example scenario is when one does not expect significant data distribution changes, in which a specialized model performs better than another built to adapt to data shifts in general. In this case, a trusty monitoring system of model-data compatibility over time could support the decision to keep the best-performing model while protecting against events that, even with a low probability, may happen.

The *visual-based monitoring of data and model compatibility* idea aims to look at future, present, and past model behavior in an integrated analysis environment. In greater detail, an adaptive visual layout in such a system should accommodate the future view (no data labels available yet for incoming data), the present view (model errors since the last evaluation with labeled data and without retraining the model), and a past view to accumulating historical data before the last model retraining. With this layout, a general-purpose system could accommodate how many data past views the application requires, depending on the user's needs and the frequency of data updates (Figure 7.2).

Regarding the workflow, a continuous visual-based monitoring system should support (at least) three tasks in supervised learning classification: *Detect*, *Understand* and *Adapt* the model (see Figure 7.2 again). Importantly, this proposal does not target any specific data domain. Instead, it provides a general description of what needs to be considered to build an efficient system without overspecifying components and applicability, which could limit generalization.

The *detect* task receives unlabeled data. It can apply automatic measures to provide data distribution change scores based on input feature distributions. The unavailability of incoming data labels at this point of the workflow justifies using data-distribution-based detection methods instead of model-performance-based ones. Still, unlabeled data do not allow for precisely computing class distribution changes (the prior probability shift). The maximum the system can do is use the class obtained from the model classification to do this computation, with the risk of model errors. On the visual side, a data visualization component can help to support foreseeing model performance with new data without its labels (see an example with linear data in the experiments in Section 7.5). Visualization helps confirm or contest the hypotheses of a drop in performance brought by automatic scores in the case of data drifts.

Understand and *adapt* tasks receive labeled data. *Understand* supports both present and past model behavior views and enables the diagnosis of model errors by conjugating visual and non-visual components to analyze error causes (e.g., data drifts, proximity to decision boundaries). *Past* is only an accumulation of historical model performance data.

After understanding the model errors, an analyst may decide to *adapt* the model. This last component is optional, as the possibility of no-retraining is also a possible outcome from the analysis in the previous diagnosing step. The adaption task contemplates the chance of retraining the model with all data, portions of it (e.g., an analyst may discard outliers), or even fitting the data to an alternative model candidate. There are automatic methods for model adaptation. Here I focus on the opportunities for human intervention by using a visual analytics system that offers the chance to confirm which data should be considered for retraining.

In addition to data inputs, model information is crucial to analyze the impact of data changes in a particular application context, given the differences in model behavior depending on the learning strategies. Therefore, model decision-boundaries built during the training phase should work as companions in all the three supported tasks. Even in the first task (*detect*), when the incoming data labels are not available, overlapping strategies of both train and new data (and the corresponding model decision-boundaries) serve to analyze the varying risks of misclassification given the relative position to that boundaries.

Lastly, my proposal could also embrace the model confidence scores in classification (for each data point, a model-dependent input produced by learners like decision trees). Visually, we can directly map in visualizations the confidence scores, when they exist, to communicate model uncertainty at the first step without data labels, the *detect* task.

7.3 State-of-the-Art in Visualizing the Data Changes

In this section, I review the literature and identify the most relevant works presenting visual methods to support the dataset shift problem. Then, I use the graphic representation of my *visual-based monitoring of data and model compatibility* to highlight which parts are covered by state-of-the-art (STAR) applications. I considered the *ACM Digital Library* and the *IEEE Xplore* in my review. With special attention, I revisited all the research papers that matched my search criteria related to the intersection between research on machine learning, dataset shift and visualization/visual analytics. I considered all works presented throughout the years in *IEEE VIS* and *EuroVis* Conferences (main tracks, all the co-occurring and corresponding Journals). From the selected papers, I analyzed all the works cited by them.

In [105], Wang et al. present *ConceptExplorer* (Figure 7.3), a visual-analysis system of concept drifts focused on multi-source time-series data. The detection of the changes depends on labeled data because it uses model accuracy score drops in the identification process. Therefore, it does not implement the detection task independently of label availability in incoming data. The system has an explanation view that compares concepts pairwise through a correlation matrix. The goal is not to understand model errors, but still, visualization plays a role in supporting explanations. In this case, concept analysis and comparison are based

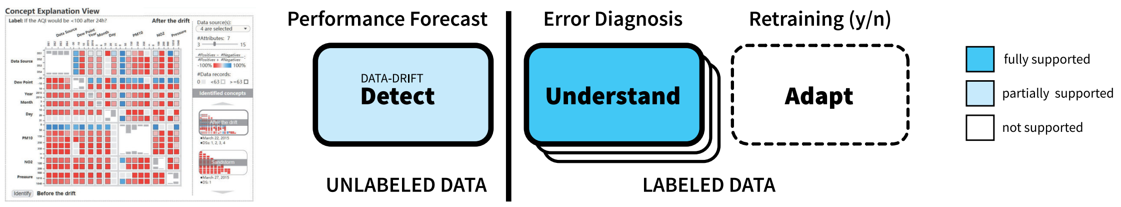


Figure 7.3: *ConceptExplorer* [105] fosters the visual analysis of concept drifts on multi-source times series data. It provides functionalities to compare concepts and identify correlations among them. It does not have the purpose of adapting the model after identifying data changes.

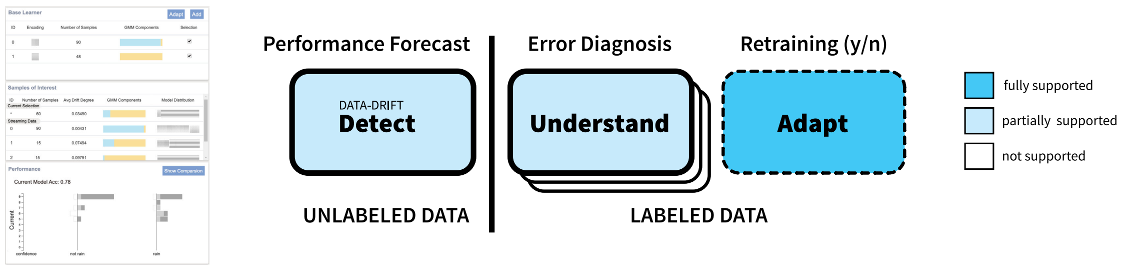


Figure 7.4: *DriftVis* [111] implements, to a certain extent, functionalities that support detecting, explaining, and correcting concept drifts in data streams. Among the three systems analyzed in this section, it is the only one that provides the model adaption component based on an ensemble method that does not require model retraining.

on correlations. Though the model adaption step is not the focus of this work, this component is not supported.

In [111] the authors present *DriftVis* (Figure 7.4), a visual analytics system that supports detecting, explaining, and correcting concept drifts in data streams. The correction module implements an ensemble method that allows model adaptation without needing to retrain with all data by adapting the weights of base learners trained on different subsets of training data. Regarding the understanding task, *DriftVis* uses a model-agnostic approach in which the goal is not to understand specific failures of model types with parameters and features that do not generalize. Instead, it considers exploring shifts in data distributions, which makes the system model independent. Lastly, in the detection component, the system supports identifying data drifts without needing labeled data. It uses data-distribution-based methods, particularly the energy distance measure, to compute drift degrees. The limitations are the impossibility of detecting prior probability shifts (changes

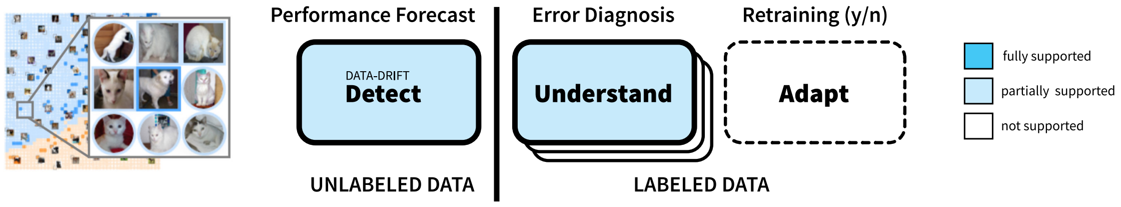


Figure 7.5: *OoDAnalyzer* [20] identifies out-of-distribution (OoD) samples in image data. It displays the thumbnails of the images detected as OoD based on an optimized grid layout on top of a t-SNE projection that shows all the data points. Part of the solution supports only image datasets.

in the proportion of the target variable) using the method and the impossibility of foreseeing changes with no impact on performance due to the limitations of the chosen t-SNE visualizations in preserving global data similarity relationships.

The last work I selected allows the identification of out-of-distribution (OoD) samples in image data [20]. *OoDAnalyzer* (Figure 7.5) performs OoD detection using deep ensemble methods without needing data labels. It also focuses on explaining them in context using visualization strategies to reveal the distribution difference between training and test sets. The system does not focus on adapting the model to data changes. However, it displays the corresponding image thumbnails of OoD samples in image datasets, thus allowing the manual labeling by expert users and further model retraining for performance improvement. Regarding the detection task, it also uses t-SNE as the base method before organizing the data points using a grid strategy to optimize the positioning of image thumbnails, minimizing overlap. Again, the known limitations of t-SNE visualizations in preserving global data similarity relationships limit the chance of visually identifying OoD samples that do not hurt model performance, independently of the OoD degree (score) computed by *OoDAnalyzer*.

7.4 First Experiment: Simple Example with Linearly Separable Data

After revisiting the literature on how visualization/visual analytics can support the dataset shift problem, I present experiments to reinforce the utility of visual methods. I aim to make a point about the limitations of data changing stats in isolation as a method to foresee impacts on performance caused by those shifts. In this first experiment, we visualize the direction of the expansion of data class boundaries when new data arrive. Then, we correlate those movements with the impact on models trained without information about changes in those boundaries.

Data Preparation We use the *scikit-learn* [76] classification data generator to have complete control over the scenarios we built. This data generator creates clusters of points that are normally distributed, introduces interdependence between these features, and allows to add various types of noise to the data. The number of classes, features, class-proportions, and the magnitude of the shift are parameters of the *scikit-learn* data generator we use. We manipulate different parameters to simulate the distinct types of data-shift we select. In the case of the covariate-shift, we change only the shift parameter in the data generation process, keeping all the other data properties the same. Then, from the shifted data-split, we sample data to build the **U** set. For the generation of the prior-probability shift scenarios (or simply *y-shift*), we manipulate only the class proportions in both **T** and **U**. Finally, in the simulations that introduce both types of shift, we use the class proportions and shift parameters of the data generator.

We explore how distinct data-shift spatial configurations impact the classification results. To obtain these spatial configurations, we divide this experiment in two parts. In the first one, we manipulate 25 different combinations of class proportions in shifted data, from which we reproduce five canonical examples in Figure 7.6. We generate eight features regarding the data generation process, and we apply a fixed amount of shift in half of them, based on the drops in performances and corresponding necessary amounts of change we got in the previous experiments. We use a Gradient Boosting Classifier, which can deal with noise in data to a certain extent, and what we modify in each run is the proportion of each class in the shifted data, which belong to the *unseen* subset. We work with binary and multiclass problems, the latter with three and four classes. Each scatter plot (Fig. 7.6) is the result of different class proportions in the shifted data. The unseen dataset has always half of its instances with no shift and equal class proportion, and the other half with shift, and varying proportions for each

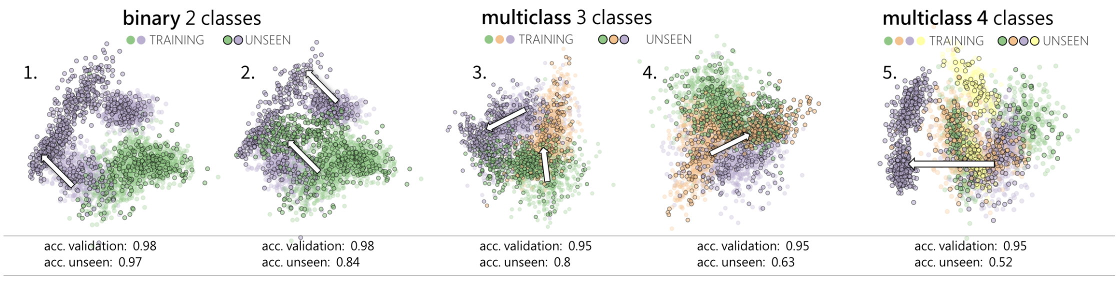


Figure 7.6: We visualize dataset-shift patterns accordingly to how the unseen data (colored dots with black stroke) overlap with the training data (colored with no stroke), in binary and multiclass problems. We use the PCA linear projection method and visualize the first two components in the scatter plots. Depending on the position of each class in unseen data in comparison with the position of the same class and the other classes in training data, there is a more significant or smaller drop in performance. We do not visualize the validation set. It serves only to report on the model performance before trying to classify the unseen shifted data and to analyze the differences.

class in this experiment. In the binary case, for example, we explore three possibilities: only class 1, only class 2 or half of both. Analogously, with three and four classes, we also explore the presence or absence of each class in all possible combinations, together with the equally balanced case containing all classes. Regarding the proportion of classes, when they are equally balanced in unseen data we have the simple covariate shift type in which this proportion does not change, only the input features.

The second part of this experiment is a short extension, to produce cases of the prior probability shift, when just the target variable changes, but there is no change in the input features To achieve that, we do not apply any shift to the features, but then start with a heavy unbalanced training data set and manipulate the unseen set to have a completely different proportion of classes. We do that also for binary and multiclass problems with three and four classes. The results are analogous to the first part of this experiment, and we summarize them in Figure 7.7.

We identify in our experiments two dominating movement patterns: i) the *contiguous* (**C**) class expansion in unseen data to an adjacent area with no overlap with the training data, in which there is no conflict with the other classes. ii) the *over-other-class*(es) (**OC**) pattern, where the expansion happens into the direction

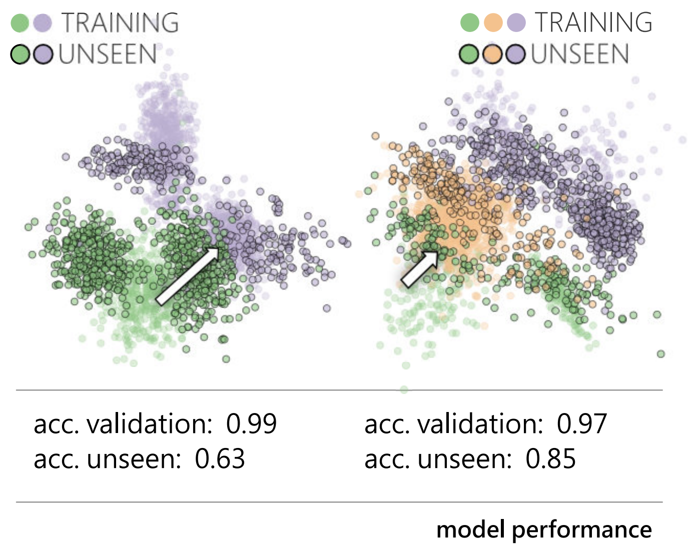


Figure 7.7: Prior probability shift. In this experiment, we show two examples of shift only in the proportion of classes from training to unseen data. When this is the case, even if there is no shift in the features (input variables), the performance often also drops. In both cases, we see that the class in green moves on top of the others (**OC** pattern), one case with a high drop in performance and the other with a moderate, proportional to the smaller overlap. In the left, we have a binary classification problem. In training and validate, the class proportion is 0.3 and 0.7. Then, we invert, and in unseen the proportion is 0.7 and 0.3. In the right, there are three classes and, analogously, we start with 0.2, 0.2 and 0.6, and in the unseen, we have the proportions of 0.2, 0.6 and 0.2.

of existing classes, crossing the decision boundaries. The overall results from this experiment show that the *contiguous* (**C**) pattern (Figure 7.6.1) does not impact the model performance with unseen data, in comparison with the scores obtained with the validation set. We also have, on the other extreme, significant drops associated with the *over-other-class*(es) (**OC**) pattern (e.g., Figures 7.6.2 to 7.6.4). Without data visualization, it would not have been possible to discriminate the no-impact and the high-impact classifier performance scenarios solely based on numerical scores of data distribution changes.

7.5 Second Experiment: Visually Emphasizing Data-class Movements

This experiment is an evolution of the first one because we implement additional visual components that make data class movement more explicit. To show how visualization is vital in identifying impacts caused by data changes that numerical scores cannot explain, we present in the following a series of implementations with a use case that reinforces the main ideas in this part of the thesis.

We work with linearly separable data and a linear dimensionality reduction method to visualize data. The main benefit of the linear techniques is that they preserve the similarity relationships among data points both locally and globally. Therefore, we can trust the visualizations to analyze how classes expand from their configuration in the training set to unseen data sets.

At the same time, the main limitation of linear projections is that they do not work with all types of data as a tool to reveal data clusters based on similarity relationships. When they fail, poor class separation in classification data results in visualizations with overplotting, compromising the analysis process.

7.5.1 Density-based Visualization of Change

The computation of differences in data densities between **T** and **U** is one of the core components of our approach. This first component comprises one data preprocessing step to compute those densities for each data instance, which uses a *Ball*

Tree [31] data structure to provide efficient indexing for nearest-neighbor search. With the Ball Tree at hand, we use a radius search strategy to find data point neighbors and corresponding densities. We start with a fixed radius empirically obtained for any dataset after data standardization. We also make use of an explicit encoding visual comparison design to communicate the density differences (in [36], Gleicher describes possible visual comparison layouts).

The intuition behind the visualization of differences is that it is more likely that a model can have trouble in classifying unseen data if something new appears, compared to what this learner saw at training time. The visualization of density-differences emphasizes when new points in **U** appear in regions with no similar instances in the training set. It also emphasizes when there is an increase in data points near the class-boundaries in **T**. As expected, the closer we get to the boundaries, the more difficult it is for a classifier to predict the label of a new data instance correctly. Thus, we are also interested in being able to visually follow an increase in the data density nearby such regions. If a data-shift originates that density-increase, it is likely that the model error will rise because it will have more trouble to separate the classes, in comparison with the classification performance collected at the time of model building.

In the radius search for neighbors, we enable the user to interactively update the radius, which consequently updates the visual encoding of the density values in **U** points. In more detail, we compute, for each data point in **U**, its neighbors in **T**. Next, we remain with all data points in **U**, but then compute their neighbors in **U**. We then build the difference between both sets and store a density difference score for every point. Finally, plot the **U** data points and color-encode the difference using a linear scale (see Figure 7.8). We apply dimensionality reduction methods like *Principal Component Analysis* (PCA), and in this case, we plot the two main components for both, **T** and **U** sets. Such pipeline enables us to communicate and visually boost the differences, if an increasing number of points is dissimilar to the training data, or if the areas next to the class-boundaries show a similar growth. Besides visually encoding the difference of densities in **U**, we use an interactive filter to show only the densest regions also in **U**. This strategy minimizes overplotting, and let us focus on what changes are relevant to analyze the dataset shift problem.

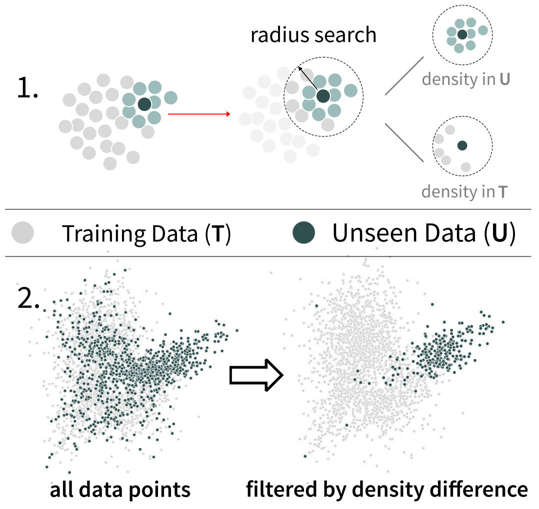


Figure 7.8: Density-difference visualization. (1) We use a *Ball Tree* data structure and perform a nearest-neighbor radius search, to identify for each point in unseen data (**U**) how many neighbors there are in both training (**T**) and **U** sets. (2) Then, we can calculate the density difference to filter and visualize only the regions in **U** in which there is a density increase, in comparison with **T**.

7.5.2 Tracking the Movement of Centroids

Besides the representation of the **T** and **U** sets in the scatter plots, we add two additional layers of information to the main visualization panel of *Dataset-shift Contours*: the **T** class boundaries, and the movement of data centroids caused by the shift. We use a Gaussian kernel with an adjustable bandwidth value (standard deviation) to plot the training data class-boundaries, or simply, *contours*. The analysis of the similarity between **T** and **U** can, for example, stem from the visualization of overlaps in the 2D projections. The more significant the overlap is, the less risk of **U** misclassification exists. However, when we add the class information layer, we can track how the **U** data instances move on top of, or far away from the class contours in **T**. These contours represent the class boundaries of model training data. New data patterns can emerge in **U** and get closer to the boundaries, which can have an impact on the classification performance.

To emphasize the data-shift at class level, we visually track the movement of the class centroid from **T** to **U**. Computing the centroids in **T** is straightforward, because of the existing class labels. We use the standard euclidean distance metric.

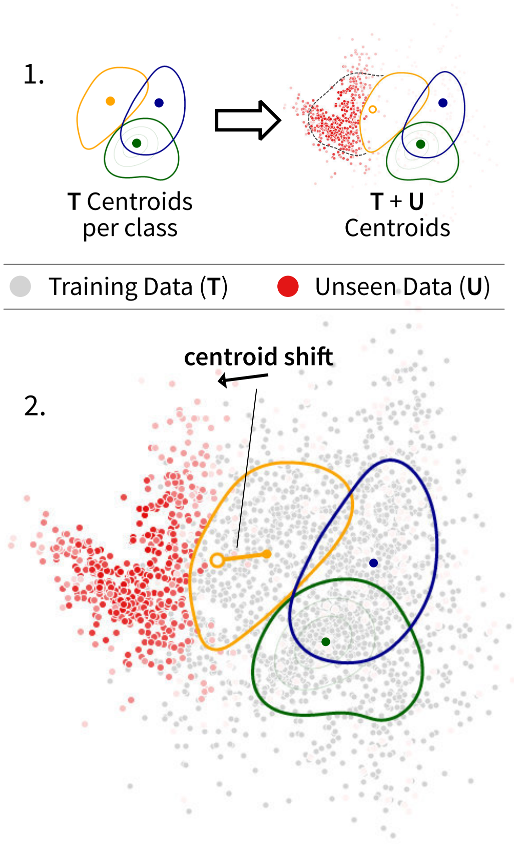


Figure 7.9: Movement of centroids. (1) We start by computing the class-centroids in training data (**T**). Then, we calculate the closest centroid for each new instance in unlabeled data (**U**) and recompute the centroids of the expanded **T** + **U** dataset. (2) Finally, we can plot the path of the class centroids-movement, from **T** to **U**, and represent it on top of linear data projections.

However, our approach aims at unlabeled classification data. Hence, there are no labels yet where the centroids of **U** can be computed. To do so, we make use of a Nearest Centroid classifier and calculate the closest centroid in **T** for each data point in **U**. Using this information as the **U** labels, we take **T** + **U** as a single dataset and update the centroids in this expanded space, which considers the training plus the unseen data (see Figure 7.9) The strategy of considering **T** + **U** as a single set to compute the new centroids brings one advantage: independent of the number of data points in **U**, we can always compute **T** + **U** centroids. **T** is already representative because it is all the data to train the model. Therefore, even if the **U** set comes as small increments, we assure that there are enough points in $\mathbf{T} + \mathbf{U}$ for centroid computation. Alternatively, a Gaussian mixture model (GMM) can be employed instead of the nearest centroid approach. In the presence of unknown data distributions, this model type assumes the data points come from a mixture of Gaussian distributions with unknown parameters.

We also provide an interactive filter that sets a threshold for the minimum distance between corresponding centroids in $\mathbf{T}$ and $\mathbf{U}$. We plot or hide each class-centroid movement according to the threshold value. This way, we can remove centroids that appear almost or entirely overplotted in the visualization, in case no centroid-movement for a given data class can be observed. Finally, we can apply a linear projection method like the Principal Component Analysis (PCA). In this scenario, all the distances between each pair of points in the projection are meaningful, and we can interpret these distances as similarity information. This characteristic of linear projection techniques enables us to build the movement-of-centroids layer of information in *Dataset-shift Contours*. The representation of the centroid movement yet holds limitations with non-linearly separable data. Following, we outline our approach with regard to non-linearly separable data and its impact on supervised learning classification that is caused by changes in the data.

7.5.3 Pattern Identification in Synthetic Data

As a proof-of-concept, we start applying our methods in two use-cases using synthetic data before testing it with benchmark datasets. The data preparation procedure for this second experiment is the same as we presented in the first experiment.

Description of the Experiment

In this experiment, we exhibit two cases of dataset shift with opposite impacts on the ability of the model to classify the unseen data correctly. In the first case, the shift causes no drop in performance at all. Conversely, in the second case, there is a significant drop when we compare the performance measured at the time of training with the one with unseen data. The motivation for this scenario is to show a clear example of similar dataset shift statistics, but completely different impacts on supervised learning, which our approach can foresee based on the patterns we

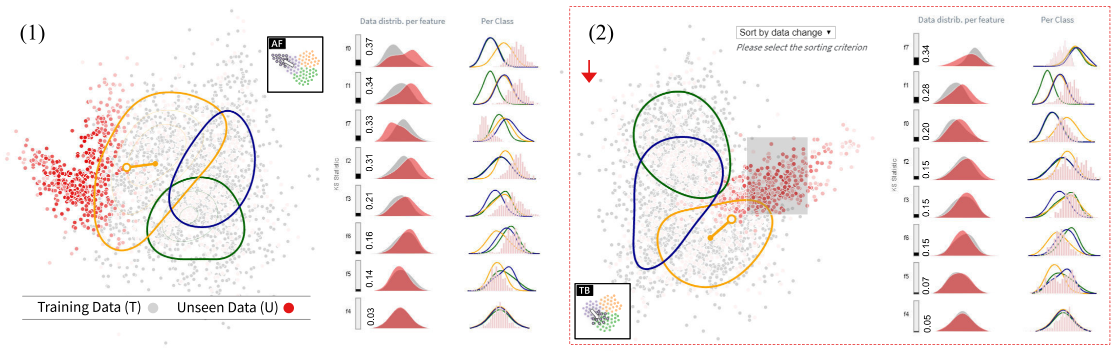


Figure 7.10: Linearly separable data. In the above examples (1) and (2), we show two cases in which a numerically measured significant dataset shift between training and unseen data subsets exists (**5 features equal or greater than a 0.2 KS statistic in** (1) **and three features in** (2)). However, in (1), there is no impact in model performance to classify unseen data. Conversely, in (2), the performance drop is significant. The *Kolmogorov–Smirnov* (KS) test indicates significant changes in data feature distributions, comparing **T** and **U**. We sort the density plots with the distributions per feature by those scores, which reveal details on data changes at the feature level. Visual inspection is the way to understand the differences that numerical scores cannot convey, namely to see how direction data-classes move.

identify and describe in the previous section. We use the *Kolmogorov–Smirnov* (KS) test [38] to measure how the data distribution of each feature changes from training to unseen sets. Then, for each data feature, we run the KS test between the corresponding distributions in both **T** and **U** sets.

The data preparation specs are almost equal between both cases, except for the way we manipulate the proportion of the classes in the unseen sets, detailed in the following. We use again the *scikit-learn* classification data generator and produce data with three classes and eight features. We introduce a covariate-shift between **T** and **U**, following the procedures we include in the data preparation description. We also manipulate the proportion of the classes in the unseen set, favoring one different data class in each of both experiments. By doing that, besides having the covariate-shift, we also introduce the prior probability-shift in new data when we manipulate the proportion of the target variable (*y-shift*). This last manipulation of class proportions produce the effects of opposite impacts on model performance between the two experiments, caused by the different directions that the unseen data expand to regions in the data space with no similar training instances. We train the models with 2000 instances and measure the performance at the time of training using a 1000 instances validation set. The unseen subset has also 1000 data instances. We train a MultiLayer Perceptron Model (MLP) in both cases, which delivers a high accuracy score of 0.97 at training time. Without retraining the model, we achieve almost the same performance (0.96) when we classify unseen data in the Figure 7.10.1 example. However, in Figure 7.10.2, there is a 0.38 drop in performance.

In our prototype, the interactive components are four sliders which enable:

1. To adjust the size of the radius that we use to compute the density of each data instance based on its neighborhood in training data.
2. To filter the unseen set based on data density.
3. To set the bandwidth (the standard deviation) of the Gaussian kernel that we use to plot the data class contours in training data.
4. To set a threshold (minimum dist), and control which centroid distances we want to show.

We also have three auxiliary visualizations:

1. Histograms to show the frequency of counts of categorical data features in training and unseen sets (only for categorical data).
2. Density plots to compare the distributions per feature between **T** and **U** (only for numerical features).
3. Density plots to show the distributions per data class in **T**, as well as superimposed histograms to show the distributions per feature of the selected instances in **U** (only for numerical features).

In conclusion, we show a concrete example of two cases in which, despite having significant changes measured by the KS test (see the KS statistics per feature in Figure 7.10), the direction the new data points expand in **U** explains and help to foresee the opposite impacts in model performance. The centroid-movement layer emphasizes these directions.

7.6 Limitations of the Experiments

The experiments serve to enrich the discussion around the visual identification of data changes. However, their implementations do not fully solve the problem and present limitations. The main one is that we cannot guarantee that it still works with non-linearly separable data. Under those circumstances, it may happen that the linear projection method I use in the visualization results in data overplotting. In this case, it is not possible to follow the movement of data classes using the centroid shift as a proxy for that.

Another limitation is that specific data-distribution changes may keep the training and unseen data centroids in almost the same position even with linearly separable data. With that respect, future work should systematize what types of data drift can be observed from the proposed visual representations and what kinds of changes are not observable.

The same limitation can also result in almost no shape difference in the contours, from training to unseen data sets. Analogously, future work should systematize types of change identifiable with the proposed approach and present if the non-identifiable ones could result in substantial performance drops or only minor impacts.

7.7 Towards a General-purpose Visual-based Monitoring of Data and Model Compatibility in Classification Problems

On the one hand, there are available methods for the automatic detection of data drifts, like the one proposed in [94], in which the data space is subdivided using a grid, and each corresponding cell is compared between two adjacent positions in time. On the other hand, visualizations play a special role whenever the labels from the incoming data packages are not available yet. During this time frame, it is not possible to evaluate the model performance and guarantee if there is a drop with new data. In addition, visualization can also provide additional guidance to model developers and data specialists in assessing the impact of the dataset shift with or without new data labels in classification problems, hand in hand with the automatic identification methods in the same application for full coverage of the problem.

This section proposes a *General-purpose Visual-based Monitoring of Data and*

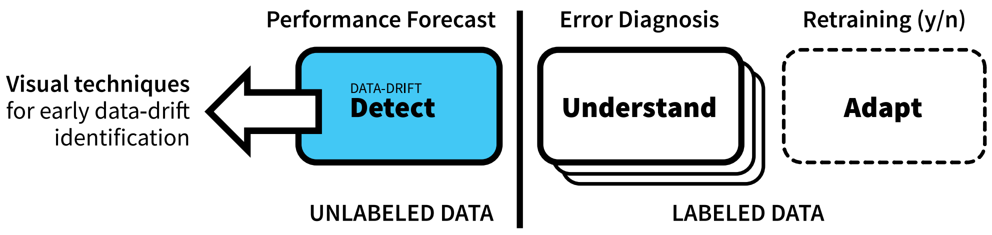


Figure 7.11: After analyzing related works in Section 7.3, I identify a gap in the detection step of dataset-shift visual analysis systems, while the incoming data labels are not available yet. In particular, to the best of my knowledge, it does not exist an approach that can discriminate which data changes result in drops in model performance and which ones do not affect the classification accuracy.

Model Compatibility in Classification Problems. I organize in subtopics the related challenges and corresponding gaps identified in STAR applications and my experiments to address this goal. Still, I do not focus on any specific data domain because the idea is to use abstract visualizations that can represent any data type. In recent work [20], the authors develop novel methods to identify out-of-distribution samples, but they focus on image datasets limiting generalization.

Model Characteristics Besides the broader applicability of being agnostic to model types when detecting dataset shifts, it is important in application contexts to consider that different learners build distinct decision boundaries in classification problems. Existing methods [72] to visualize those boundaries could incorporate model diversity and are a relevant extension to my work. A specific region in the data space may be complex for one model to separate into distinct classes and easy for another model to accomplish the same task. Then, when incoming data move in the direction of this particular region, impacts are most likely also different in each case.

If the goal is to decide whether a model needs retraining, the attribution of model behavior needs consideration. For instance, an overfitted model (e.g., hard-margin linear SVM) may need retraining under data shift. In contrast, a robust model (e.g., Soft-margin linear SVM), which may initially perform worse than the overfitted one, might not need any retraining on a specific type of unseen data under the same shift.

Another model-specific information that could be considered is the degree of uncertainty natively generated as one of the outputs in the classification process (by models like Decision Trees, as previously mentioned). Together with the model decision boundaries, this additional layer of information helps to foresee misclassification risks before the new data labels are available. Data points near the boundary with a high degree of classification uncertainty informed by the model would be the most difficult ones to deal with, and visualization can encode that information.

Still, on the specificities of model types, evaluation is necessary to reveal if analyzing the distribution changes of the features in the raw input data is a good proxy for models that perform the classification in the latent space, for example, kernel SVM or CNN based classifiers.

Data Projections In machine learning applications like image recognition, the number of input variables necessary to build a good learner quickly reaches hundreds. With so many variables, it becomes almost impossible for an analyst to track data changes in individual visualizations. Ranking strategies may also serve as an alternative to display the most relevant variables. However, in addition to ranks, dimensionality reduction is a natural technique to apply in the presence of high-dimensional data.

Throughout the experiments in this chapter, I instantiated the utility of data visualizations as a tool that can foresee class-movement patterns in a manner that numeric scores in isolation cannot accomplish. In particular, the projection of all data variables using linear dimensionality reduction methods played a central role in revealing global patterns of data change. However, as mentioned, the linear methods may not work with non-linearly separable data. Among the non-linear dimensionality reduction methods, *t-SNE* excels at providing separation among data classes for various data types. Although, it does not provide comparable projections, as each run generates different outputs due to its stochastic nature.

An extension of my work is experimenting with *Dynamic t-SNE* [81] or *Joint t-SNE* [106]. Both aim to generate comparable projections of multiple high-dimensional datasets when one needs to represent, for instance, successive events over time, based on adaptations on the original t-SNE implementation [68]. Instead of using the data input features, an alternative strategy could incorporate the differences between two nearest neighbor graphs into the projection and explore the changes over time. Evaluation is necessary to check if the stabilization methods of dynamic and joint t-SNE introduced by its constraints would be enough to compare global data-shift patterns coming from distinct projections.

Model and Data Life-cycles Depending on the application problem, the time to obtain incoming data labels may be too high. Therefore, a general-purpose monitoring system of data drifts could keep an eye directed to the future, visually foreseeing the impact of data distribution changes in new data without the labels, while at the same time looking back to the model errors when the labels are available and using them to provide explanations. By explanations, I mean, for instance, understanding if model errors occur on data concentrated at decision boundaries with no data drift or if the data changes are the main reason for sudden performance drops.

Another target application regarding the data life-cycle is monitoring the data class evolution, i.e., how class definitions remain valid over time. As a consequence of persistent data changes, it may be the case that a class generates, by splitting it, two subclasses, for instance. In the text domain, class-splitting is a typical operation in topic modeling when one must classify text collections using predefined topics. Despite the correlation of this kind of monitoring with my visual-based monitoring proposal, it is out of the scope of my work the merge and split operations over existing classes. For consideration, they should appear as a new extension to this work.

Part III

Conclusion

8 Future Research Directions

Contents

8.1 Limitations

Dimensionality Reduction Data projection using dimensionality reduction techniques has extensive use in data visualization research and applications due to its ability to preserve most of the relevant structure in data, such as outliers, clusters, or underlying manifolds [88]. As mentioned in Section 1.3, I use those techniques throughout Parts I and II of this thesis. In that section, I also mentioned the simplicity of the representation and the widespread adoption of the scatterplot visualization technique, the most common form to represent dimension-reduced data [93].

Especially in Part II, in model generalization, there is the need to visualize all data features together to inspect data-class movements when data changes over time. Under this scenario and depending on the data domain, the number of features in data can scale to an amount that justifies the need for data projection. However, dimensionality reduction methods have their trade-offs. Inevitably, representing a higher number of features in two dimensions involves information loss. The expectation is to preserve most of the relevant data structure, but this goal is sometimes not achievable due to artifacts generated by those methods and data characteristics [88, 92]. An alternative is visualizing each feature directly without preprocessing the data using projections to two-dimensional space. I implement visualization techniques for direct feature representation in Chapter 5, in the *DataShiftExplorer*. It works until a given scale and for identifying certain dataset-shift types regarding its application.

Scalability I work with classification throughout the thesis. The first issue regarding the scalability of a visualization approach to the diversity of possible classification problems is the number of classes. Concerning that scalability dimension, one relevant fact is that most real-world classification data have up to 20 (twenty) classes (see the survey on [82], which states that 0.85 of the selected problems stay under this limit). For this magnitude, there are visualization approaches that can easily accommodate it, like pixel-based, small multiples, or parallel coordinates techniques. The latter is the basis of my customized implementation presented in Chapter 6.

Regarding the scalability of visualizations to the number of data slices collected over time, like in streaming data contexts, I propose in the next section, as future work, the intersection of dataset-shift visual analysis research with all the existing efforts to visualize trajectories in general, usually with geo-referenced data, but in this context to reveal model-behavior trajectories over time.

The last aspect related to the scalability of my proposed solutions is the number of data points (dataset size). This dimension of the problem talks with limits for visualization and the performance of the interactive approaches under heavy computation. I worked with both synthetic and benchmark datasets of small and moderate sizes. I did not work with big data, so performance issues were not the focus of my work, which is a limitation. However, after iterating through a series of experiments, I consider the data type a more significant challenge than the data size. When dealing with mixed data (numerical and non-numerical attributes), the alternatives to preprocessing (e.g., similarity computations) become limited, and this is a common challenge for information visualization research in general. On the visualization side, well-spread approaches like using heatmaps to deal with overplotting in scatterplots, for instance, which I use, deal with dataset size issues.

Evaluation Finally, in the scale of dataset sizes I use, the performance of the applications and the scalability of the visual solutions I apply worked consistently, as expected. However, a comprehensive evaluation considering additional scenarios to stress the proposed solutions concerning application performance issues was out of my scope. Additional user studies considering usability and interpretability issues, besides quantitative evaluation, would be welcome in such an evaluation.

8.2 Future Work

In my research, the categorization of model-data relationships in supervised learning is multi-folded. On the one hand, it directly inspires the design of interactive applications to support supervised learning (Chapters 3, 4, 6, and 7). On the other hand, it helps to reveal research gaps and discuss future directions (Chapter 7 and this Chapter).

M:N relationship extensions In Chapters 3 and 4, I work with the M:1 relationship, and in Chapters 6 and 7 with the 1:N model-data relationship. The M:N relationship, which corresponds to multiple models and data subsets in both model and data spaces, respectively, is not explored yet by this thesis (Figure 8.1). A direct application of developing the M:N case is joining the interactive ensemble model building with the analysis of model generalization and stability across multiple (N) incoming data subsets. The visual analysis of numerous incoming data spaces supports the identification of the dataset shift problem, as I do in Chapters 6 and 7, but in these cases with only one model in the model space.

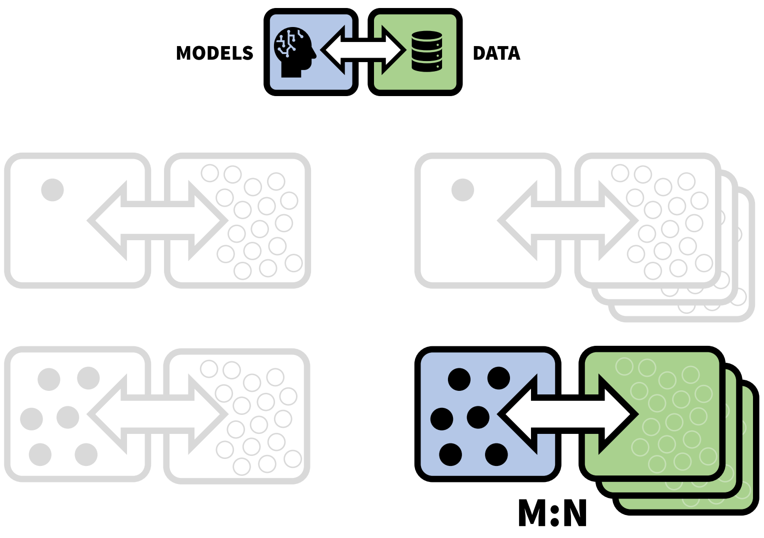


Figure 8.1: Research opportunities. Among the four possible model-data relationships proposed in this thesis, the M:N case is the natural extension for future work. A combination of interactive ensemble model-building capabilities (Chapter 4) with dataset shift identification (Chapters 6 and 7) is an example of extension to the current work. Besides identifying abrupt data changes, this extension could bring the possibility to analyze ensemble model generalization to new data in general, even in the presence of subtle data changes, in addition to the numerical methods to infer generalization presented in Chapter 4.

Data probing Regarding the mentioned limitations associated with dimensionality reduction techniques (Section 8.1), there are alternatives to mitigate them in dataset-shift visual analysis pipelines. Data probing [98] is one of the approaches. Considering that data projections to the 2D space do not always retain high-dimensional data similarity information, future work can advance into implementing an initial step that the data specialist could check, with the data available at hand, if visual representations convey data similarity as expected. In such an application, automatic preprocessing could generate subsets of training and unseen datasets in distinct configurations to simulate the arrival of incoming data. Then, the analyst could inspect the unseen data visual patterns and check if they correspond with, e.g., the no impact or significant impact on model performance. An alternative approach to turning on or off projections in a linked view application context is projection quality metrics, like stress. Based on this type of measure and predefined thresholds, a specialist could decide whether to trust visual representations coming from dimensionality reduction.

Model-performance trajectories Another extension to my work is exploring different visual comparison layouts other than the superposition of data subsets. An alternative visual comparative approach could support, for instance, tracking the evolution of model performance across a series of distinct and successive data subspaces. A significant adaptation to my work could allow supporting incremental learning, in which data arrive in a streaming fashion, in small but continuous portions over time. In the model space, one could design a visual representation of a model path showing the evolution of the model performance by using any metric of interest. In the data space, one could explore the possibility of retaining the last states of it, but at the same time offering the chance to use the model path in the model space to interactively go back and forth across different time stamps of the model life-cycle.

Tracking class evolution Besides tracking how the model performance evolves, another future research direction is adapting the data space representation to monitor how class boundaries change in classification problems. When data changes, the model may need retraining. However, there is also the case in which class definitions become deprecated, like topic merge or topic split in the text domain, in topic modeling. Under this scenario, visualization can support the detailed analysis and monitoring of these class boundary movements. When changes are enormous, a domain specialist with the proper tools can decide whether it is time to merge or split an existing class, for instance, and restart from the beginning of the model building process.

Related reseach topics Leaving behind my M:N relationships and looking to information visualization and visual analytics research related to machine learning from an overall perspective, there is a trend towards *explainable AI* (artificial intelligence) [63]. As ML becomes ubiquitous and models trained to make automated decisions influence citizens'daily lives, there is a need for transparency forced by government regulation. Research on visual analytics has a long tradition of supporting decision-tree model analysis due to the relative simplicity of these models (like the authors did in the BaobabView [104] system). Transversely to distinct supervised and non-supervised learning techniques, Endert et al. presented a STAR report [33] on the integration of machine learning and visual analytics. The authors highlight the capacity to build trust in classification models by integrating interaction and visualization to model outcomes. Explainability and trust go hand-in-hand in this context.

Another opportunity for visualization research that goes hand in hand with the growing adoption of machine learning are platforms developed for non-programmers that support standard users to build ML pipelines visually. Therefore, we can expect a growing need for research in the intersection between visualization and machine-learning fields in the following years. This expansion goes toward promoting the standard user as its target user, not only the skilled specialists. Lastly, the mentioned increase in scope represents a potential growth in the importance of visualization research as an essential tool for bringing together citizens and machine-learning processes.

Lastly, I did not explore the 1:1 model-data relationship case because it offers fewer opportunities to analyze, visually compare and manipulate model and data objects. My focus is the visual integration of model and data spaces, and with fewer objects on both sides, the opportunities become scarce. However, we can use the idea of M distinct models in the model space to represent the same model with different parameter settings, for instance, or even trained with varying sets of features. In this situation, the model type is the same, but we could then replicate it to represent the series of possible model building setups. In the mentioned case of the same model with different parameter settings, the visualization of the model path performance would be a direct application. Patterns of how models evolve could offer essential insights and guide model builders to categorize dynamic behavior types.

Bibliography

[1] Leandro Maciel Almeida and Pedro Sereno Galvão. "Ensembles with Clustering-and-Selection Model Using Evolutionary Algorithms." In: *Intelligent Systems (BRACIS), 2016 5th Brazilian Conference on*. IEEE. 2016, pp. 444–449.

[2] Bilal Alsallakh, Allan Hanbury, Helwig Hauser, Silvia Miksch, and Andreas Rauber. "Visual methods for analyzing probabilistic classification data." In: *IEEE transactions on visualization and computer graphics* 20.12 (2014), pp. 1703–1712.

[3] Saleema Amershi, Maya Cakmak, William Bradley Knox, and Todd Kulesza. "Power to the people: The role of humans in interactive machine learning." In: *Ai Magazine* 35.4 (2014), pp. 105–120.

[4] Saleema Amershi, Max Chickering, Steven M Drucker, Bongshin Lee, Patrice Simard, and Jina Suh. "Modeltracker: Redesigning performance analysis tools for machine learning." In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM. 2015, pp. 337–346.

[5] F. J. Anscombe. "Graphs in Statistical Analysis." In: *The American Statistician* 27.1 (1973), pp. 17–21. DOI: 10.1080/00031305.1973.10478966.

[6] Woodrow Barfield and Ugo Pagallo. *Research handbook on the law of artificial intelligence*. Edward Elgar Publishing, 2018.

[7] Yoav Benjamini. "Opening the box of a boxplot." In: *The American Statistician* 42.4 (1988), pp. 257–262.

[8] Enrico Bertini and Denis Lalanne. "Surveying the complementary role of automatic data analysis and visualization in knowledge discovery." In: *Proceedings of the ACM SIGKDD Workshop on Visual Analytics and Knowledge Discovery: Integrating Automated Analysis with Interactive Exploration*. ACM. 2009, pp. 12–20.

[9] Steffen Bickel, Michael Brückner, and Tobias Scheffer. “Discriminative Learning under Covariate Shift with a Single Optimization Problem.” In: *Dataset Shift in Machine Learning*. MIT, 2009, pp. 161–177.

[10] Ekaba Bisong. “Google AutoML: cloud vision.” In: *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Springer, 2019, pp. 581–598.

[11] Leo Breiman. “Bagging predictors.” In: *Machine learning* 24.2 (1996), pp. 123–140.

[12] Leo Breiman. “Random Forests.” In: *Machine Learning* 45.1 (2001), pp. 5–32.

[13] Michael Brooks, Saleema Amershi, Bongshin Lee, Steven M Drucker, Ashish Kapoor, and Patrice Simard. “FeatureInsight: Visual support for error-driven feature ideation in text classification.” In: *Visual Analytics Science and Technology (VAST), 2015 IEEE Conference on*. IEEE. 2015, pp. 105–112.

[14] Gavin Brown and Ludmila I Kuncheva. ““Good” and “bad” diversity in majority vote ensembles.” In: *International Workshop on Multiple Classifier Systems*. Springer. 2010, pp. 124–133.

[15] Doina Caragea, Dianne Cook, Hadley Wickham, and Vasant Honavar. “Visual methods for examining SVM classifiers.” In: *Visual Data Mining*. Springer, 2008, pp. 136–153.

[16] Rich Caruana, Art Munson, and Alexandru Niculescu-Mizil. “Getting the most out of ensemble selection.” In: *Data Mining, 2006. ICDM’06. Sixth International Conference on*. IEEE. 2006, pp. 828–833.

[17] Rich Caruana, Alexandru Niculescu-Mizil, Geoff Crew, and Alex Ksikes. “Ensemble selection from libraries of models.” In: *Proceedings of the twenty-first international conference on Machine learning*. ACM. 2004, p. 18.

[18] Alex Cattle. *Getting started with AI*. `https://ubuntu.com/blog/getting-started-with-ai`. (Accessed online Nov., 2021). 2019.

[19] John M Chambers. *Graphical Methods for Data Analysis*. Chapman and Hall/CRC, 2017.

[20] Changjian Chen, Jun Yuan, Yafeng Lu, Yang Liu, Hang Su, Songtao Yuan, and Shixia Liu. “Oodanalyzer: Interactive analysis of out-of-distribution samples.” In: *IEEE transactions on visualization and computer graphics* 27.7 (2020), pp. 3335–3349.

[21] Joey Cherdarchuk. *Visualizing distributions.* `http://www.darkhorseanalytics.com/blog/visualizing-distributions-3`. (Accessed online Nov., 2021). 2016.

[22] David A Cieslak and Nitesh V Chawla. “A framework for monitoring classifiers’ performance: when and why failure occurs?” In: *Knowledge and Information Systems* 18.1 (2009), pp. 83–108.

[23] Michael Correll and Michael Gleicher. “Error bars considered harmful: Exploring alternate encodings for mean and error.” In: *IEEE transactions on visualization and computer graphics* 20.12 (2014), pp. 2142–2151.

[24] Michael Correll, Mingwei Li, Gordon Kindlmann, and Carlos Scheidegger. “Looks Good To Me: Visualizations As Sanity Checks.” In: *IEEE transactions on visualization and computer graphics* 25.1 (2018), pp. 830–839.

[25] Thomas M Cover and Joy A Thomas. *Elements of information theory.* John Wiley & Sons, 2012.

[26] T.F. Cox and A.A. Cox. *Multidimensional Scaling, Second Edition.* Chapman & Hall/CRC Monographs on Statistics & Applied Probability. CRC Press, 2000. ISBN: 9781420036121.

[27] Chicago Police Department. *Crimes - 2001 to present.* URL: `http://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2`.

[28] Thomas G Dietterich. “Ensemble methods in machine learning.” In: *International workshop on multiple classifier systems.* Springer. 2000, pp. 1–15.

[29] Banu Diri and Songul Albayrak. “Visualization and analysis of classifiers performance in multi-class medical data.” In: *Expert Systems with Applications* 34.1 (2008), pp. 628–634.

[30] Gregory Ditzler, Manuel Roveri, Cesare Alippi, and Robi Polikar. “Learning in nonstationary environments: A survey.” In: *IEEE Computational Intelligence Magazine* 10.4 (2015), pp. 12–25.

[31] Mohamad Dolatshah, Ali Hadian, and Behrouz Minaei-Bidgoli. “Ball*-tree: Efficient spatial indexing for constrained nearest-neighbor search in metric spaces.” In: *arXiv preprint arXiv:1511.00628* (2015).

[32] Pedro Domingos. “A few useful things to know about machine learning.” In: *Communications of the ACM* 55.10 (2012), pp. 78–87.

[33] Alex Endert, William Ribarsky, Cagatay Turkay, BL William Wong, Ian Nabney, I Díaz Blanco, and Fabrice Rossi. “The state of the art in integrating machine learning into visual analytics.” In: *Computer Graphics Forum.* Vol. 36. 8. Wiley Online Library. 2017, pp. 458–486.

[34] Yoav Freund and Robert E. Schapire. "Experiments with a new boosting algorithm." In: *Thirteenth International Conference on Machine Learning*. San Francisco: Morgan Kaufmann, 1996, pp. 148–156.

[35] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. "A survey on concept drift adaptation." In: *ACM computing surveys (CSUR)* 46.4 (2014), pp. 1–37.

[36] Michael Gleicher. "Considerations for Visualizing Comparison." In: *IEEE transactions on visualization and computer graphics* 24.1 (2018), pp. 413–423.

[37] Algorithmics Group. *MDSJ: Java Library for Multidimensional Scaling (Version 0.2)*. `http://www.inf.uni-konstanz.de/algo/software/mdsj/`. (Accessed online Dec., 2015). 2009.

[38] Michiel Hazewinkel. "Kolmogorov-smirnov test." In: *Encyclopedia of Mathematics, Springer* (2001).

[39] Francisco Herrera. *Dataset Shift in Classification: Approaches and Problems*. `http://iwann.ugr.es/2011/pdf/InvitedTalk-FHerrera-IWANN11.pdf`. (Accessed online Nov., 2021). 2011.

[40] John Jay Hilfiger. *Graphing Data with R: An Introduction*. O'Reilly Media, Inc, 2015.

[41] Jerry L Hintze and Ray D Nelson. "Violin plots: a box plot-density trace synergism." In: *The American Statistician* 52.2 (1998), pp. 181–184.

[42] Mia Hubert and Ellen Vandervieren. "An adjusted boxplot for skewed distributions." In: *Computational statistics & data analysis* 52.12 (2008), pp. 5186–5201.

[43] Heinrich Jiang, Been Kim, and Maya Gupta. "To Trust Or Not To Trust A Classifier." In: *arXiv preprint arXiv:1805.11783* (2018).

[44] I.T. Jolliffe. *Principal component analysis*. Springer series in statistics. Springer-Verlang, 1986. ISBN: 9780387962696.

[45] Robert Jung. *ensembleLibrary version 1.0.6*. `http://weka.sourceforge.net/doc.packages/ensembleLibrary`. (Accessed online Nov., 2021). 2017.

[46] Anna Jurek, Yaxin Bi, Shengli Wu, and Chris Nugent. "A survey of commonly used ensemble-based classification techniques." In: *The Knowledge Engineering Review* 29.5 (2014), pp. 551–581.

[47] Kaggle. *San Francisco Crime Classification*. 2015. URL: `https://www.kaggle.com/c/sf-crime`.

[48] Peter Kampstra et al. “Beanplot: A boxplot alternative for visual comparison of distributions.” In: *Journal of Statistical Software* (2008).

[49] Ashish Kapoor, Bongshin Lee, Desney Tan, and Eric Horvitz. “Interactive optimization for steering machine classification.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. 2010, pp. 1343–1352.

[50] Alboukadel Kassambara. *ggpubr: ggplot2 Based Publication Ready Plots*. `http://rpkgs.datanovia.com/ggpubr/`. (Accessed online Nov., 2021). 2019.

[51] Daniel Keim, Gennady Andrienko, Jean-Daniel Fekete, Carsten Görg, Jörn Kohlhammer, and Guy Melançon. “Visual analytics: Definition, process, and challenges.” In: *Information visualization*. Springer, 2008, pp. 154–175.

[52] Wolfgang Kienreich and Christin Seifert. “Visual exploration of feature-class matrices for classification problems.” In: *International Workshop on Visual Analytics (EuroVA)*. 2012, pp. 37–41.

[53] Robert Kosara, Fabian Bendix, and Helwig Hauser. “Parallel sets: Interactive exploration and visual analysis of categorical data.” In: *IEEE transactions on visualization and computer graphics* 12.4 (2006), pp. 558–568.

[54] Josua Krause, Adam Perer, and Enrico Bertini. “INFUSE: interactive feature selection for predictive modeling of high dimensional data.” In: *IEEE transactions on visualization and computer graphics* 20.12 (2014), pp. 1614–1623.

[55] Meelis Kull and Peter Flach. “Patterns of dataset shift.” In: *First International Workshop on Learning over Multiple Contexts (LMCE) at ECML-PKDD*. 2014.

[56] S. Kullback and R. A. Leibler. “On Information and Sufficiency.” In: *Ann. Math. Statist.* 22.1 (Mar. 1951), pp. 79–86. DOI: `10.1214/aoms/1177729694`.

[57] Ludmila I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.

[58] Ludmila I Kuncheva and Christopher J Whitaker. “Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy.” In: *Machine learning* 51.2 (2003), pp. 181–207.

[59] Ludmila I Kuncheva, Christopher J Whitaker, Catherine A Shipp, and Robert PW Duin. “Limits on the majority vote accuracy in classifier fusion.” In: *Pattern Analysis & Applications* 6.1 (2003), pp. 22–31.

[60] Alexander Lex, Marc Streit, Christian Partl, Karl Kashofer, and Dieter Schmalstieg. "Comparative analysis of multidimensional, quantitative data." In: *IEEE Transactions on Visualization and Computer Graphics* 16.6 (2010), pp. 1027–1035.

[61] Yuan-Hong Liao, Amlan Kar, and Sanja Fidler. "Towards Good Practices for Efficiently Annotating Large-Scale Image Classification Datasets." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 4350–4359.

[62] M. Lichman. *UCI Machine Learning Repository*. `http://archive.ics.uci.edu/ml`. (Accessed online Nov., 2021). 2013.

[63] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. "Explainable ai: A review of machine learning interpretability methods." In: *Entropy* 23.1 (2021), p. 18.

[64] Shixia Liu, Xiting Wang, Mengchen Liu, and Jun Zhu. "Towards better analysis of machine learning models: A visual analytics perspective." In: *Visual Informatics* 1.1 (2017), pp. 48–56.

[65] Shixia Liu, Xiting Wang, Mengchen Liu, and Jun Zhu. "Towards better analysis of machine learning models: A visual analytics perspective." In: *Visual Informatics* 1.1 (2017), pp. 48–56.

[66] Zhuan Liu, Qun Dai, and Ningzhong Liu. "Ensemble selection by GRASP." In: *Applied intelligence* 41.1 (2014), pp. 128–144.

[67] Yafeng Lu, Rolando Garcia, Brett Hansen, Michael Gleicher, and Ross Maciejewski. "The State-of-the-Art in Predictive Visual Analytics." In: *Computer Graphics Forum*. Vol. 36. Wiley Online Library. 2017, pp. 539–562.

[68] Laurens van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE." In: *Journal of Machine Learning Research* 9.Nov (2008), pp. 2579–2605.

[69] Justin Matejka and George Fitzmaurice. "Same stats, different graphs: Generating datasets with varied appearance and identical statistics through simulated annealing." In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM. 2017, pp. 1290–1294.

[70] Thorsten May, Andreas Bannach, James Davey, Tobias Ruppert, and Jörn Kohlhammer. "Guiding feature subset selection with an interactive visualization." In: *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*. IEEE. 2011, pp. 111–120.

[71] Robert McGill, John W Tukey, and Wayne A Larsen. "Variations of box plots." In: *The American Statistician* 32.1 (1978), pp. 12–16.

[72] MA Migut, M Worring, and CJ Veenman. “Visualizing multi-dimensional decision boundaries in 2D.” In: *Data Mining and Knowledge Discovery* 29.1 (2015), pp. 273–295.

[73] Jose G Moreno-Torres, Troy Raeder, RocíO Alaiz-RodríGuez, Nitesh V Chawla, and Francisco Herrera. “A unifying view on dataset shift in classification.” In: *Pattern Recognition* 45.1 (2012), pp. 521–530.

[74] Christopher Olah. *Visualizing Representations: Deep Learning and Human Beings.* `http://colah.github.io/posts/2015-01-Visualizing-Representations/`. (Accessed online Nov., 2021). 2015.

[75] Luciana Padua, Hendrik Schulze, Krešimir Matković, and Claudio Delrieux. “Interactive exploration of parameter space in data mining: Comprehending the predictive quality of large decision tree collections.” In: *Computers & Graphics* 41 (2014), pp. 99–113.

[76] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python.” In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[77] Moacir P Ponti Jr. “Combining classifiers: from the creation of ensembles to the decision fusion.” In: *Graphics, Patterns and Images Tutorials (SIBGRAPI-T), 2011 24th SIBGRAPI Conference on.* IEEE. 2011, pp. 1–10.

[78] Viswanath Poosala, Peter J. Haas, Yannis E. Ioannidis, and Eugene J. Shekita. “Improved Histograms for Selectivity Estimation of Range Predicates.” In: *SIGMOD Rec.* 25.2 (June 1996), pp. 294–305. ISSN: 0163-5808. DOI: `10.1145/235968.233342`. URL: `http://doi.acm.org/10.1145/235968.233342`.

[79] Reid Porter, James Theiler, and Don Hush. “Interactive machine learning in data exploitation.” In: *Computing in Science & Engineering* 15.5 (2013), pp. 12–20.

[80] Peter Prettenhofer. *Dataset Shift in Machine Learning.* `https://github.com/pprett/dataset-shift-osdc16`. (Accessed online Nov., 2021). 2016.

[81] Paulo E. Rauber, Alexandre X. Falcão, and Alexandru C. Telea. “Visualizing Time-Dependent Data Using Dynamic t-SNE.” In: *Proceedings of the Eurographics / IEEE VGTC Conference on Visualization: Short Papers.* EuroVis ’16. Groningen, The Netherlands: Eurographics Association, 2016, pp. 73–77.

[82] Donghao Ren, Saleema Amershi, Bongshin Lee, Jina Suh, and Jason D Williams. “Squares: Supporting interactive performance analysis for multi-class classifiers.” In: *IEEE transactions on visualization and computer graphics* 23.1 (2017), pp. 61–70.

[83] Penny Rheingans and Marie Desjardins. “Visualizing high-dimensional predictive model quality.” In: *Proceedings of the conference on Visualization’00.* IEEE Computer Society Press. 2000, pp. 493–496.

[84] Bastian Rieck and Heike Leitte. “Enhancing Comparative Model Analysis using Persistent Homology.” In: *Visualization for Predictive Analytics Workshop, Paris, France.* 2014.

[85] Nils Rodrigues and Daniel Weiskopf. “Nonlinear dot plots.” In: *IEEE transactions on visualization and computer graphics* 24.1 (2017), pp. 616–625.

[86] Fabio Roli. “Multiple classifier systems.” In: *Encyclopedia of Biometrics* (2015), pp. 1142–1147.

[87] Dominik Sacha, Michael Sedlmair, Leishi Zhang, John A Lee, Jaakko Peltonen, Daniel Weiskopf, Stephen C North, and Daniel A Keim. “What you see is what you can change: Human-centered machine learning by interactive visualization.” In: *Neurocomputing* 268 (2017), pp. 164–175.

[88] Dominik Sacha, Leishi Zhang, Michael Sedlmair, John A Lee, Jaakko Peltonen, Daniel Weiskopf, Stephen C North, and Daniel A Keim. “Visual interaction with dimensionality reduction: A structured literature analysis.” In: *IEEE transactions on visualization and computer graphics* 23.1 (2016), pp. 241–250.

[89] Claude Sammut and Geoffrey I Webb. *Encyclopedia of machine learning.* Springer Science & Business Media, 2011.

[90] Peter D Sasieni and Patrick Royston. “Dotplots.” In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 45.2 (1996), pp. 219–234.

[91] Bruno Schneider, Dominik Jäckle, Florian Stoffel, Alexandra Diehl, Johannes Fuchs, and Daniel A. Keim. “Visual Integration of Data and Model Space in Ensemble Learning.” In: *Symposium on Visualization in Data Science (VDS) at IEEE VIS 2017.* IEEE. 2017.

[92] M Sedlmair, Matt Brehmer, S Ingram, and T Munzner. “Dimensionality reduction in the wild: Gaps and guidance.” In: *Dept. Comput. Sci., Univ. British Columbia, Vancouver, BC, Canada, Tech. Rep. TR-2012-03* (2012).

[93] Michael Sedlmair, Tamara Munzner, and Melanie Tory. “Empirical guidance on scatterplot and dimension reduction technique choices.” In: *IEEE transactions on visualization and computer graphics* 19.12 (2013), pp. 2634–2643.

[94] Tegjyot Singh Sethi, Mehmed Kantardzic, and Hanquing Hu. “A grid density based framework for classifying streaming data in the presence of concept drift.” In: *Journal of Intelligent Information Systems* 46.1 (2016), pp. 179–211.

[95] Donald Shepard. “A two-dimensional interpolation function for irregularly-spaced data.” In: *Proceedings of the 1968 23rd ACM national conference.* 1968, pp. 517–524.

[96] Catarina Silva and Bernardete Ribeiro. “Visualization of Individual Ensemble Classifier Contributions.” In: *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems.* Springer. 2016, pp. 633–642.

[97] Bernard W Silverman. *Density estimation for statistics and data analysis.* Routledge, 2018.

[98] Julian Stahnke, Marian Dörk, Boris Müller, and Andreas Thom. “Probing projections: Interaction techniques for interpreting arrangements and errors of dimensionality reductions.” In: *IEEE transactions on visualization and computer graphics* 22.1 (2015), pp. 629–638.

[99] Amos J Storkey. “When training and test sets are different: characterizing learning transfer.” In: *In Dataset Shift in Machine Learning.* MIT Press, 2009, pp. 3–28.

[100] Masashi Sugiyama, Neil D Lawrence, Anton Schwaighofer, et al. *Dataset shift in machine learning.* The MIT Press, 2017.

[101] Justin Talbot, Bongshin Lee, Ashish Kapoor, and Desney S Tan. “EnsembleMatrix: interactive visualization to support machine learning with multiple classifiers.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* ACM. 2009, pp. 1283–1292.

[102] Edward R Tufte. *The visual display of quantitative information.* Vol. 2. Graphics press Cheshire, CT, 2001.

[103] JW Tukey. *Exploratory data analysis.* Addison-Wesley Publishing Company, 1977.

[104] Stef Van Den Elzen and Jarke J van Wijk. “Baobabview: Interactive construction and analysis of decision trees.” In: *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on.* IEEE. 2011, pp. 151–160.

[105] Xumeng Wang, Wei Chen, Jiazhi Xia, Zexian Chen, Dongshi Xu, Xiangyang Wu, Mingliang Xu, and Tobias Schreck. “ConceptExplorer: Visual analysis of concept drifts in multi-source time-series data.” In: (2020), pp. 1–11.

[106] Yinqiao Wang, Lu Chen, Jaemin Jo, and Yunhai Wang. “Joint t-SNE for Comparable Projections of Multiple High-Dimensional Datasets.” In: *IEEE Transactions on Visualization and Computer Graphics* 28.1 (2021), pp. 623–632.

[107] Michael Waskom. *Seaborn: statistical data visualization.* `http://seaborn.pydata.org`. (Accessed online Nov., 2021). 2018.

[108] James Wexler. *Facets: An open source visualization tool for machine learning training data.* `http://ai.googleblog.com/2017/07/facets-open-source-visualization-tool.html`. (Accessed online Nov., 2021). 2017.

[109] Hadley Wickham and Lisa Stryjewski. “40 years of boxplots.” In: *Am. Statistician* (2011).

[110] David H. Wolpert. “Stacked generalization.” In: *Neural Networks* 5 (1992), pp. 241–259.

[111] Weikai Yang, Zhen Li, Mengchen Liu, Yafeng Lu, Kelei Cao, Ross Maciejewski, and Shixia Liu. “Diagnosing concept drift with visual analytics.” In: *IEEE Conference on Visual Analytics Science and Technology.* 2020.

[112] Ting Zhang and Qun Dai. “Hybrid ensemble selection algorithm incorporating GRASP with path relinking.” In: *Applied Intelligence* 44.3 (2016), pp. 704–724.

[113] Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms.* CRC press, 2012.