MSc Artificial Intelligence
Master Thesis

To Trust or Not to Trust a Regressor

Estimating and Explaining Trustworthiness of Regression Predictions

by

Kim de Bie

11077379

15 July 2020

48 ECTS
November 2019 - July 2020

Supervisors:
Prof. Dr. Hinda Haned
Ana Lucic MSc

Assessor:
Dr. Pengjie Ren

University of Amsterdam

Abstract

Machine learning algorithms are often used in hybrid systems in which humans are expected to prevent failure by correcting algorithmic mistakes. However, in these settings, humans have limited tools at their disposal to recognize erroneous predictions. On the one hand, machine learning systems often do not provide information about the reliability of individual predictions. On the other hand, most explainable machine learning methods, which provide insight into how an algorithm has reached a decision, are not specifically equipped to explain errors: predictions are explained without awareness of their correctness.

To address this gap, we propose RETRO-VIZ, which stands for **RE**gression **TR**ust sc**O**res with **VI**suali**Z**ations. RETRO-VIZ is a model-agnostic method for measuring and explaining the trustworthiness of predictions from regression models when the true error is unknown. It consists of two parts: (i) RETRO, a numeric estimate of the trustworthiness of a prediction when the true error is unknown, and (ii) VIZ, a visual explanation for the estimated trustworthiness. The RETRO-score is based on a nearest-neighbor method and assumes that similar instances should receive a similar prediction. If a new instance and its neighbors in the train data are similar, the prediction is trustworthy. If they are not, the prediction is untrustworthy. The new instance and its neighbors are visualized in a VIZ-plot from which the reasons for a low or high RETRO-score can be determined.

To evaluate RETRO, we study the correlation between the RETRO-score and the absolute error in predictions across 117 experimental settings. We find that this correlation is negative in all cases, as expected, which shows that RETRO is useful for recognizing erroneous predictions. We compare the performance of RETRO across models, causes and magnitudes of error and data dimensionalities. In addition, we evaluate VIZ-explanations in a user study with 41 participants. We find that VIZ-plots allow users to identify (i) whether and (ii) why predictions are trustworthy in a task derived from their day-to-day activities and that users experience the plots as helpful in their daily work.

Acknowledgments

This thesis marks the end of a journey that started eight years ago, when I left Zwolle as a fresh high-school graduate. Back then, I certainly had an idea of what I wanted to get out of my student time, but a master's degree in Artificial Intelligence was not part of the plan - if only because I barely knew what it was. And yet, after a degree in a not-very-related field, I decided I would give it a shot. Transitioning into this field has been one of the biggest challenges I've encountered in my student time, but it has been extremely rewarding. Along the way, I have received plenty of support and I would like to use this opportunity to thank a few people who helped me get here.

Firstly, I would like to thank Hinda and Ana, my supervisors, for their encouragement and trust. Thank you for making this project feel like a team effort. This thesis certainly wouldn't have been what it is now without your massive investments of time and wisdom. You made sure that I never felt alone or truly lost and pushed me to have fun in the process, which has been very motivating. I've gotten a lot more out of these past eight months than I expected and I hope it has been rewarding for you too.

Secondly, I am grateful that I had the opportunity to conduct this research at Ahold Delhaize. This allowed me to keep my research deeply rooted in real-world problems and gave me access to amazing colleagues. Bart, Anton, Emanuele - thank you for being great company in the office and as virtual colleagues.

I am very grateful to my friends, who have been a fantastic source of joy and support over the course of my student time. Also, a big thank-you to the DSSG 2019 Warwick-Turing cohort, for an unforgettable summer and for shaping my thinking about all things data.

I would never have made it here without the encouragement of my family. Mama, thank you for always trusting me to find my way. Papa en Annemiek, thank you for unconditionally being there for me. Jeroen en Joep, thanks for being the best brothers I could wish for. This thesis and what it stands for is as much yours as it is mine.

Lastly, Thijs, thank you for always cheering me on and for always believing that all of this was possible, even when I didn't. It would have been a lot more difficult without you. I can't wait to see what's next.

Contents

Chapter 1

Introduction

Machine learning algorithms are increasingly used in high-stakes domains. For example, algorithms are used by judges to predict recidivism [Tan et al., 2018a], by doctors to aid cancer screening [McKinney et al., 2020] and by banks to predict credit card fraud [Awoyemi et al., 2017]. This is not without risk: an algorithm is an imperfect approximation of reality, which is constrained by the model architecture and the data that the model was trained on. As such, a model is bound to make mistakes, especially when it is asked to generalize beyond situations it is familiar with [Amodei et al., 2016]. To prevent failures that would result from an AI system working autonomously, algorithms are often used in hybrid systems where humans aid in the decision-making process. In such 'algorithm-in-the-loop' systems, the responsibility to follow or to deviate from an algorithmic prediction remains with a human [Green and Chen, 2019a]. Humans can decide to disregard a particular prediction if they believe it is erroneous or untrustworthy, thereby acting as safety mechanisms to prevent large errors from being made [Elish, 2019].

Metaphorically, this co-operation between human and algorithm mirrors that between a human pilot in an airplane and the autopilot. Under normal circumstances, the human pilot will not interfere in the decisions of the autopilot. Only when circumstances deviate from the ordinary, the human will take over control [FAA, 2017]. Human pilots are trained extensively to recognize when the autopilot should be switched off and the airplane's system will provide warnings in such situations. In contrast, humans that co-operate with predictive algorithms often do not have such tools at their disposal. Beyond global performance metrics such as the error on a test set, many machine learning systems do not provide estimates of the reliability of individual predictions [Nushi et al., 2018]. In addition, many complex machine learning models, such as deep neural networks, provide the user with little insight into how predictions are reached, which makes it even more difficult to assess whether or not a prediction is trustworthy [Bansal et al., 2019b].

To obtain an optimal co-operation between human and algorithm, it is crucial that the human understands the uncertainty in algorithmic predictions [Wortman Vaughan and Wallach, 2016]. Beyond global error metrics, this requires that humans have an accurate understanding of when the algorithm might be making errors in individual cases [Kendall and Gal, 2017; Nushi et al., 2018]. Some algorithms, such as Bayesian models or neural models with a softmax output, provide a measure of confidence in individual predictions which may help users gauge their trustworthiness. However, many algorithms, ranging from standard neural regressors to decision trees, do not provide such confidence scores [Nushi et al., 2018]. In addition, confidence scores are exclusively numeric assessments of the performance of an algorithm and therefore do not provide any insights into the reasons why a prediction might be wrong. Therefore, the degree to which such confidence scores help humans understand the trustworthiness of predictions is limited [Ribeiro et al., 2016].

Explanations have been proposed as a tool for assessing the trustworthiness of individual algorithmic predictions [Doshi-Velez and Kim, 2017]. A common approach to explaining algorithmic predictions is to provide insights into the weight the algorithm assigns to each input variable, which can help the human assess whether a prediction is reasonable [e.g. Lundberg and Lee, 2017; Ribeiro et al., 2016]. However, a limiting factor of existing explainable AI (or XAI) methods is that many are not specifically equipped to explain errors: predictions are explained without awareness of their correctness. In practice, researchers have struggled to demonstrate that XAI methods allow users to recognize (un)trustworthy predictions [Lai et al., 2020].

1.1 Research Questions

Overall, current methods are insufficient to help humans understand *when* and *for what reason* algorithmic predictions are (un)trustworthy in production settings where the true error is unknown. On the one hand, confidence scores for individual predictions do not provide reasons for an expressed low confidence. On the other hand, most XAI methods do not differentiate between correct and incorrect predictions. To address this gap, we propose a method that provides both a quantitative estimation of the trustworthiness of regression predictions when the true error is unknown as well as a visual explanation for this estimate. The leading research question of this study is therefore:

Can we design a model-agnostic method for estimating and explaining the trustworthiness of regression predictions that helps users understand (i) whether and (ii) why an algorithmic prediction is (not) trustworthy?

This study proposes RETRO-VIZ, or **RE**gression **TR**ust sc**O**res with **VI**suali**Z**ations. It consists of two main parts: (i) RETRO, a numeric estimate of the trustworthiness of a prediction when the true error is unknown, and (ii) VIZ, a visualization that explains the reasons for the estimated trustworthiness.

The goal of RETRO-VIZ is to provide insight into algorithmic error in a way that aids human-algorithm co-operation. An evaluation with human subjects is therefore an important part of this study. Beyond objectively evaluating whether RETRO-VIZ helps users to recognize the trustworthiness of predictions, we want to know whether the proposed method is subjectively perceived as satisfying by potential users. In practice, it is users who decide (not) to use an algorithm or an explanation method. Indeed, "if the users do not trust a model [...], they will not use it" [Ribeiro et al., 2016]. Therefore, potential improvement in objective performance can only be realized when humans accept the method as valuable. For this reason, we are interested in objectively as well as subjectively evaluating RETRO-VIZ with users. Our research questions are the following:

RQ1 Do the estimates of trustworthiness that RETRO produces correlate with the errors in algorithmic predictions, and if so, how?

RQ2 Under which conditions does RETRO perform best, given different (a) model architectures, (b) data dimensionalities and (c) causes and magnitudes of error?

RQ3 To what degree does RETRO-VIZ *objectively* and *subjectively* provide insight into algorithmic performance in a way that aids human-algorithm co-operation? Specifically, we investigate the following:

a. To what extent do VIZ-plots *objectively* help users distinguish trustworthy algorithmic predictions from untrustworthy predictions?

b. To what extent do VIZ-plots *objectively* help users assess the reason for the (lack of) trustworthiness of a prediction?

c. To what extent do users *subjectively* experience VIZ-plots as valuable for assessing the trustworthiness of algorithmic predictions?

1.2 The Need for Explainable Estimates of Trustworthiness at Ahold Delhaize

Methods to improve the understanding of when (not) to rely on an algorithmic predictions are highly relevant for industry practitioners. This research was supported by Ahold Delhaize, a multinational food retailer with 21 brands and stores across 11 countries. The research was executed at the headquarters of the company in Zaandam, the Netherlands and is largely motivated by the needs of its data scientists and analysts. Many of the tasks relevant for Ahold Delhaize are regression problems rather than classification tasks, such as estimating store revenue or estimating the effect of an upcoming promotion. Therefore, this research proposes a method that can be used in regression settings.

Having a grasp on the trustworthiness of algorithmic predictions and understanding whether an algorithm performs sufficiently well in individual cases is important for a wide range of tasks within Ahold Delhaize. Even when complex machine learning algorithms are currently not used for a task, improved tools for assessing algorithmic error can make the introduction of such methods more acceptable. For example, one of the major brands of Ahold Delhaize currently uses a relatively simple, transparent regression model to predict sales for individual stores. Stakeholders have expressed hesitation to adopt more complex methods, as they are worried that the lack of transparency in such methods makes it more difficult to assess whether the model is making mistakes. For instance, for a revenue forecasting system relying on a complex algorithm, it is unknown whether a prediction for a future date will match the actual sales or whether it is erroneous. While for simpler model architectures, the reasoning of the model can be approximately understood, it is much more difficult to understand how a complex model reaches its decisions and therefore whether these predictions are likely to be correct.

Chapter 2

Related Work

This chapter provides an overview of the existing literature that is relevant to the current research. First, we provide an overview of methods related to uncertainty in algorithmic predictions. Next, we outline work in the domain of explainable and interpretable machine learning. Thirdly, we discuss contributions in the field of human-computer interaction, particularly those relating to the design and evaluation of systems in which humans and algorithms collaborate. Lastly, we discuss the concept of trustworthiness as it is used across the three fields discussed in the preceding sections and relate this to the definition we use in our research.

2.1 Uncertainty in Machine Learning

This section focuses on uncertainty estimation for machine learning algorithms. First, we describe why having an understanding of the uncertainty in algorithmic predictions is desirable. Next, we discuss the causes of algorithmic failure, followed by a discussion of existing methods for uncertainty estimation and their limitations.

2.1.1 Motivations for providing uncertainty estimations

With the growth of real-world applications of machine learning methods, the attention for risks and challenges in deploying machine learning algorithms has also increased. Therefore, it is becoming increasingly important to foresee and prevent algorithmic failure. In particular, the field of *AI safety* is concerned with the harmful effects that might occur when artificial intelligence is not designed carefully enough [Amodei et al., 2016]. One of the major risks identified is that of a lack of robustness to adversarial inputs and to distributional changes. On the one hand, inputs to an algorithm can be adversarially manipulated to elicit unwanted behavior [Szegedy et al., 2013]. On the other hand, there may simply be a mismatch between the data the model was trained on and the situation the model encounters in the real world, so that the model's predictions are unreliable.

Algorithmic errors can lead to harmful outcomes directly when algorithmic decisions are applied without human intervention, but may also lead to suboptimal performance in algorithm-in-the-loop systems [Green and Chen, 2019a]. For example, in the medical domain, applying an erroneous and therefore misleading algorithmic prediction can have severe consequences [Ross and Swetlitz, 2018]. In addition, seeing an algorithm fail repeatedly makes users reluctant to use the algorithm, even if the algorithm is shown to outperform a human in general [Dietvorst et al., 2015]. Again, this may have adverse consequences: a reluctance of humans to rely on an algorithm that outperforms them leads to a worse outcome than when the algorithm had been trusted.

To mitigate the risks arising from erroneous algorithmic predictions and to improve the functioning of algorithm-in-the-loop systems, we can make use of uncertainty estimates [Gal, 2016]. For example, when a prediction receives an uncertainty score above a certain threshold, the user may be alerted and asked to verify the prediction or the algorithm may abstain from predicting altogether, so that the responsibility for a decision fully remains with a human [Virani et al., 2019].

2.1.2 Types of uncertainty in algorithmic predictions

There are several reasons why an algorithm may produce erroneous predictions. Identifying different types of uncertainty helps to model such erroneous behavior. Generally, two main categories of uncertainty are distinguished [Kendall and Gal, 2017]. Firstly, *aleatoric uncertainty* is uncertainty due to inherent noise or randomness in the data. It stems from the inherent randomness in some events. For example, it is inherently impossible to predict the outcome of rolling a dice without uncertainty. This type of uncertainty is inevitable in some events and cannot be addressed by creating better models.

Secondly, *epistemic uncertainty* reflects the lack of knowledge in the model. Theoretically, it can be completely solved by creating better models. On the one hand, epistemic uncertainty may arise because the current model is a bad fit for the data in general, so that it overfits or underfits the data. On the other hand, the model may be well-fit to the data that was available during training, but the underlying distribution of the data has changed so that the model no longer fits during inference. Epistemic uncertainty stemming from a misfit between data at train and inference time is called *distributional uncertainty* or uncertainty due to *dataset shift* [Candela et al., 2009].

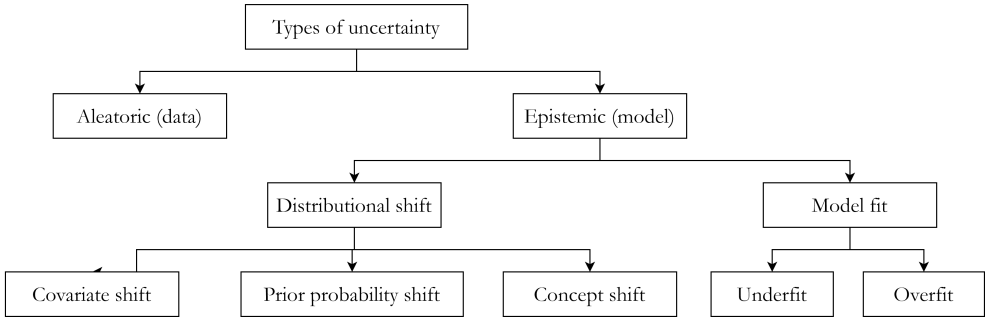


Figure 2.1: A (non-exhaustive) overview of different types of uncertainty in algorithmic predictions.

Distributional uncertainty arises from various types of dataset shift, which can have several causes. For example, it may be that the training data was not sampled from the full distribution of the data. Even if the data was sampled appropriately, the nature of the data may simply change over time, causing a natural dataset shift. Here, the three main types of dataset shift are discussed; for a full discussion we refer the interested reader to Candela et al. [2009].

Firstly, *covariate shift* means that the distribution of the independent variable x changes. While the relationship between x and y does not change, the shift in distribution may imply that the model does not fit the new data well and may produce erroneous predictions. This type of dataset shift is visualized in Figure 2.2.

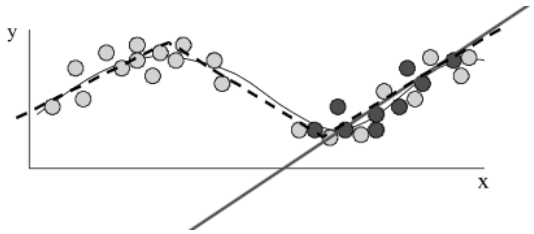


Figure 2.2: Covariate shift visualized. The darker dots represent the data available at train time; the straight line fits these points best. The lighter dots represent the data at inference time, for which the straight line is not a good fit. From Candela et al. [2009].

Secondly, *prior probability shift* entails that the distribution of the dependent variable y changes, while x remains the same. Then, the model trained on the old distribution of y will produce predictions that do not fit the new data. This situation is visualized in Figure 2.3.

Lastly, in *concept shift*, the relationship between x and y changes. While both x and y may independently still come from the same distribution, the relationship between them is now different. Again, in this situation, the trained model will not fit the data at inference time.

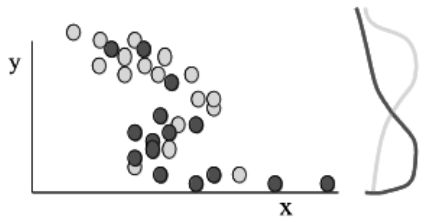


Figure 2.3: Prior probability shift visualized. The darker dots represent the data available at train time; the darker line fits these points best. The lighter dots represent the data at inference time, for which the lighter line is the best fit. From Candela et al. [2009].

2.1.3 Methods for uncertainty estimation

Given the causes of algorithmic error as described in the previous section, we now turn to the methods that have been proposed to estimate the uncertainty in algorithmic predictions. We also highlight the Trust Score as proposed by Jiang et al. [2018], which forms the basis of the method described in Chapter 3.

Uncertainty estimates as default output Some algorithms provide a measure of confidence in their predictions by default. For example, the output layer of a neural classifier usually contains a softmax function which returns a probability for each class [Goodfellow et al., 2016]. Other algorithms that return probabilities include for example Bayesian models, or random forest classifiers where probabilities over the trees of the random forest can be extracted. These probabilities can be leveraged by the user to understand whether a particular prediction should be trusted or whether the prediction is likely to be incorrect. Algorithms which provide uncertainty estimates in their outputs therefore seem a natural choice in scenarios where probabilities are required.

Limitations of default uncertainty estimates Uncertainty estimates that are derived directly from the algorithm itself suffer from two main limitations. Firstly, they have been shown to be poorly calibrated. An uncertainty estimate is calibrated when it corresponds to a probability. For example, in classification, it would be desirable for 30% of the samples for which the uncertainty estimate is 0.3 to actually have the predicted label. In practice, uncertainty estimates are often not calibrated well [Guo et al., 2017]. Secondly, the estimates are often unreliable: for example, for adversarial inputs, neural networks can predict the wrong class for an image with very high confidence [Goodfellow et al., 2014]. This can be understood through the lens of covariate shift: while the adversarial example does not resemble any of the training inputs, it still lies on a part of the input space for which the model exhibits high confidence.

Other approaches to uncertainty estimation Several approaches have been suggested to remedy the limitations of algorithms that directly provide uncertainty estimates. Firstly, calibration methods have been proposed to alleviate the mismatch between model uncertainty outputs and true probabilities [Guo et al., 2017]. However, calibration does not solve the issue of reliability [Jiang et al., 2018]. Secondly, related approaches attempt to model uncertainty directly through changing the architecture of the model [Gal and Ghahramani, 2016; Lakshminarayanan et al., 2017; Papernot and McDaniel, 2018]. However, in practice, it may not always be feasible to alter the model architecture, for example when a model is already in production.

Trust Scores Recently, several methods have been proposed to estimate predictive uncertainty while being *model-agnostic*, i.e. without depending on particular characteristics of the model architecture. Jiang et al. [2018] proposed a 'Trust Score', which provides a measure of confidence in individual predictions from a classifier. It estimates the *trustworthiness* of a prediction by measuring the agreement between a classifier and a modified nearest-neighbor classifier on a test example. Intuitively, if a new instance is classified into a very different class than similar instances in the train data, there is reason to believe that the prediction might be faulty. Then, a prediction is *trustworthy* if it is reasonable in light of the training data in the sense that the training data behaves in a similar way to the new instance.

We note that trustworthiness in this context is not identical to the accuracy of a prediction, but Jiang et al. [2018] show that there is a strong correlation between the two. Secondly, we note that the intuition behind the method is similar to the basic idea behind case-based reasoning (CBR) methods, which build on the assumption that similar problems should have similar solutions [Aamodt and Plaza, 1994; Kenny and Keane, 2019; Li et al., 2018].

The method proposed by Jiang et al. [2018] assumes an already-trained and possibly non-transparent, highly complex classifier and learns the confidence scores separately. The Trust Score is calculated as follows. First, for each class $l \in Y$ a high-density set $H_\alpha(f_l)$ is established, which contains the training data examples that belong to class l with outliers removed. Here, outliers are those points with the largest Euclidean distance to the other points in the class, where the fraction of points to be removed equals $1-\alpha$. The Trust Score ξ for point x given classifier h is defined as:

$$\xi(h,x) = \frac{d\left(x, H_\alpha[f_{\tilde{h}(x)}]\right)}{d\Big(x, H_\alpha[f_{h(x)}]\Big)} \tag{2.1}$$

with d as a distance function (e.g. Euclidean distance) and

$$\tilde{h}(x) = \underset{l \in Y, l \neq h(x)}{\operatorname{argmin}} d\big(x, H_\alpha[f_l]\big) \tag{2.2}$$

i.e. the nearest-not-predicted class. A visual example of how the Trust Score works is provided in Figure 2.4.

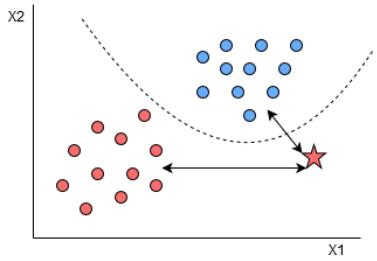


Figure 2.4: Visual explanation of the Trust Score as developed by Jiang et al. [2018]. The red and blue circles represent the two-dimensional train instances which are perfectly separated by a classifier along the dotted line. The star represents a new instance which is classified into the red class. However, as the star lies much closer to the blue class (the nearest-not-predicted class), it is likely that this prediction is wrong.

Limitations of existing methods for post-hoc uncertainty estimation The Trust Score captures uncertainty due to distributional shift: when points are far removed from the training data, they will receive a low Trust Score. However, as Rajendran and LeVine [2019] showed, the method does not explicitly capture model uncertainty or aleatoric uncertainty and fails to recognize algorithmic error arising from these causes. Another significant limitation of Trust Scores is that they are unnormalized (i.e. they have no fixed range), which makes them impossible to interpret in isolation.

While Rajendran and LeVine [2019] proposed an alternative to the Trust Score that captures a wider range of uncertainty types and that produces normalized scores, their method still suffers from several limitations. Firstly, the methods as proposed by Jiang et al. [2018] and Rajendran and LeVine [2019] are only applicable to classification problems and do not have a straightforward extension to regression algorithms. Secondly, both methods only provide a numeric output, which does not inform the user of the reasons for the (lack of) confidence in a prediction. This is insufficient in algorithm-in-the-loop scenarios [Green and Chen, 2019b] in which humans are expected to make informed decisions about updating algorithmic predictions which they do not trust. To do so, an understanding of *why* a particular prediction is (un)trustworthy is essential.

2.2 Explainable and Interpretable Machine Learning

Given the limitations of exclusively numeric uncertainty estimates as described in the previous section, *explainable AI* or *XAI* methods have been proposed as an alternative way to help humans assess whether algorithmic predictions are trustworthy [Ribeiro et al., 2016]. Here, the underlying assumption is that an increased insight into how the algorithm has reached a prediction can help humans assess whether the prediction is credible. Without an explanation, it is very difficult to understand the reasoning of many machine learning methods that are in use today: the recent increases in performance of machine learning methods have come at a cost of increasing complexity [Schmidhuber, 2015]. As a result, for many of the predictive algorithms that are currently in fashion, it is not straightforward for the human to understand how the algorithm has reached a decision, even when uncertainty estimates are available. Increasing the interpretability of complex models helps users to assess the functioning of an algorithm and therefore can help users make better use of algorithmic predictions [Doshi-Velez and Kim, 2017].

In the remainder of this section, we provide an overview of the existing literature on AI explainability and interpretability. First, we discuss motivations for the development of interpretability methods. Next, we describe existing methods, followed by a discussion of their limitations.

2.2.1 Why do we need interpretability?

Interpretability and explainability are often used interchangeably [Lipton, 2018]. In an attempt to draw a distinction between the two, Guidotti et al. [2018] define interpretability as "the ability [...] to provide meaning in understandable terms to a human". An explanation is a way to make an algorithmic prediction more interpretable, or "an interface between humans and a decision maker that is at the same time both an accurate proxy of the decision maker and comprehensible to humans". Thus, while interpretability or providing meaning to humans is the goal, an explanation is a method to achieve it. Four main motivations for the development of methods that improve algorithmic interpretability can be distinguished.

1. **Improving trust in algorithms.** Interpretable methods can help improve human trust in algorithmic predictions. Previous research indicates that humans often tend to mistrust algorithmic predictions. Having insight into the performance of a model (i.e. having seen it fail or outperform humans) alone does not resolve this aversion; on the contrary, it seems to reduce trust even further [Dietvorst et al., 2015]. It is argued that explanations can help alleviate the skepticism towards opaque algorithmic predictions [Ribeiro et al., 2016]. By providing insight into the reasoning process of an algorithm, rather than showing its predictions alone, humans might be more inclined to trust the algorithm.

2. **Legal and societal demands.** There are increasing legal as well as social demands to provide interpretable methods. For instance, the GDPR, an European Union-wide regulatory

framework, requires that affected individuals are provided "meaningful information about the logic" behind algorithmic decisions that are applied to them [Wachter et al., 2017a]. Such requirements are also societal: studies have found that people are uncomfortable with being the subject of fully-automated decisions that they do not understand [Binns et al., 2018; Eslami et al., 2015].

3. **Aid to safeguard external objectives.** Improving interpretability of algorithms may help humans safeguard external goals. To exemplify this, Lipton [2018] describes a hiring algorithm, which should optimize for productivity but should also take ethical and legal requirements into account. However, the algorithm itself can only optimize for productivity. Explanations that provide insight into the factors the model has relied on to achieve a decision may help humans understand whether ethical and legal requirements are met. Nonetheless, Lakkaraju and Bastani [2020] showed that misleading explanations can be produced, which show that certain factors were not important for an algorithmic prediction when they actually were. This makes relying on explainable AI to safeguard external objectives risky.

4. **Error detection.** Model explanations may help with debugging and detecting errors: if the algorithm is making decisions on the basis of irrelevant factors, this could be an indication that there is something wrong [Ribeiro et al., 2016]. In this sense, explanations can be seen as complementary to uncertainty estimations as described in Section 2.1. In a study of how organizations use explanation methods in practice, Bhatt et al. [2020] found that this is the most common use case for explanations. This is quite surprising, given that existing explanation methods are not explicitly tailored for this purpose: all model predictions are explained in the same way, without awareness of the correctness of the prediction. Moreover, in a user study, Kaur et al. [2020] showed that of all potential objectives, users find that explanations are least equipped to detect errors. In general, researchers have struggled to demonstrate that explanations lead to an improvement in decision quality [Lai et al., 2020].

2.2.2 Methods in interpretable machine learning

This section discusses existing methods in interpretable machine learning which have been designed to satisfy the objectives outlined in the previous section. A taxonomy of existing interpretability methods can be found in Figure 2.5. Some methods, such as linear regression methods or shallow decision trees, are intrinsically interpretable. This refers to degree to which algorithmic predictions can be understood through 'introspection' or by simply looking at the model itself [Biran and Cotton, 2017]. For more complex models that are not readily interpretable, explanations have to be produced in another way. One approach is to jointly learn a prediction and an explanation [e.g. Alvarez-Melis and Jaakkola, 2018; Guo et al., 2018]. Thus, a complex model is augmented with the ability to provide its own explanations. Alternatively, *post-hoc* methods provide explanations

without changing the model itself. An advantage of this approach is that the model functionality and therefore its performance are not compromised.

Post-hoc methods can be separated into methods that provide either global or local explanations. Global methods aim to express how the algorithm functions in general [e.g. Tan et al., 2018b], while local methods aim to provide insight into individual predictions [e.g. Lundberg and Lee, 2017; Ribeiro et al., 2016; Shrikumar et al., 2017]. In this research, we focus on *post-hoc*, *local* methods as the method we develop falls into this category.

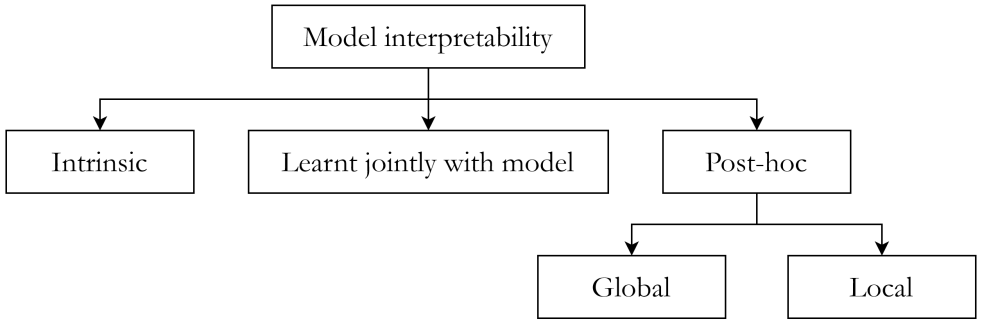


Figure 2.5: A taxonomy of interpretability methods.

Existing post-hoc, local methods provide several types of explanations. With a focus on explanations for tabular data (rather than text or image data), we provide an overview of the most prominent types of explanations.

1. **Feature attribution methods** provide feature importances, which show which aspects of the input were most important in creating a prediction. LIME [Ribeiro et al., 2016] and SHAP [Lundberg and Lee, 2017] are examples of feature attribution methods that have received much attention in the literature. LIME finds feature weights by approximating the complex model with a simpler, inherently interpretable model in a neighborhood around a particular instance. SHAP is based on the game-theoretic concept of Shapley values. The method determines feature importances by iteratively switching individual features on and off to test all possible subsets of features. For both methods, an explanation for a regression problem could simply consist of a table listing the input's features and their weights.

2. **Counterfactual methods** explain what must be changed in the feature values to achieve a desired prediction [e.g. Grath et al., 2018; Tolomei et al., 2017; Wachter et al., 2017b]. A counterfactual explanation of a prediction provides the minimal change to the feature values that changes the prediction to another, predefined output. In this way, a counterfactual explanation provides an answer to the question: "why did you predict this and not something else?" An advantage of counterfactual explanations is that they are actionable, as they provide insight into what actions should be taken to change the output of an algorithm.

3. **Prototypes and/or criticisms** can provide insight into the workings of a model and how it behaves for different inputs [e.g. Kim et al., 2016]. A prototype is an example that is typical of a (set of) predicted value(s), while a criticism is an instance that is not well explained by the prototypes. As such, prototypes and criticisms can help humans build a mental model of the data space underlying the algorithm.

In addition to these approaches, which are the main categories that can be distinguished in the literature, Lucic et al. [2020] propose a method that is particularly relevant to our research as it explains errors produced by regression models, which is similar to what we propose. The method explains large regression errors by showing which features fall outside the range for which a model can produce reasonable predictions. However, this method can only be used to explain *known* errors, i.e. errors for which the ground-truth target value is available. In contrast, we propose a method that can be used in production settings where the true error is not known and must be estimated.

2.2.3 Limitations of current explainability approaches

Despite the development of a multitude of interpretability methods as outlined in the previous section, the field of interpretable machine learning has been criticized by various scholars. These critiques can be clustered into four main topics.

1. **Lack of grounding in existing literature.** Multiple studies haved pointed out that XAI has largely disregarded findings in other fields, such as the social and cognitive sciences [Alvarez-Melis et al., 2019; Miller et al., 2017; Mittelstadt et al., 2019]. These fields have a long history of scholarship discussing the nature of explanations and what makes for a good explanation. Miller [2019] provides an overview of explanations as studied in the social sciences and urges explainability researchers to incorporate this work into their methods.

2. **Exclusive focus on 'model internals'.** Yin et al. [2019] argue that existing work in interpretability has only focused on so-called 'model internals'. There is an almost exclusive focus on providing explanations of the predictions of an algorithm in terms of the importance of input variables. Thus, as argued above, explainability and interpretability are often conflated and there is very little thinking about what makes AI systems more interpretable beyond providing explanations for predictions.

3. **Failure to incorporate user preferences.** Empirical studies have shown that users do not always like explanations as they are produced by recent methods [Narayanan et al., 2018] and fail to leverage them to assess the quality of algorithmic predictions [Lai and Tan, 2019; Lai et al., 2020; Poursabzi-Sangdeh et al., 2018]. This is problematic, as explanations that are perceived as meaningless can lead to a lack of trust and an underestimation of the system's performance [Nourani et al., 2019]. Related to the second line of criticism, this raises

the question of whether explanations are the best mechanism to achieve interpretability, or whether the focus of XAI should be widened to include other mechanisms for increasing the transparency of complex methods [Lipton, 2018].

4. **Impossibility of globally faithful explanations.** Rudin [2019] argues that trying to explain a non-transparent model is an inherently flawed undertaking and that providing globally faithful explanations is impossible. After all, if an interpretable model existed that globally produced the same results as the non-transparent model, the non-transparent model would have been redundant. Therefore, she argues, there should be a focus on the development of methods that are inherently interpretable.

2.3 Insights from Human-Computer Interaction

In this section, we discuss contributions from the field of human-computer interaction relating to uncertainty estimation and explainability. The goal of RETRO-VIZ is to help humans make better use of algorithmic decisions by increasing their understanding of *when* and *why* algorithms fail. Thus, the developed method aims to improve human-algorithm co-operation in situations where algorithmic predictions are not automatically applied, but where humans can choose to override such predictions. Green and Chen [2019b] describe such systems as 'algorithm-in-the-loop' systems. The field of human-computer interaction (HCI) studies how humans and algorithms interact and provides ample insights into human-algorithm collaboration. Here, we specifically focus on insights pertaining to the interaction between human and algorithm given the uncertainty in algorithmic predictions.

In what follows, we first discuss insights from HCI about the need for an accurate understanding of the uncertainty in algorithmic predictions. Then, we link this to the concepts of algorithm aversion and over-reliance, which provide further insight into why misunderstanding algorithmic uncertainty is problematic. Lastly, we explore the insights that HCI provides into why explanations can increase user understanding of algorithmic uncertainty. In Figure 2.6, we provide a schematic overview of the main concepts introduced in this section and their relationships to each other.

2.3.1 Why an accurate understanding of uncertainty is essential

Bansal et al. [2019a] use the concept of a *mental model* to explain why an accurate understanding of uncertainty is essential in algorithm-in-the-loop scenarios. When a human is tasked with deciding how to make use of an algorithmic prediction, she must develop a mental model or a subjective understanding of the capabilities of the algorithm [Bansal et al., 2019a]. Ideally, if the algorithm is correct, the user should recognize that this is the case and should adopt its prediction; similarly, the user should recognize erroneous predictions so that they can be adjusted. However, users often have limited tools at their disposal to create a mental model about the uncertainty in algorithmic

predictions. Common global performance metrics, such as the error on a test set, do not inform the user about when and how the algorithm fails in individual cases [Nushi et al., 2018]. For example, the error rate may not be uniform across all segments of the input space.

The lack of information about model uncertainty is likely to lead to a *representation mismatch*: a discrepancy between a user's mental model or subjective understanding of how well an algorithm performs and the objective performance of the algorithm [Bansal et al., 2019a]. If the mental model and the actual capabilities of the algorithm do not match (i.e. the user accepts algorithmic predictions when they are wrong and rejects them when they are correct), the performance of the system as a whole can degrade [Bansal et al., 2019b]. This shows that an increased objective performance of an algorithm does not automatically translate to an increased end-to-end performance in hybrid human-AI systems. Therefore, aligning a user's mental model and the capabilities of the algorithm is crucial for optimal system performance.

2.3.2 Algorithm aversion and algorithm over-reliance

In this section, we discuss the phenomena of *algorithm aversion* and *algorithm over-reliance*, which are the two main ways in which a user's mental model of an algorithm can be flawed. As such, this section provide further insight into why a *representation mismatch* or a mismatch between a user's mental model and the factual performance of the algorithm is problematic in practice.

Dietvorst et al. [2015] find that users become averse to algorithms after seeing them fail, even when the algorithm outperforms a human in general. In a follow-up study, the authors find that this effect is mitigated to some extent when users can override an algorithmic prediction [Dietvorst et al., 2018]. Somewhat contradictory, Springer et al. [2017] show that people trust an algorithm when it is framed as intelligent even if its performance is actually random, which is known as *algorithm over-reliance*. This creates a dilemma: do humans trust algorithmic predictions *too much* or *too little*?

Lee [2018] resolves this dichotomy by focusing on the characteristics of the tasks that algorithms are asked to perform. She finds that in mechanical tasks, humans are (too) willing to rely on algorithms, whereas algorithm aversion occurs for tasks that are perceived as requiring human skill. In addition, Kocielnik et al. [2019] find that setting correct expectations about the uncertainty in a model beforehand aids the acceptance of the model. In the following section, we explore how explanations can help users to develop an accurate mental model of algorithmic uncertainty by decreasing both algorithmic aversion and algorithm over-reliance, and therefore may help to improve overall performance in hybrid human-AI systems.

2.3.3 The role of explanations in perceived algorithmic uncertainty

The importance of intuitive predictions Further exploring the concepts of algorithmic aversion and over-reliance, Elmalech et al. [2015] show that users tend to rely on predictions from an algorithm more when these intuitively make sense to them. In contrast, when algorithmic predictions are not intuitive, users tend to reject them, even if their own predictions are of worse quality. This shows that it is crucial that algorithmic predictions are intuitive to users, as non-intuitive predictions can lead to algorithmic aversion.

Explanations as aid to human intuition In absence of an explanation, humans can only rely on their existing intuition about reasonable relationships between inputs and outputs to assess the quality of a prediction from a complex model. This is problematic, as human intuition is often erroneous, particularly about statistical notions [see e.g. Granberg and Brown, 1995]. Explanations can help humans to adjust and improve upon their existing intuition: an explanation can help to make something counter-intuitive seem much more reasonable [Binns et al., 2018]. In this way, XAI methods can help to improve human-algorithm collaboration, as they can help to increase intuitiveness of correct but unintuitive predictions, thereby improving the user's mental model of an algorithm.

Non-intuitive explanations lead to algorithmic aversion Not all explanations improve intuitiveness and non-intuitive explanations can increase algorithmic aversion. Nourani et al. [2019] find that people underestimate an algorithm's accuracy when it provides weak explanations that are not 'humanly meaningful' in the sense that they do not align with human intuition. For example, in a cat classification task, an explanation that focuses on the animal's facial features is more meaningful than an explanation that focuses on non-deterministic and background areas. In a qualitative study, Binns et al. [2018] largely confirm these findings and show that explanations that do not match the reasoning process of humans can lead to a decreased trust in the algorithm.

Misleading explanations lead to algorithmic over-reliance Providing explanations to improve user understanding of algorithmic uncertainty is not without risk. Lakkaraju and Bastani [2020] show that unfaithful explanations can lead humans to believe that an algorithm performs better than it actually does. In this light, it is crucial that explanations are evaluated on the extent to which they improve a user's mental model and therefore the quality of human-algorithm collaboration. Doshi-Velez and Kim [2017] distinguish application- and human-grounded assessment of explanation methods from evaluations without real users and argue that the former are essential to understand the impact of explanations on human-algorithm co-operation.

Evaluating explanations in context The work by Poursabzi-Sangdeh et al. [2018] is a rare example of evaluation of model interpretability in a human-grounded task. In this study, the authors evaluate whether humans co-operate more efficiently with algorithms depending on the interpretability of the model. Surprisingly, the authors find that (1) people do not follow the predictions of an interpretable model more when it is beneficial to do so; and (2) transparency can in fact hamper people's ability to detect when a model makes large mistakes. This confirms that explainability methods cannot automatically be assumed to lead to better mental models and to improve performance of hybrid human-AI systems, showing that careful evaluation is required.

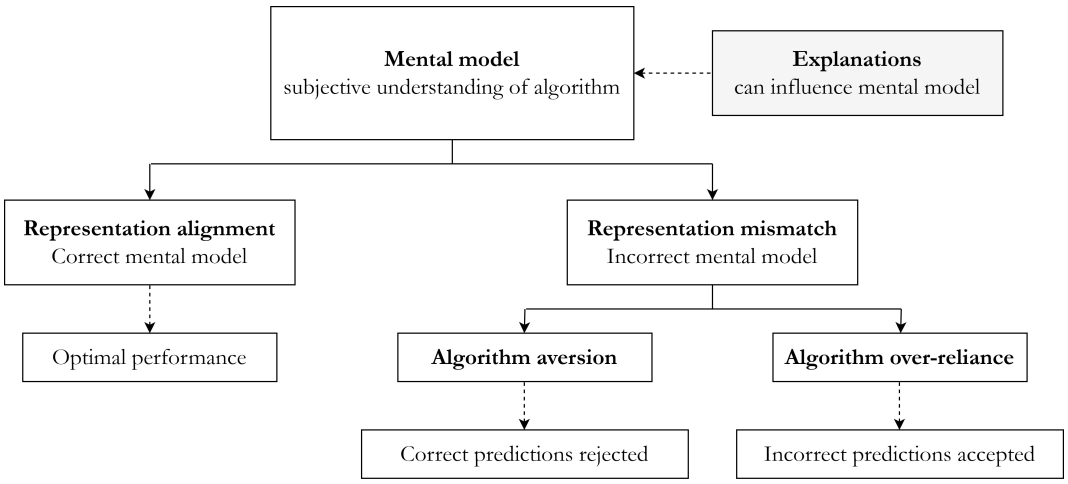


Figure 2.6: Schematic overview of concepts introduced in Section 2.3.

2.4 Trustworthiness in Machine Learning

As outlined in Chapter 1, this research proposes a method that measures the *trustworthiness* of regression predictions. The concepts of trust and trustworthiness are used differently in all three research areas as described in the previous sections. In this section, we outline how trustworthiness is defined in the literature on uncertainty estimation, XAI and HCI, and relate this to the definition of trustworthiness that we use in the current research. In particular, we define what is meant by trustworthiness in the present research and how this must be distinguished from the correctness of a prediction.

Trustworthiness in uncertainty estimation As discussed in Section 2.1, Jiang et al. [2018] propose a 'Trust Score', which estimates the trustworthiness of a prediction by measuring the agreement between a classifier and a nearest-neighbor classifier on a test instance. Then, a prediction is *trustworthy* if it is reasonable in light of the training data in that the train data behaves in

a similar way to the new instance. As the method we propose for the numeric estimation of trustworthiness is heavily inspired by Jiang et al. [2018], we adopt this definition of trustworthiness: a prediction is trustworthy if it is aligned with the data the model was trained on.

The notion of trustworthiness as 'grounded-ness' in train data relates to the different types of epistemic uncertainty as discussed in Section 2.1.2. Namely, for uncertainty due to model over- and underfit, we expect that predictions are erroneous because the model does not fit the underlying data distribution well. As a result, predictions for new instances will not be consistent with the train data distribution. Secondly, for distributional shift, new instances do not resemble the train data at all. However, for aleatoric uncertainty, the uncertainty is not necessarily related to trustworthiness in the sense of alignment with the train data. Specifically, a model may produce a highly erroneous prediction for an instance if it suffers from large aleatoric uncertainty or uncertainty due to inherent randomness, even if the instance and the prediction align closely with the train data.

Based on the above, we explicitly distinguish the *trustworthiness* and *correctness* of a prediction. For their Trust Score, Jiang et al. [2018] show that there is a strong correlation between the two. Similarly, we expect that trustworthy predictions, as they are grounded in the training data, tend to be more accurate and assess this in **RQ1** and **RQ2** (see Chapter 4 and 5). However, we acknowledge that trustworthiness and correctness are not synonymous: trustworthiness does not capture the inherent or aleatoric uncertainty present in algorithmic predictions.

Trustworthiness in XAI In Section 2.2, we mention that 'improving trust' in algorithmic predictions is an important motivation for the development of XAI methods and is featured prominently in work by e.g. Ribeiro et al. [2016]. However, as Lipton [2018] points out, a clear definition of trust or trustworthiness is lacking in the XAI literature. The implicit understanding is that an algorithm is trustworthy if its predictions are based on factors that are reasonable from the perspective of a domain expert instead of on 'spurious correlations' [Ribeiro et al., 2016]. Moreover, Ribeiro et al. [2016] distinguish trust in a prediction, which is a willingness to take action based on a prediction, and trust in an algorithm, which is a belief that a model will behave reasonably if deployed. Likewise, Doshi-Velez et al. [2017] explicitly relate explanations and trustworthiness, as explanations can "help validate whether a process was performed appropriately or erroneously" which helps increase trust in algorithmic predictions. Similarly to the definition used by Jiang et al. [2018], this sense of trust(worthiness) is focused on the accuracy of the model.

Trustworthiness in HCI In Section 2.3, we discuss how a user's mental model or beliefs about the workings of an algorithm influence the willingness to rely on algorithmic predictions. This notion of trust is centered on a belief that the algorithm will produce accurate predictions, similarly to what Jiang et al. [2018] and Ribeiro et al. [2016] propose. However, other authors in the field of HCI use a much broader definition of trust. For example, Lee and See [2004] define trust as "the attitude that an agent will help achieve an individual's goals in a situation characterized by

uncertainty and vulnerability". Similarly, Toreini et al. [2020] relate trust in AI to the belief that an AI system will behave in a way that is generally beneficial to the trustor, which they relate to larger issues of justice and fairness (e.g. does this system discriminate against me?).

Toreini et al. [2020] point out that the 'ability' of a system, which relates to an objective quality of predictions, is only a part of trustworthiness in this sense, but also that trustworthiness cannot exist without it. In our research, we take a much narrower perspective on trustworthiness and focus on ability or the objective quality of predictions alone. Moreover, we focus on trustworthiness of individual predictions rather than of the AI system as a whole.

Chapter 3

RETRO-VIZ: Identifying and Explaining Trustworthiness

In this section, we introduce RETRO-VIZ, a method for identifying and explaining the trustworthiness of regression predictions. RETRO-VIZ stands for **RE**gression **TR**ust sc**O**res with **VI**suali**Z**ations. It consists of two components, namely RETRO, a numeric estimate of trustworthiness of a prediction when the true error is unknown, and VIZ, a visual explanation for the estimated trustworthiness.

Defining trustworthiness RETRO-VIZ measures and visualizes *trustworthiness*, which relates to a notion of reasonableness or credibility in light of the train data (see Section 2.4). Following Jiang et al. [2018], we consider a prediction to be trustworthy if it is aligned closely with the train data. In contrast, if a new instance and its prediction behave very differently from the train data, we intuitively reject that the model is capable of producing a reasonable prediction for the instance and consider it untrustworthy. Thus, identically to the Trust Score method by Jiang et al. [2018] described in Section 2.1.3, trustworthiness in RETRO-VIZ is based on the notion that *similar instances should receive a similar prediction.* If similar instances receive a very different prediction, this indicates that the prediction might be wrong and the estimated trustworthiness should be low. The more dissimilar the predictions, the lower the trustworthiness of the prediction. In addition, if there are no instances that are similar to the new instance in the training data, trustworthiness should also be low: it is unlikely that the model can capture the new instance well.

3.1 RETRO: Numerically Estimating Trustworthiness

The numeric estimation of the trustworthiness of regression predictions is inspired by the method proposed by Jiang et al. [2018], which estimates trustworthiness of predictions by a classifier. Their Trust Score relies on a notion of distance between a new instance and training instances. If a new instance is classified into class A, but in fact lies closer to training examples of class B, trust in the prediction should be low. Building on this notion, we extend Trust Scores to regression with the RETRO-method as outlined below.

Assumptions and prerequisites RETRO requires an already-trained regression model as well as the data that the model was trained on. In order to accommodate all possible regression models, we treat the model as a black box and therefore do not require access to the internals or parameters. The model is trained as a regressor with exclusively numeric input and output variables. Thus, all non-numeric variables have to be removed or have to be cast as numeric variables insofar as this is possible. Beyond these requirements, the method is fully model-agnostic.

RETRO consists of three phases which are outlined in Box 3.1. In Phase 1, we create a reference set based on the training data which forms the basis for the RETRO-score. In Phase 2, we determine the relationship of the new instance to its neighbors in the reference set, where we expect the new instance to resemble its neighbors for a high-quality prediction. In Phase 3, we calculate the RETRO-score and normalize it if desired. In the following sections, we discuss all steps in detail.

Phase 1. Preparing the reference set

1A. Filter largest errors from training data
Remove those instances for which the regressor produces erroneous predictions.

1B. Reduce dimensionality of the data (optional)
Overcome the curse of dimensionality in kNN approaches.

Phase 2. Determining the relationship to neighbors

2A. Find the closest neighbors of the new instance in the reference set
Intuitively, similar instances should have similar target values.

2B. Find average distance to closest neighbors in the reference set
For high-quality predictions, we expect the neighbors to be close.

2C. Find distance to mean target value for closest neighbors in the reference set
For high-quality predictions, we expect the instances to have similar targets.

Phase 3. Scoring and normalization

3A. Calculate the RETRO-score
The larger the distances in 2B and 2C, the lower the RETRO-score.

3B. Normalize RETRO-score (optional)
Make sure the RETRO-score lies between 0 and 1.

Box 3.1: Overview of the three phases of RETRO.

3.1.1 Phase 1 - Preparing the reference set

RETRO relies on a reference set which is based on the training data. As will be outlined in Phase 2, RETRO leverages the distance between the new instance and similar instances from this set to estimate the trustworthiness of a prediction. To create the reference set, we consider the following.

- **The reference set must contain instances that are captured well by the model.** If the regression model has been trained on data that is similar to the new instance (i.e. the model has seen similar inputs before), we assume that the model will be able to make high-quality predictions for the new instance. However, this assumption does not hold when the model has in fact not learned enough about a region of the input space to produce high-quality predictions, despite it being in the training data distribution. We address this in Step 1A by removing the instances with the largest absolute error from the training set.

- **Instances in the reference set must be of appropriate dimensionality.** As will be outlined in Phase 2, neighbors of instances are selected through a k-Nearest Neighbors (kNN) algorithm. However, a limitation of kNN is that it becomes meaningless in very high-dimensional spaces: the distance to the closest neighbor approaches that to the instance that is furthest away, i.e. all instances are equidistant [Beyer et al., 1999]. To resolve this 'curse of

dimensionality', it is desirable to reduce the dimensionality for high-dimensional instances, which we do in Step 1B.

Given these considerations, we describe Step 1A and 1B in technical detail below.

Step 1A: Filter largest errors from training data

Figure 3.1 illustrates the procedure for removing the training instances with the largest error. The training instances are all fed to the trained regressor. The absolute error between the predicted target variable $\hat{y}$ and the ground-truth target variable y is calculated for each instance. The training instances with an error in the α-largest proportion of the instances are removed. The proportion of instances to be removed, α, is a parameter set by the user. The reference set now consists of instances for which the regressor can produce relatively good predictions.

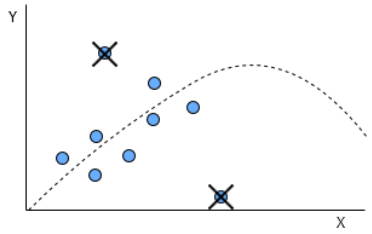


Figure 3.1: Filtering the largest errors from the train data. In this figure, a polynomial regression model is applied to a one-dimensional input. The blue instances represent the train instances x with their ground-truth output variable y. The dotted line represents the learned model. Two instances are far from the fitted line and would contain a large error if they were fed to the model.

Step 1B (optional): Reduce the dimensionality of the data

We note that standard methods for reducing the dimensionality of instances, such as principle component analysis (PCA) or neural autoencoders [Goodfellow et al., 2016], are not sufficient for our purpose. This is because PCAs and neural autoencoders reduce dimensionality in such a way that the original *input variables* can be recovered as closely as possible [Goodfellow et al., 2016]. Instead, we want to maintain those aspects of the input that are important for making the *prediction*, i.e. for retrieving the *target variable*.

To resolve this, we use a Multi-Layer Perceptron. We consider that neural networks project their input data into a space that is eventually simple enough to make a linear prediction [Papernot and McDaniel, 2018]. Then, if we gradually reduce the dimensionality of the input in the layers up to the output layer, we obtain a version of the input that can intuitively be seen as a reduced representation that highlights those aspects that are most important for making the prediction. This reduced version of the input can then be fed to a kNN algorithm (in Phase 2) without suffering from the curse of dimensionality as alluded to above.

Concretely, the procedure works as follows: given the reference instances obtained in Step 1A, we train a Multi-Layer Perceptron. The network has the input features as input and the target feature as output. The penultimate layer of this network (i.e. the layer before the final prediction is made), denoted as layer N, should have the desired, reduced dimensionality. Once trained, both the reference data and the new instance for which the RETRO-score is to be calculated are fed to this network. The transformed input is extracted at layer N. Figure 3.2 provides a visual explanation of this step.

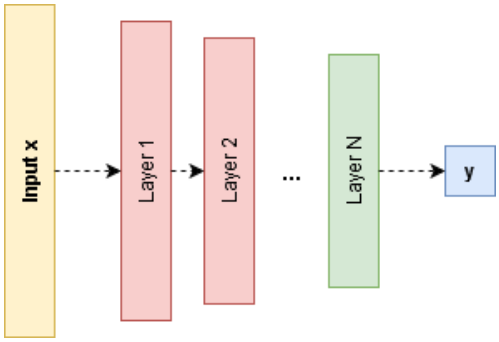


Figure 3.2: Dimensionality reduction through a predictive Multi-Layer Perceptron. The reduced instance is extracted at layer N (in green), which is the last layer before the prediction is made.

3.1.2 Phase 2 - Determining the relationship to neighbors

In Phase 2, we assess whether the predicted target value for the new instance seems trustworthy in the light of the reference set created in Phase 1. For this, we look at the neighbors of the new instance, i.e. the instances in the reference data whose input features are the most similar to the new instance, which we retrieve in Step 2A. For high-quality predictions, we expect that the following relationships exist between the new instance and its neighbors:

- **The new instance lies close to its neighbors.** In contrast, if the new instance has no neighbors at a close distance within the reference set, the new instance is likely to be an outlier. In this case, it is unlikely that the region around the new instance is captured well by the model and predictions for the new instance are likely to be erroneous. Step 2B captures this aspect.
- **The target prediction for the new instance is close to the neighbors' target value.** Given a new instance and the closest neighbors in the reference data, we expect that the new instance should receive a prediction that is close to the ground-truth target variable for the neighboring instances. In constrast, if the distance between the target prediction for the new instance and the target variables of the neighbors is very large, the prediction is considered less trustworthy. We capture this aspect in Step 2C.

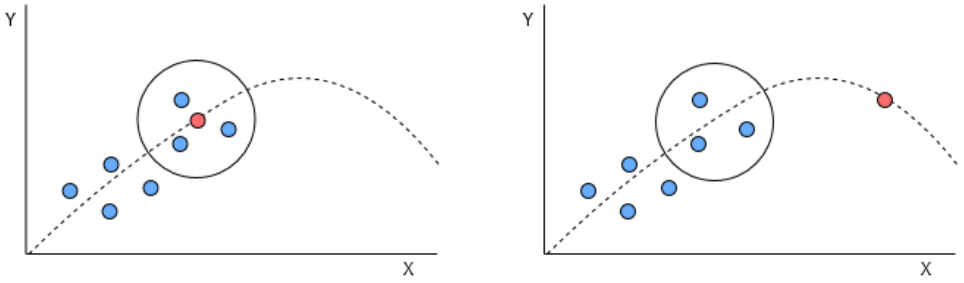
Given these expectations about the relationship between the new instance and the reference set, we describe Step 2A, 2B and 2C in technical detail below.

Step 2A: Find the closest neighbors of the new instance in the reference set

The number of neighboring instances that we want to retrieve is defined as K. To determine which instances are closest to the new instance, we define a distance metric, for which we use Euclidean distance by default. The Euclidean distance δ_k between the new instance p and reference instance q_k is defined as follows, where n is the number of input features:

$$\delta_k(p, q_k) = \sqrt{\sum_{i=1}^{n} \left(p^i - q_k^i\right)^2} \tag{3.1}$$

Then, the K instances with the lowest δ_k are the closest neighbors of the new instance in the reference set. Note that the features must be normalized before calculating the Euclidean distance, to prevent that the distance is biased by the magnitude of the feature values. Secondly, note that we use the reduced instances (obtained in Step 1B) if a dimensionality reduction has been applied. This step is equal to the k-Nearest Neighbors (kNN) algorithm. The selection of neighbors is visualized in Figure 3.3.



(a) New instance close to reference data. (b) New instance far from reference data.

Figure 3.3: Finding the closest neighbors in the reference data (here for $K = 3$). The blue dots represent the one-dimensional reference instances x with their ground-truth output variable y. The dotted line represents the learned model. The red dot has a predicted value y on this line. The blue dots inside the circle are those reference instances closest to the new observation (in the x-dimension). On the left, the new instance lies closer to the reference data than on the right.

Defining distance While we use Euclidean distance, other distance metrics could be equally valid depending on the dataset. For example, cosine distance could be a valid metric if we assume that for a particular dataset, instances at a close angle from each other should receive a similar prediction. Secondly, we weigh all features equally as we observed that this leads to the best performance, but alternatives such as weighing features by their importance could be considered.

Computational complexity A downside of kNN is its computational complexity. When the reference data consists of a very large number of instances, it may be prohibitively expensive to calculate the distance between the new instance and each reference instance. If this is the case, two strategies may be considered to reduce the size of the reference set: (i) set a larger percentage of instances to be discarded in Step 1A, or (ii) take a smaller random sample from the training data before beginning the RETRO-procedure. We discuss potential drawbacks of these strategies in Chapter 8.

Step 2B: Find average distance to closest neighbors

To capture the situation in which new instances are outliers as compared to the reference data, we measure the average distance of the new instance to its closest neighbors, which we denote as d_1. We define d_1 as follows. Given the Euclidean distance δ_k of the new instance to each of its K neighbors as defined in the previous step, we take the average of all δ_k to find the mean distance to the neighbors.

$$d_1 = \frac{1}{K} \sum_{k=1}^{K} \delta_k \tag{3.2}$$

The larger d_1, the more likely that we have encountered *distributional shift* (see Section 2.1.2) and therefore the lower the RETRO-score should be. We visualize this step in Figure 3.4.

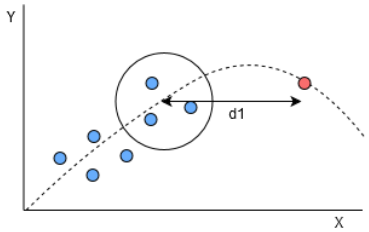


Figure 3.4: Finding the average distance to closest neighbors (in the x-dimension). The blue points represent the reference instances x with their ground-truth output variable y. The dotted line represents the learned model. The red dot has a predicted value y on this line. Because the new instance lies far away from the blue instances, this prediction is not very trustworthy.

Step 2C: Find distance to mean target value for closest neighbors

For high-quality predictions, we expect that similar instances have similar targets. To express this, we first take the average of the ground-truth target variables y_k across all K neighbors to find μ_y, or the mean target value of the neighbors.

$$\mu_y = \frac{1}{K} \sum_{k=1}^{K} y_k \tag{3.3}$$

Next, we determine the absolute distance between μ_y and the predicted value for the new instance $\hat{y}_p$, which we denote as d_2. The larger d_2, the lower the expected quality of the prediction. The calculation of d_2 is visualized in Figure 3.5.

$$d_2 = |\mu_y - \hat{y}_p| \tag{3.4}$$

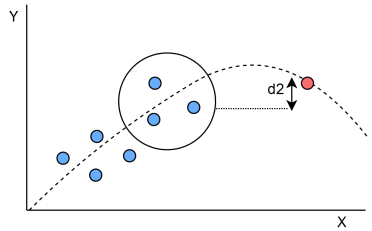


Figure 3.5: Finding the distance to the closest neighbors (in the y-dimension). The blue points represent the reference instances x with their ground-truth target variable y. The dotted line represents the learned model. The red dot has a predicted value y on this line. d_2 is the distance of the red dot's $\hat{y}$ to the average y of the blue dots, which is quite small here.

3.1.3 Phase 3 - Scoring and normalization

In Phase 3, we finally determine the RETRO-score for a prediction. It consists of two steps, which are the following:

- **Calculating the RETRO-score.** Based on d_1 and d_2, we calculate the RETRO-score. The larger d_1 and d_2, the less trustworthy the prediction, so the lower the score should be. This is outlined in Step 3A.
- **Normalizing the RETRO-score.** The RETRO-score R as it is calculated in step 3A has a potential range of $[-\infty, 0]$, given that both d_1 and d_2 are unbounded. This makes R very difficult to interpret in isolation. To overcome this, we normalize R to a range of $[0, 1]$. The method used is outlined in Step 3B. Normalization is optional and can be applied or omitted depending on user requirements.

We describe Step 3A and 3B in technical detail below.

Step 3A: Calculate the RETRO-score

Given d_1, the distance of the new instance to its neighbors, and d_2, the distance of the predicted target variable for the new instance to the mean ground-truth target variable of its neighbors, we can now calculate the RETRO-score R. It is defined as follows:

$$R = -\left(\beta \cdot d_1 + [1 - \beta] \cdot d_2\right) \tag{3.5}$$

Here, β may be used to weight the contribution from d_1 relative to d_2. Setting it to $\beta = 0.5$ leads to an equal contribution of both terms.

Step 3B (optional): Normalize the RETRO-score

To normalize the RETRO-score, we require a minimum and maximum value. We obtain these by calculating the RETRO-score for all instances in the full training set. From these scores, we extract a minimum and maximum, which will form the bounds of the RETRO-scores for new instances. To calculate the RETRO-scores for the training data, we follow the procedure in Step 2A through 3A. Each time, the training instance for which the RETRO-score is calculated is excluded from the reference set. Note that this procedure only has to be executed once for every model for which we want to obtain RETRO-scores, unless the model is trained on new data.

Given the minimum and maximum RETRO-score as obtained on the training data, which we denote as $\tilde{R}_{min}$ and $\tilde{R}_{max}$, we can now normalize the RETRO-score R for the new instance. This is done according to the following formula:

$$R_{norm} = \begin{cases} \dfrac{R - \tilde{R}_{min}}{\tilde{R}_{max} - \tilde{R}_{min}} & \textbf{if } \tilde{R}_{min} \leq R \leq \tilde{R}_{max} \\ 0 & \textbf{if } R < \tilde{R}_{min} \\ 1 & \textbf{if } R > \tilde{R}_{max} \end{cases} \tag{3.6}$$

Interpreting the normalized score By relying on the bounds of the RETRO-score for the training data, the normalized scores on new instances acquire an interpretable meaning. Namely, a normalized score of 0 means that the prediction for an instance is trusted *less* than any of the predictions in the training data. Instead, a normalized score of 1 implies that the new prediction is trusted *more* than any of the training data instances and there is little reason to believe that this prediction is erroneous.

Limitation of normalizing A downside of the normalization procedure is that very low or high RETRO-scores, which are more extreme than the scores found on the training set, now all collapse into a normalized score of 0 or 1. This is especially relevant in cases of distributional shift, where the scores on shifted data instances are likely to be much lower than those found on the training data. Then, with normalization, we lose the ability to differentiate between different severities of distributional shift. This scenario is visualized in Figure 3.6.

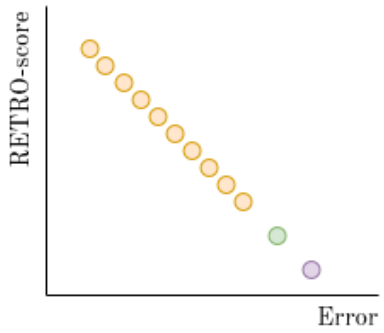


Figure 3.6: The yellow dots are the scores found on the training set vs. their error; the green and purple dots are the scores for new instances. A downside of the proposed normalization procedure is that the purple and green dot will now both receive a RETRO-score of 0, as they lie below the minimum score of the training data.

3.2 VIZ: Visually Explaining the Uncertainty Estimate

After introducing RETRO in the previous section, which helps us understand *whether* predictions are trustworthy or not, we now turn to VIZ, a visualization method which explains *why* predictions are (un)trustworthy. The goal of VIZ is to provide the user with an actionable tool that helps her understand why a particular prediction is (not) reliable in light of the train data. That is, the tool must express that predictions which closely resemble the training data are trustworthy, while predictions which are not backed up by training data are less trustworthy.

Intended users The method is targeted at users who interact with algorithms in 'algorithm-in-the-loop' settings as described in Chapter 1. We assume that such users are likely to be experts in the domain the model applies to, as they have been tasked with making decisions within the domain. However, they are not necessarily the developers of the regression model, nor can they be assumed to have advanced statistical knowledge. Therefore, the explanations must be understandable to data scientists and non-data scientists alike.

3.2.1 Using Parallel Coordinate Plots to explain trustworthiness

RETRO is based on a new instance to which we are applying a regression model, the resulting predicted output variable and K neighboring instances in the training data. The explanation must therefore provide insight into the similarity between these instances (or the absence thereof). However, similarity between multi-dimensional data points is not straightforward to comprehend. To address the difficulty of building a mental image of multi-dimensional data, Inselberg [1985] proposed the Parallel Coordinate Plot (PCP). In a PCP, multiple dimensions of a given dataset are visualized on parallel axes. This allows for identification of subgroups of data and relationships between axes. An example of a classic PCP is shown in Figure 3.7.

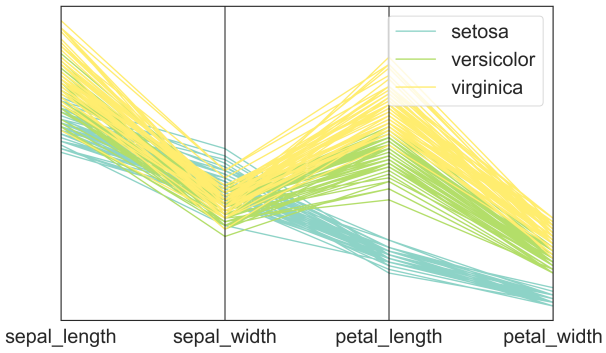


Figure 3.7: A classic parallel coordinate plot (PCP) visualizing Fisher's Iris dataset. The colors indicate the different species. The PCP enables the viewer to discover specific characteristics of the dataset. For example, we observe that 'setosa' tend to have the smallest petal width and length.

While PCPs are ordinarily used for visualizing entire datasets, they can also be used for visualizing a smaller number of instances. We propose to use PCPs for visualizing the new instance and its K neighbors. We name the resulting plots VIZ, or **VI**suali**Z**ations that act as explanations for the scores provided by RETRO.

The length of the feature axes of the VIZ-plot is based on the maximum range of a feature in the train set, so that the bottom of the axis represents the minimum value of the feature and the top represents the maximum value in the train set. If the new instance falls outside this range, the range is extended as needed. In this way, VIZ-plots provide the user with a sense of the distance between instances relative to the training set as a whole. As we will confirm in our user study (Chapter 6), VIZ-plots have two main advantages:

- **VIZ-plots show *why* a prediction is (not) trustworthy.** Based on the value of the RETRO-score, we identify three main types of VIZ-plots. These plots are visually distinct and represent the three main reasons for high/low scores. Firstly, if the RETRO-score is high, the new instance and its neighbors lie close to each other in their feature and target

values (see Figure 3.8). In contrast, if a RETRO-score is low, there are two main reasons for this. On the one hand, it could be that the new instance has no training instances that lie relatively close to it, because the new instance deviates in one or multiple features (see Figure 3.9). On the other hand, the predicted output value can lie far away from the ground-truth output variable from the K neighbors (see Figure 3.10).

- **VIZ-plots allow users to leverage their domain expertise.** This is because they allow the user to inspect deviations in specific features. Perhaps the new instance is anomalous in a particular way that corresponds to a known pattern, so the user knows what to do based on her experience. If the deviation is surprising, the domain expert can investigate the causes of the unusual values of the specific features, which helps to understand whether the deviation is problematic and to design a response. For example, in Figure 3.9, we observe that the new instance is a much larger store (measured by floor_surface) and spends more on advertising than the most similar stores in the train set, so it is likely that true sales are higher than predicted by the model.

While we provide static examples of VIZ-plots in Figures 3.8, 3.9 and 3.10, the plots are interactive in reality: for instance, users may hover over the lines to see exact feature values. A live version of a VIZ-plot can be viewed at `https://www.kimdebie.nl/pages/retroviz.html`.

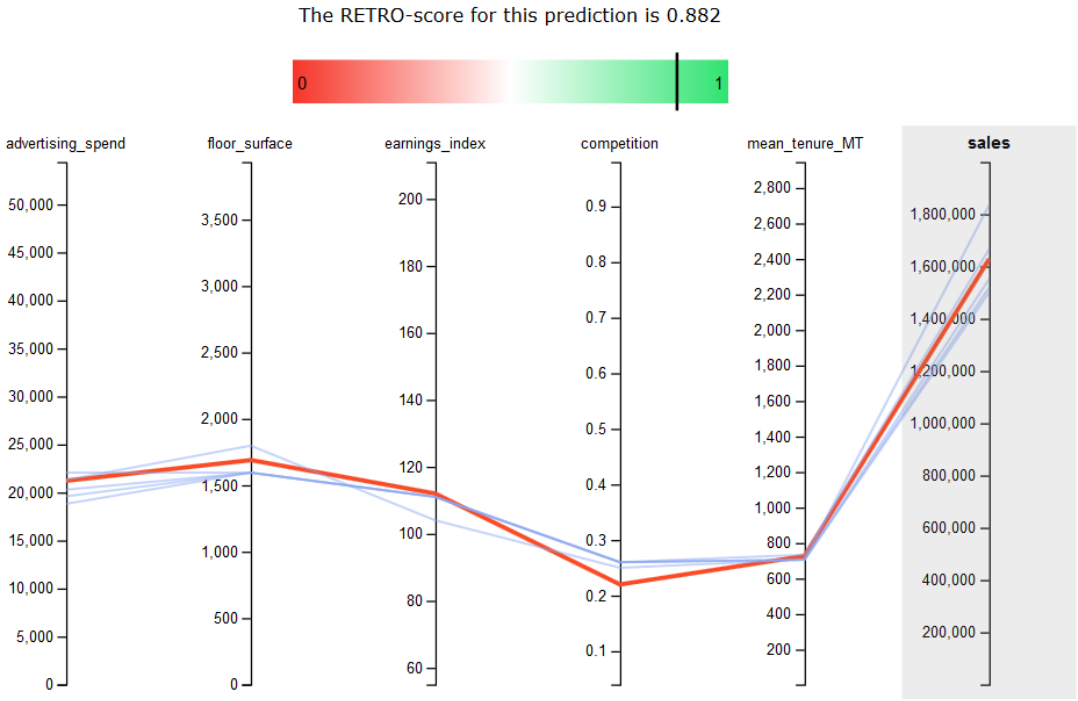


Figure 3.8: Example RETRO-VIZ output for a trustworthy prediction. This model predicts sales based on five features. The RETRO-score is 0.882 for the test instance (red). The VIZ-explanation shows that its nearest neighbors from the reference set (blue) have feature values that are similar to its own as well as a similar predicted target. This indicates that the prediction is well-represented in the training set and that the prediction is trustworthy.

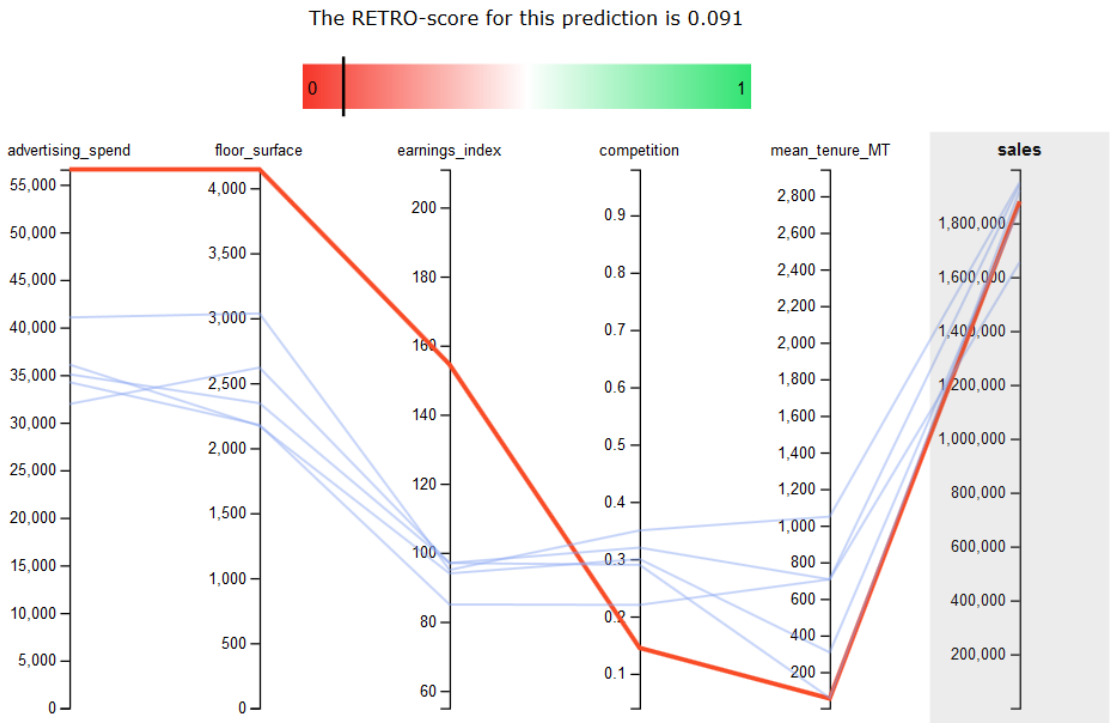


Figure 3.9: Example RETRO-VIZ output for an untrustworthy prediction. This model predicts sales based on five features. The RETRO-score is 0.091 for the test instance (red). The VIZ-explanation shows that its nearest neighbors from the reference set (blue) have feature values that are quite different from its own, even though they have similar predictions. This indicates that the new instance is not well-represented in the training set and the prediction may be unreliable.

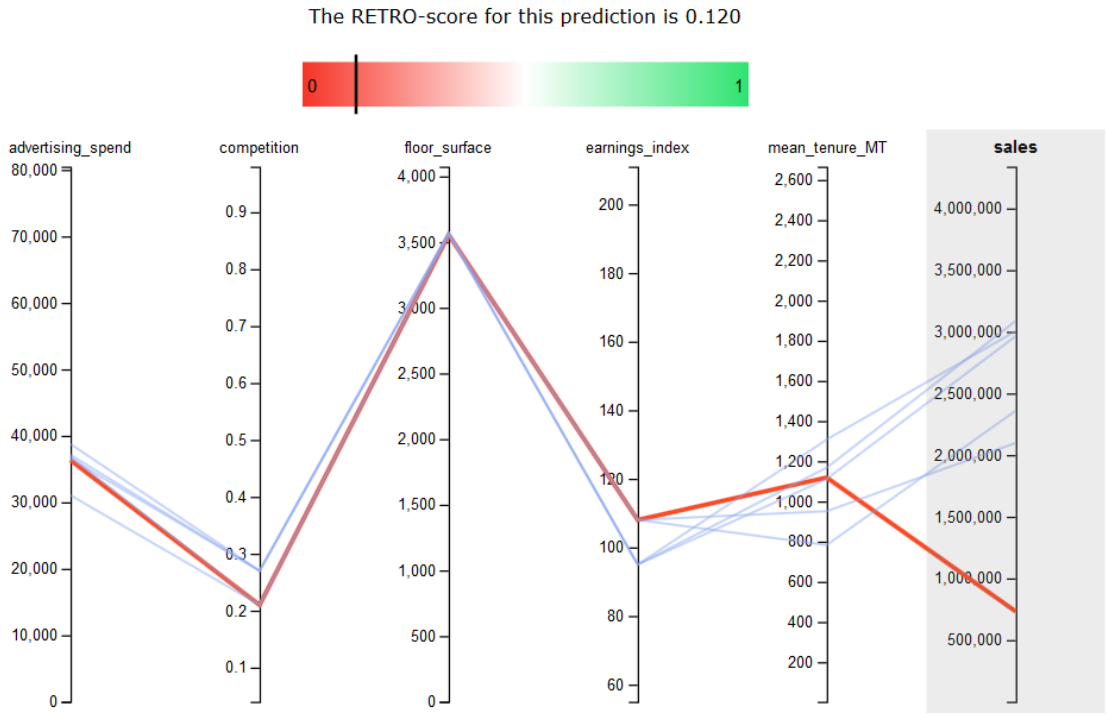


Figure 3.10: Example RETRO-VIZ output for an untrustworthy prediction. This model predicts sales based on five features. The RETRO-score is 0.120 for the test instance (red). The VIZ-explanation shows that its nearest neighbors from the reference set (blue) have feature values that are quite similar to its own, but the predicted value deviates strongly from its neighbors. This prediction is untrustworthy as seems unreasonable in light of the train data and therefore it receives a low score.

3.2.2 Dealing with high-dimensional data

An important limitation of the VIZ plots is that they can only reasonably display a small number of features. We found that plots with roughly 10 variables are still of sufficient quality, but readability of the plots decreases beyond that point. When datasets have more features than this, we must decrease the number of axes that is shown at any one time, for which we identify two solutions. The best solution is highly dependent on a specific use case and user preferences, but both are relatively simple to implement.

1. **Select the most predictive features** in the training data, for example by calculating the SHAP-value for each of the features [Lundberg and Lee, 2017]. In this way, deviations and similarities in the most important features are highlighted.

2. **Rely on user expertise** and domain knowledge to select the most important features to display. In certain scenarios, the user may be interested in deviations (and similarities) in specific features alone.

Reduced dimensions A second issue with regards to dimensionality arises when the inputs have been transformed to a smaller dimension (as in Step 1B of the RETRO method in Section 3.1). In this scenario, the RETRO-score is calculated on the basis of the dimensionality-reduced (and thus transformed) inputs and neighbors and not on the basis of the original features. Nonetheless, we recommend showing the original, non-transformed features in the visualization, if necessary reduced in number according to the strategies described above. As the dimensionality-reduced, transformed inputs have no interpretable meaning, displaying them in the visualization does not help the user to understand the source of any discrepancies, which is why showing the original features is preferable.

3.2.3 Ordering of the axes

In Parallel Coordinate Plots, the ordering of the axes has a large impact on the quality of the resulting visualization. Massie et al. [2004] found that PCPs are easier to read when there is as little vertical movement of the lines across the axes as possible, i.e. when lines do not constantly move between the top and the bottom of the plot across the axes. To reduce this vertical movement, we order axes based on the similarity of the features following the approach by Massie et al. [2004].

The ordering of the axes is done *globally*: ordering is performed once for a sample of the training data and the ordering remains constant across differences instances of the visualization (i.e. when multiple predictions are examined in succession). This is preferred to a reordering for each separate instance, as Massie et al. [2004] found that a constant reordering leads to confusion in interpreting the plots. Lastly, the axis that represents the target variable is always displayed last and is shaded in grey to clearly separate this axis from the input feature axes.

The input feature axes are sorted as follows (see also Massie et al. [2004]). Firstly, a random sample of m instances from the training data is taken. We obtained satisfying results with $m = 100$. It is important that the data is normalized beforehand, as this allows for the comparison of variables with regards to their position on the axis. Next, the similarity between all pairs of axes that represent input features is calculated. The similarity between a pair of axes A_i and A_j is measured using the similarity across the m instances that are located on these axes. An instance i_k consists of n variables so that $i_k = (a_{1k}, ..., a_{nk})$. Similarity is defined as the inverse Euclidean distance between the normalized variable values. Then, the axis similarity across instances $i_1 ... i_k$ for a pair of axes (A_i, A_j) is defined as follows.

$$\text{Similarity}(A_i, A_j) = \sum_{k=1}^{m} \text{similarity}(a_{ik}, a_{jk}) \tag{3.7}$$

After similarity between all pairs of axes has been determined, the most similar pair is placed on the left of the plot. Then, the most similar axis that is still remaining is placed next to it. This continues until all axes have been placed on the plot.

3.2.4 Extensions of the VIZ-plots

In addition to the basic version of the VIZ-plot as it is displayed above, the visualization could be extended in several ways. The usefulness of these extensions is highly dependent on the user context, as well as on the technical expertise of the user: for users with less advanced machine learning knowledge, some of the extensions may be confusing. They include the following:

- **Display feature importance**. The visualization can be extended to include a measure of feature importance, such as the global or local SHAP value of each of the features [Lundberg and Lee, 2017]. This could help the user determine whether the deviation in a particular feature is likely to have a large effect on the prediction quality.
- **Control over neighbor selection**. The user can be given more control over the neighbors that are shown in the visualization. For example, the number of neighbors can be made adjustable, or an option to select specific instances from the train set may be added.
- **Control over features**. The user may be given more control over the features that are shown. These may be limited or extended in number, or specific features may be selected by the user. This is especially helpful when the number of features is too large to display.

Chapter 4

Evaluating the RETRO-score

In this chapter, we describe the experimental set-up that was used to evaluate the RETRO-score. We assess to what extent the trustworthiness scores correlate with error and compare the correlation across a range of experimental settings. This allows us to answer **RQ1** and **RQ2** as described in Section 1.1.

4.1 Data

We evaluate RETRO on 12 public datasets. These datasets originate from the *scikit-learn* library[1] and the UCI ML repository [Dua and Graff, 2020]. In addition, we use an internal Ahold Delhaize dataset, so that we have 13 datasets in total. The datasets were selected such that they cover a variety of domains and sizes. As RETRO requires that all features are numeric, all non-numeric features were discarded from the data (see Section 3.1). Dataset characteristics can be found in Table 4.1.

Train-test split We use a train-test split of $80 - 20$, so that 80% of the data is used to train the models and the RETRO-score is calculated on the 20% of remaining data. Hyperparameters of the regression models applied to the data are determined by the requirements of the experiments (see Section 4.4): experiments are designed such that specific types of algorithmic errors are expected, so the model settings are not optimized using a validation set.

Data normalization Before training the models, the data is normalized. This increases the performance of some of the models [see e.g. Goodfellow et al., 2016]. In addition, it is required for the calculation of the RETRO-scores (see Step 2A in Section 3.1.2). The features (including the target value) are scaled individually, in such a way that they lie between -1 and 1 in the train set.

[1] `https://scikit-learn.org/stable/`

Dataset	Feat.	Observ.	Target	Source
Cycle power	4	9,568	Energy output of a cycle power plant.	UCI ML
Airfoil	5	1,503	Sound pressure level in a wind tunnel.	UCI ML
Store sales	5	33,655	Monthly sales of stores.	Ahold Delhaize
Fish toxicity	6	908	Toxicity of water chemicals to fish species.	UCI ML
Abalone	7	4,177	Abalone age from physical measurements.	UCI ML
Autompg	7	398	Fuel consumptions of different cars.	UCI ML
California housing	8	20,640	Median house value by district.	scikit-learn
Energy efficiency	8	768	Cooling load requirements of buildings.	UCI ML
Diabetes	10	442	Diabetes disease progression.	scikit-learn
Wine quality	11	4,898	Wine quality from chemical properties.	UCI ML
Boston housing	13	506	Median house value.	scikit-learn
Superconductor	81	21,263	Critical temperature of a superconductor.	UCI ML
Communities	100	2,215	Violent crime in communities.	UCI ML

Table 4.1: Datasets on which RETRO is evaluated and their characteristics. We list the name of the dataset, the number of features, the number of observations (for the train and test set combined), the target variable that is to be predicted and the source of the dataset.

4.2 Models

In principle, RETRO-VIZ is model-agnostic and can be applied to any type of regressor. To obtain a clear understanding of when RETRO-VIZ performs well and when it does not, we evaluate the RETRO-score on several models. These include the following:

Linear models: Linear Regression

Neural models: Multi-Layer Perceptron

Tree-based models: Decision Tree, Random Forest

Gaussian models: Gaussian Process Regressor

Support Vector models: Support Vector Regressor

All models are implemented using *scikit-learn*[1]. The settings of the models (such as the depth of the Decision Tree, or the number of iterations used to train the Multi-Layer Perceptron) differ across experiments and are further specified in Section 4.4.

4.3 Parameter Settings for RETRO

RETRO requires several parameters that are to be set by the user. In our experiments, we use settings that we found to work well across a range of datasets and models. The parameters and their settings are the following:

- α is the proportion of instances that is to be discarded from the training set (see Section 3.1.1). We use $\alpha = 0.1$, so that the 10% of instances for which the regression model produces the worst outcomes are discarded.

- For datasets with more than 15 features (*Superconductor* and *Communities*, see Table 4.1), we reduce the dimensionality of the data to 10 using an MLP (see Section 3.1.1). The MLP has three intermediate layers of sizes $[50, 20, 10]$ and an output layer of size 1. The MLP is trained using an Adam optimizer [Kingma and Ba, 2014] for a maximum of 100 iterations or until convergence. The reduced inputs from the penultimate layer (with size 10) are obtained and used as inputs in the calculation of the RETRO-score.

- K is the number of neighbors that is selected from the reference set (see Section 3.1.2). We use $K = 5$.

- β weighs the contribution of the two components of the RETRO-score. The first component corresponds to the distance from a new instance to its neighbors (i.e. the closest instances found in the reference set). The second component corresponds to the distance between the predicted target value for the new instance and the ground-truth target values for the neighbors. We use $\beta = 0.5$ in our experiments, so that both terms contribute equally. See Section 3.1.3 for further explanation of the use of this parameter.

- We do not normalize the RETRO-score (see Section 3.1.3).

4.4 RQ1: Correlation of RETRO with Predictive Error

In this section, we describe the quantitative evaluation of the RETRO-score that is used to answer **RQ1**. The goal of this evaluation is to analyze the degree to which RETRO, which is a measure of the trustworthiness of predictions, correlates with the errors in predictions by regression models.

4.4.1 Metrics

We use two metrics to assess the quality of the RETRO-score. Firstly, we examine the Pearson correlation between RETRO-scores and errors. Secondly, we assess to what extent RETRO is sensitive to large errors in particular by studying the overlap between the largest errors and the lowest RETRO-scores. Here, we describe both metrics in detail.

Pearson correlation The RETRO-score, which is an estimate of the trustworthiness of a prediction in light of the training data, should be aligned as closely as possible with the error in predictions. For a large error, the RETRO-score should be low; a smaller error should correspond to a higher RETRO-score. We assess this quantitatively through the Pearson correlation coefficient

ρ between the RETRO-score and the predictive error. To measure error, we use the absolute error for a given prediction. The Pearson correlation coefficient ρ is defined as follows:

$$\rho_{R,E} = \frac{\text{Cov}(R, E)}{\sigma_R \sigma_E} \tag{4.1}$$

where R equals the RETRO-score and $E = |Y - \hat{Y}|$ i.e. the absolute errors in the predictions.

The Pearson correlation coefficient ρ has a range of $[-1, 1]$. The closer to (negative) 1, the stronger the correlation between the two variables, while a coefficient of 0 indicates that there is no correlation between the variables. In our case, a negative correlation is desirable: when the absolute error in a prediction is large, we want the RETRO-score to be low and vice versa. In addition, the stronger the correlation, i.e. the closer ρ lies to -1, the better.

Overlap between large errors and low RETRO-scores While there is usually some tolerance for smaller errors in algorithmic predictions, in many settings, it is especially risky if large errors go unnoticed. Therefore, it is particularly important that RETRO picks up on the largest (absolute) errors. We measure this by looking at the 50% of the test instances that contain the largest errors and the 50% of test instances that received the lowest RETRO-scores. In the perfect situation, these sets would be identical: the predictions containing the largest errors should receive the lowest RETRO-scores. We measure the fraction of instances that overlap:

$$Overlap = \frac{\#(E^* \cap R^*)}{\#E^*} \tag{4.2}$$

where E^* denotes the set of instances in the test set with the largest absolute error and R^* denotes set of the instances with the lowest RETRO-scores. Note that $\#E^*$ and $\#R^*$ can be used interchangeably in the denominator. This metric has a range of $[0, 1]$ where a value of 0.5 indicates a random performance and values closer to 1 are more desirable. By looking at the 50% of largest errors, we focus on reasonably large errors, but not exclusively on the most extreme errors alone.

4.4.2 Experiments: Analyzing RETRO by error cause

We are interested in the extent to which RETRO-scores can help users recognize error in algorithmic predictions. Lai et al. [2020] point out that explainability methods are often evaluated on their capacity to help users detect failures in algorithmic predictions without regard for the cause of the error and that such evaluations have often been unsuccessful. Instead, Lai et al. [2020] suggest that explainability methods should be evaluated on their capacity to help users detect predictive failures resulting from specific causes, such as a mismatch between train and test data. Rajendran and LeVine [2019] use a similar approach to assess their method for uncertainty estimation, which we adopt in this study: we design our experiments such that we expect model predictions to contain errors that arise from a specific cause. We achieve this by tweaking the **models** and their settings

as well as the **datasets** used to evaluate performance.

Causes of error As outlined in Section 2.1.2, predictive error may arise from several causes. In our analysis, we focus on three types of algorithmic uncertainty, namely uncertainty due to *distributional shift (A)*, uncertainty because of *overfit (B)* and *underfit (C)*. We use these error causes because they represent common sources of algorithmic failure, both during model development (overfit/underfit) and when the model is in production (distributional shift). Details of the experiments are specified below.

Experiment A1. Natural distribution shift

As outlined in Section 2.1.2, distribution shift means that the data used for training a model behaves according to a different distribution than the data that the model is applied to in a new setting. If this is the case, the model is likely to produce erroneous predictions on the test set despite fitting the train data well. In three of our datasets, train and test data can be split such that distribution shift naturally occurs. The datasets are the following:

- The *Store sales* dataset can be split into smaller stores with a monthly sales value of $<$ 2 million euros and stores with a larger sales value, where both groups of stores behave differently. We use smaller stores as train set and larger stores as test set.
- The *Autompg* dataset is designed to predict fuel consumption of cars and is organized by city. Cars from city 1 tend to have different fuel requirements than cars from cities 2 and 3. Therefore, we train the models on data from city 1 and evaluate on data from cities 2 and 3.
- The *Wine quality* dataset consists of white as well as red wines. However, the relationship between chemical characteristics of the wine and quality is different for both types of wine. We train models on the white wines and test on the red wines.

Below, we list the models used to assess RETRO-scores under natural distributional shift, as well as the settings used in our experiments:

- *Linear Regression.*
- *Multi-Layer Perceptron.* Layer sizes: $(50, 20, 10)$. Iterations: 100. Activation function: ReLU. Optimization method: Adam [Kingma and Ba, 2014]. Learning rate (initial): 0.001.
- *Decision Tree.* Maximum depth: 15. Split criterion: MSE.
- *Random Forest.* Number of trees: 100. Maximum tree depth: 20. Split criterion: MSE.
- *Support Vector Regressor.* Kernel: RBF.

Experiment A2. Induced distributional shift

Not all datasets can be split in a way that naturally leads to distribution shift. Therefore, we manipulate the remaining datasets to create an artificial or induced shift. The data is manipulated in such a way that the distributions of the train and test set are different, so that a regressor produces poor-quality predictions for the test set despite fitting the train data well. We disturb 50% of the instances in the test set in such a way that they lie outside the original distribution, i.e. half of the test set suffers from distributional shift. The procedure we use is the following:

1. Find the 30% of most important features in the dataset (from features $(x_1, ..., x_K)$) by fitting a Decision Tree of depth 15 to the data and selecting the features with the highest SHAP value [see Lundberg and Lee, 2017].

2. Disturb these features so that their value lies outside the original distribution. Take the maximum value of the feature k, x_k^{max}. For each instance i, generate noise n_k^i from a Gaussian distribution $n_k^i \sim \mathcal{N}(x_k^{max}, 0.1 \cdot x_k^{max})$ and add this noise to the original feature: $\tilde{x}_k^i = x_k^i + n_k^i$.

This procedure is applied to all datasets that do not contain a natural distribution shift (i.e. all datasets in Table 4.1 except *Store sales*, *Autompg* and *Wine quality*). The models used to assess the performance of the RETRO-scores under induced distributional shift are identical to those used to assess natural distributional shift.

Experiment B. Model overfit

A model overfits when it fits the training data 'too well': it has picked up on randomness in the training data that is not part of the underlying distribution and extrapolates this noise onto new predictions. As a result, the model produces high-quality predictions on the training data (with an error close to 0), but performs poorly on a held-out test set. To induce this type of error, we use model architectures that we found to be particularly prone to overfitting. These are the following:

- *Decision Tree.* Maximum depth: 10000. Split criterion: MSE.
- *Gaussian Process Regressor.* Kernel: RBF.

The RETRO-score is evaluated on all datasets as listed in Table 4.1. No distributional shift is induced, and the train and test set are split randomly.

Experiment C. Model underfit

A model underfits when it is not complex enough to fit a dataset well: the model is too simple to account for the variation in the data. The predictions on both the train and test set are of poor quality. We induce this error by using models that are deliberately too simple to fit the data:

- *Multi-Layer Perceptron.* Layer sizes: (10). Iterations: 5. Activation function: ReLU. Optimization method: Adam [Kingma and Ba, 2014]. Learning rate (initial): 0.001.
- *Decision Tree.* Maximum depth: 1. Split criterion: MSE.

The RETRO-score is evaluated on all datasets as listed in Table 4.1. Like in Experiment B, no distributional shift is induced, and the train and test set are split randomly.

4.5 RQ2: Performance across Model Architectures, Data Dimensionalities and Errors

In this section, we describe the experimental design used to answer **RQ2**. To assess under which conditions RETRO correlates strongly with error and when it does not, we compare performance across a wide range of settings. We compare the performance of the RETRO-scores across the following dimensions:

- **Model architectures**. By testing with a range of models, we assess how RETRO behaves under different architectures.
- **Data dimensionalities**. We apply RETRO to 13 different datasets, which span a range of domains and sizes. This allows us to evaluate whether the number of features influences the performance of the RETRO-score.
- **Error causes and magnitudes**. As stated in Section 4.4, we assess RETRO by triggering specific causes of error by manipulating the data or model architecture. In this way, we can evaluate to which causes of error RETRO-scores are more or less sensitive. In addition, we evaluate to what extent RETRO is sensitive to errors of different magnitudes.

To evaluate performance across these settings, we assess the variation of the Pearson correlation coefficient as specified in Section 4.4. Note that this part of the evaluation builds on the results obtained in Experiments A, B and C (as described in the previous section), but compares them explicitly along the specified dimensions.

Chapter 5

Results of the RETRO Evaluation

In this chapter, we answer research questions **RQ1** and **RQ2** as posed in Section 1.1 through a quantitative evaluation of RETRO. In Section 5.1, we evaluate to what extent RETRO-scores correlate with predictive error, thereby answering **RQ1**. In Section 5.2, we compare the performance of the RETRO-scores across models, data dimensionalities and error causes and magnitudes, which allows us to answer **RQ2**.

Jupyter notebooks to reproduce the experiments are available at `https://www.github.com/kimdebie/retroviz-tutorial`. Note that the *Store sales* dataset, which contains Ahold Delhaize data, is not available for confidentiality reasons, but all other datasets are publicly available.

5.1 RQ1: Correlation of RETRO with Predictive Error

We expect that RETRO, which measures trustworthiness, correlates with the error in algorithmic predictions. If the model produces a prediction that contains a large error, the RETRO-score should be low; conversely, if a model prediction contains a small error, the RETRO-score should be high. As outlined below, this expectation is confirmed in our experiments.

Recap of metrics As outlined in Section 4.4, we use two metrics to evaluate the RETRO-score. Firstly, we look at the Pearson correlation coefficient ρ between the absolute error and the RETRO-scores. Values closer to -1, indicating a strong negative relationship between the error and the RETRO-score, are more desirable. Secondly, we study whether RETRO is responsive to large errors by looking at the 50% of instances that contain the largest error and the 50% of instances that received the lowest RETRO-score. In the perfect situation, these sets are identical and have an overlap ratio of 1.

Main results We first provide an overview of results across all experiments. In total, we use 13 datasets and 9 models, resulting in a total of 117 experimental settings. In all cases, the correlation between RETRO-scores and errors is negative, which confirms our expectation that on average, predictions with larger errors receive lower RETRO-scores. Across all settings, we find an average Pearson correlation ρ of $-\mathbf{0.561}$. All Pearson coefficients are provided in Table 5.1. In addition, we find a mean overlap between the largest errors and the lowest RETRO-scores of $\mathbf{0.732}$. An overview of the overlap coefficients is provided in Table 5.2.

	A. Distributional shift					*B. Overfit*		*C. Underfit*	
Dataset	LR	SVR	MLP	DT-15	RF	GP	DT-10k	MLP-SH	DT-1
Cyclepower	-0.947	-0.370	-0.690	-0.327	-0.372	†	-0.389	-0.906	-0.741
Airfoil	-0.880	-0.739	-0.697	-0.495	-0.567	-0.999	-0.199	-0.909	-0.393
Store sales	-0.062	-0.782	-0.663	-0.614	-0.650	-1.000	-0.534	-0.780	-0.834
Fish toxicity	-0.695	-0.589	-0.563	-0.460	-0.419	-1.000	-0.414	-0.791	-0.456
Abalone	-0.634	-0.427	-0.399	-0.063	-0.109	-0.999	-0.609	-0.552	-0.212
Autompg	-0.566	-0.892	-0.954	-0.017	-0.210	-0.992	-0.542	-0.914	-0.586
Cal. housing	-0.995	-0.607	-0.964	-0.263	-0.261	-1.000	-0.475	-0.271	-0.433
Energy eff.	-0.980	-0.415	-0.948	-0.472	-0.409	-0.734	-0.225	-0.940	-0.458
Diabetes	-0.056	-0.237	-0.246	-0.341	-0.305	-0.751	-0.255	-0.460	-0.147
Wine quality	-0.163	-0.289	-0.512	-0.483	-0.234	-0.999	-0.448	-0.311	-0.212
Boston h.	-0.348	-0.863	-0.793	-0.864	-0.657	-0.699	-0.433	-0.687	-0.412
Supercond.	-0.946	-0.275	-0.361	-0.546	-0.511	-1.000	-0.463	-0.719	-0.745
Communities	-0.880	-0.323	-0.186	-0.718	-0.408	-0.617	-0.604	-0.620	-0.535
Mean	**-0.627**	**-0.524**	**-0.614**	**-0.436**	**-0.393**	**-0.899**	**-0.430**	**-0.682**	**-0.474**
SD	***0.358***	***0.234***	***0.264***	***0.238***	***0.170***	***0.150***	***0.134***	***0.228***	***0.214***

Table 5.1: Pearson correlation coefficient ρ between the absolute error and the RETRO-score for all experimental settings. The greener the cells, the stronger the correlations; correlations closer to -1 are more desirable. See the respective Experiments sections for the model acronyms. †: GP did not lead to overfitting, so the value was omitted.

	A. Distributional shift					*B. Overfit*		*C. Underfit*	
Dataset	LR	SVR	MLP	DT-15	RF	GP	DT-10k	MLP-SH	DT-1
Cyclepower	0.999	0.777	0.998	0.766	0.787	†	0.590	0.868	0.787
Airfoil	0.980	0.907	0.967	0.828	0.881	0.702	0.649	0.874	0.636
Store sales	0.497	0.692	0.663	0.646	0.657	0.669	0.657	0.745	0.843
Fish toxicity	0.901	0.857	0.769	0.714	0.769	0.901	0.681	0.835	0.692
Abalone	0.763	0.715	0.677	0.526	0.541	0.672	0.641	0.694	0.567
Autompg	0.634	0.886	0.789	0.512	0.618	0.800	0.600	0.950	0.725
Cal. housing	1.000	0.801	0.859	0.633	0.617	0.735	0.641	0.638	0.721
Energy eff.	1.000	0.766	0.948	0.623	0.714	0.753	0.519	0.896	0.662
Diabetes	0.533	0.622	0.644	0.644	0.689	0.778	0.622	0.644	0.622
Wine quality	0.570	0.614	0.655	0.625	0.604	0.843	0.637	0.595	0.536
Boston housing	0.627	0.961	0.843	0.863	0.804	0.686	0.647	0.725	0.725
Superconductor	0.905	0.780	0.791	0.675	0.679	0.974	0.593	0.806	0.795
Communities	0.940	0.660	0.740	0.650	0.695	0.750	0.700	0.730	0.645
Mean	**0.796**	**0.772**	**0.796**	**0.670**	**0.696**	**0.772**	**0.629**	**0.769**	**0.689**
SD	***0.197***	***0.110***	***0.122***	***0.102***	***0.094***	***0.094***	***0.046***	***0.111***	***0.090***

Table 5.2: Fraction of instances that overlap between the set of instances from the test set with an absolute error in the top-50% (i.e. the instances containing the largest errors) and the set of instances with the lowest RETRO-score. Ideally, these sets are identical and have an overlap of 1. †: GP did not lead to overfitting, so the value was omitted.

5.1.1 Results of experiments: Analyzing RETRO by error cause

In this section, we analyze the results of our three main experiments as described in Section 4.4.2. We are interested in the performance of the RETRO-scores given different causes of error. Therefore, we assess scenarios with *distributional shift (A)*, *model overfit (B)* and *model underfit (C)* separately. For each error cause, we manipulate regression models and/or data such that predictions on the test set are likely to suffer from the specified error cause. For a full description of the error causes and the models and data used, we refer to Chapter 4. An overview of the error causes and the corresponding experiments is provided in Table 5.3. The performance of the trained models across Experiments A, B and C as measured through the Root Mean Squared Error (RMSE) on the train and test set is given in Appendix A.

Experiment	**Cause of error**	**Description**	**Train performance**	**Test performance**
A	Distributional shift	Change between train and test distribution	Good	Poor
B	Model overfit	Model picks up on random patterns in train data	Extremely good	Poor
C	Model underfit	Model does not fit data well	Poor	Poor

Table 5.3: Causes of error across experiments.

Results of Experiment A. Distributional shift

The Pearson correlations found in Experiment A, in which we assess whether RETRO recognizes errors that result from distributional shift, are provided in Table 5.4 and summarized in Figure 5.1. To reiterate, we examine distributional shift using a Linear Regression (LR), a Support-Vector Regressor (SVR), a Multi-Layer Perceptron (MLP), a Decision Tree with depth 15 (DT-15) and a Random Forest (RF). While results differ across datasets and model types, the Pearson correlation coefficient ρ is negative in all cases, with an average value of $\mathbf{-0.519}$. In addition, we find that the mean overlap between the 50% of instances with the largest absolute error and the lowest RETRO-score is $\mathbf{0.746}$.

As ρ is always negative, more accurate predictions generally receive higher RETRO-scores than less accurate predictions, as desired. These results indicate that the RETRO-scores are helpful to capture errors in predictions resulting from a shifting distribution between the train and the test set. Performance differs across model types: for the tree-based models, the mean ρ is lower (on average $\mathbf{-0.436}$ for the Decision Tree and $\mathbf{-0.393}$ for the Random Forest) than for the other models ($\mathbf{-0.627}$ for the Linear Regression, $\mathbf{-0.524}$ for the Support Vector Regressor and $\mathbf{-0.614}$ for the Multi-Layer Perceptron, on average). We discuss potential reasons for this in Section 5.2.

Dataset	**LR**	**SVR**	**MLP**	**DT-15**	**RF**
Cyclepower	-0.947	-0.370	-0.690	-0.327	-0.372
Airfoil	-0.880	-0.739	-0.697	-0.495	-0.567
Store sales	-0.062	-0.782	-0.663	-0.614	-0.650
Fish toxicity	-0.695	-0.589	-0.563	-0.460	-0.419
Abalone	-0.634	-0.427	-0.399	-0.063	-0.109
Autompg	-0.566	-0.892	-0.954	-0.017	-0.210
California housing	-0.995	-0.607	-0.964	-0.263	-0.261
Energy efficiency	-0.980	-0.415	-0.948	-0.472	-0.409
Diabetes	-0.056	-0.237	-0.246	-0.341	-0.305
Wine quality	-0.163	-0.289	-0.512	-0.483	-0.234
Boston housing	-0.348	-0.863	-0.793	-0.864	-0.657
Superconductor	-0.946	-0.275	-0.361	-0.546	-0.511
Communities	-0.880	-0.323	-0.186	-0.718	-0.408
Mean	**-0.627**	**-0.524**	**-0.614**	**-0.436**	**-0.393**
SD	***0.358***	***0.234***	***0.264***	***0.238***	***0.170***

Table 5.4: Pearson correlation coefficient ρ between error and the RETRO-score for Experiment A for relevant models and all datasets. The greener the cells, the stronger the correlation. The mean ρ and its standard deviation for each model architecture is provided in the bottom row. [c3]Note that these results are identical to those in Table 5.1, but only the subset of results for Experiment A is shown.[c4]

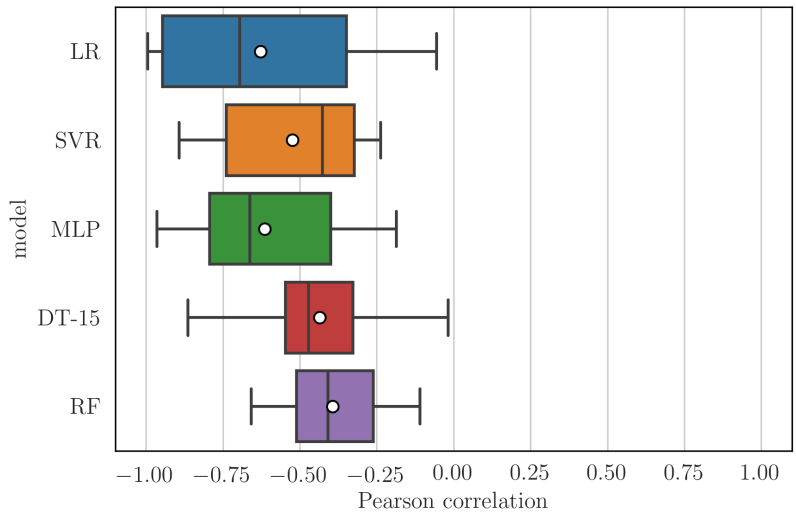


Figure 5.1: Visualization of the Pearson correlation distribution for models facing distributional shift. While the box plots show the full range of correlations across all 13 datasets, the white dot indicates the mean Pearson correlation coefficient ρ for each model type.

Results of Experiment B. Model overfit

The Pearson correlations found in Experiment B, which looks at the relationship between RETRO-scores and absolute error for predictions from overfit models, are given in Table 5.5 and summarized in Figure 5.2. We evaluate model overfit on a Gaussian Process and a very deep Decision Tree (depth 10,000). We confirm that we indeed achieve an overfit: the performance on the train data is extremely good, whereas the models perform much worse on the test data. For example, for the *Wine quality* dataset, the Root Mean Squared Error (RMSE) for the Gaussian Process is 0.001 on the train set and 5.975 on the test set. Similarly, for the *Diabetes* dataset, the RMSE for the Decision Tree is 0.000 on the train set and 0.525 on the test set. The performance of the trained models is provided in Appendix A.

The value of ρ is negative in all cases, as desired. We observe that the Gaussian Process Regressor (GP) performs much better than the Decision Tree with depth $10,000$ (DT-10k): the Gaussian Process Regressor ρ-correlation is $\mathbf{-0.899}$ on average, compared to a mean of $\mathbf{-0.430}$ for the Decision Tree. The overlap between the 50% of instances with the largest absolute error and the lowest RETRO-score is also higher for the GP than for the DT (**0.772** and **0.629** respectively). Again, we discuss reasons for the divergence in Section 5.2.

Dataset	GP	DT-10k
Cyclepower	†	-0.389
Airfoil	-0.999	-0.199
Store sales	-1.000	-0.534
Fish toxicity	-1.000	-0.414
Abalone	-0.999	-0.609
Autompg	-0.992	-0.542
California housing	-1.000	-0.475
Energy efficiency	-0.734	-0.225
Diabetes	-0.751	-0.255
Wine quality	-0.999	-0.448
Boston housing	-0.699	-0.433
Superconductor	-1.000	-0.463
Communities	-0.617	-0.604
Mean	**-0.899**	**-0.430**
SD	***0.150***	***0.134***

Table 5.5: Pearson correlation coefficient ρ between error and the RETRO-score for Experiment B for relevant models and all datasets. The greener the cells, the stronger the correlation. The mean ρ and its standard deviation for each model is provided in the bottom row. †: Model did not overfit, so result is not included. (Note that these results are identical to those in Table 5.1, but only the subset of results for Experiment B is shown.)

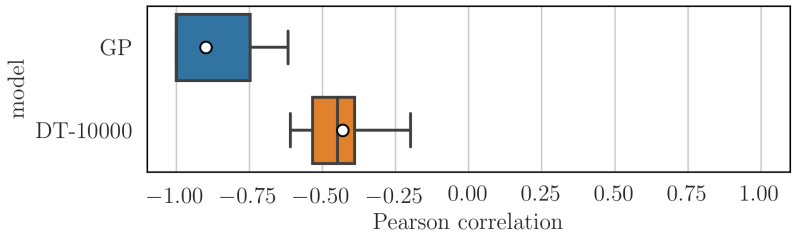


Figure 5.2: Visualization of the Pearson correlation distribution for overfit models. While the box plots show the full range of correlations across all 13 datasets, the white dot indicates the mean Pearson correlation for each model type.

Results of Experiment C. Model underfit

In Experiment C, we assess the performance of the RETRO-score for predictions resulting from underfit models. The Pearson correlations for this experiment are listed in Table 5.6, which shows the performance resulting from a shallow Multi-Layer Perceptron with a single layer of 10 neurons (MLP-SH) and a Decision Tree with depth 1 (DT-1). The results are summarized in Figure 5.3.

Again, ρ is negative in all cases, as desired. Like in the previous experiments, we observe that RETRO is less good at capturing erroneous predictions resulting from the Decision Tree (mean ρ of $\mathbf{-0.474}$) than from the Multi-Layer Perceptron (mean ρ of $\mathbf{-0.682}$). Similarly, the mean overlap between the 50% of instances with the largest absolute error and the lowest RETRO-score is **0.772** and **0.629** respectively.

Dataset	**MLP-SH**	**DT-1**
Cyclepower	-0.906	-0.741
Airfoil	-0.909	-0.393
Store sales	-0.780	-0.834
Fish toxicity	-0.791	-0.456
Abalone	-0.552	-0.212
Autompg	-0.914	-0.586
California housing	-0.271	-0.433
Energy efficiency	-0.940	-0.458
Diabetes	-0.460	-0.147
Wine quality	-0.311	-0.212
Boston housing	-0.687	-0.412
Superconductor	-0.719	-0.745
Communities	-0.620	-0.535
Mean	**-0.682**	**-0.474**
SD	***0.228***	***0.214***

Table 5.6: Pearson correlation coefficient ρ between error and the RETRO-score for Experiment C for relevant models and all datasets. The greener the cells, the stronger the correlation. The mean ρ and its standard deviation for each model is provided in the bottom row. (Note that these results are identical to those in Table 5.1, but only the subset of results for Experiment C is shown.)

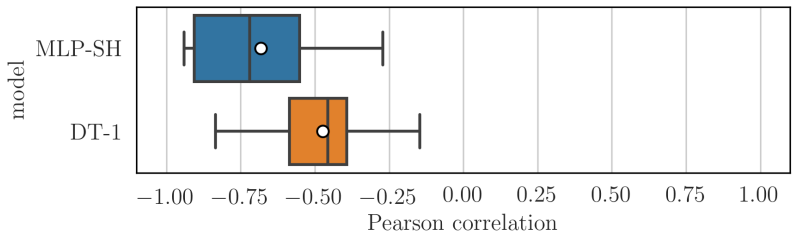


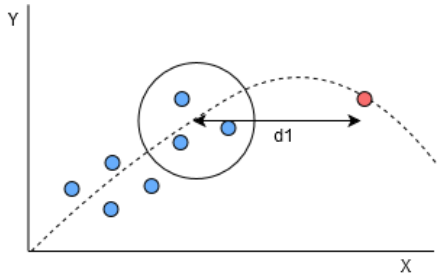
Figure 5.3: Visualization of the Pearson correlation distribution for underfit models. While the box plots show the full range of correlations across all 13 datasets, the white dot indicates the mean Pearson correlation for each model type.

5.1.2 Analysis: relative contribution of the two RETRO components

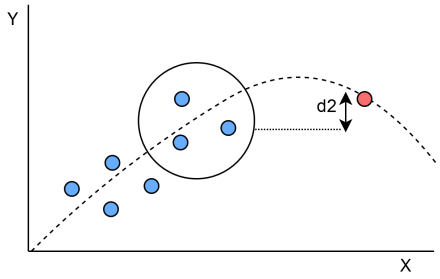
In this section, we provide additional analysis of the experimental results as described in the previous section. Specifically, we analyze why the RETRO-score correlates with errors of all three causes by looking at the architecture of RETRO. We consider that the score consists of two components. These are d_1, the distance between a new instance and its neighbors, and d_2, the distance between the predicted target value for the new instance and the ground-truth target value for its neighbors (see Section 3.1.2). We expect that both components respond to different causes of error in a different way, as we explain below. In addition, the two components and their expected behavior across error types are explained in Figure 5.4.

RETRO-component d_1: distributional shift When distributional shift occurs, new instances do not resemble the data the model was trained on. The prediction for a new instance is expected to be erroneous because the new instance lies far away from its 'closest' neighbors in the reference set. Therefore, we expect that component d_1 is most responsive to this type of error and will have a large magnitude.

RETRO-component d_2: model over- and underfit For model over- and underfit, the train and test distribution are alike. Retrieved neighbors from the reference set are expected to be reasonably close to the new instance, so component d_1 should be small. Instead, the issue is that the model is not able to make reasonable predictions for new instances, i.e. predictions that lie close to the ground-truth targets of the neighboring instances. As a result, we expect component d_2 to be of a larger magnitude.



Component d_1: the distance between an instance and its neighbors. Expected to be large for **distributional shift**.

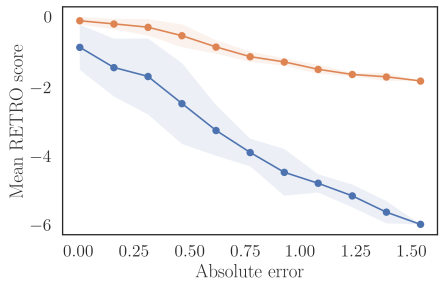


Component d_2: the distance between the predicted target value and the ground-truth target value for its neighbors. Expected to be large for model **over- and underfit**.

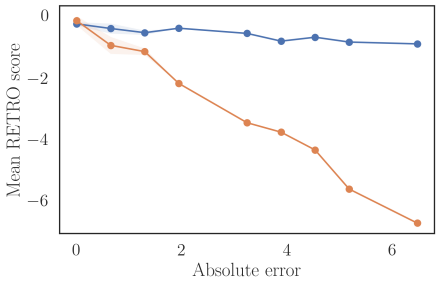
Figure 5.4: The two components of the RETRO-score. The blue dots represent the one-dimensional reference instances with input variable X and output variable Y. The red dot represents the new instance and the dotted line represents the fitted model. The dots inside the circle are the closest neighbors to the new instance. (Note that these figures are identical to Figure 3.4 and Figure 3.5 in Section 3.1.)

To assess whether the two RETRO-components indeed behave as expected, we plot the magnitude of both components under the different error causes in Figure 5.5. As an example, we look at the *Autompg* dataset, which we use to train an MLP for inducing distributional shift, a Gaussian Process to induce overfitting and a shallow MLP to induce underfitting, all with identical settings as compared to Experiments A, B and C. We chose the *Autompg* dataset and these models because they led to a good RETRO-performance: if the interaction between the two terms exists as hypothesized, it should be particularly clear under these settings. In settings where RETRO does not perform well, we expect that this interaction between the terms is also present less strongly.

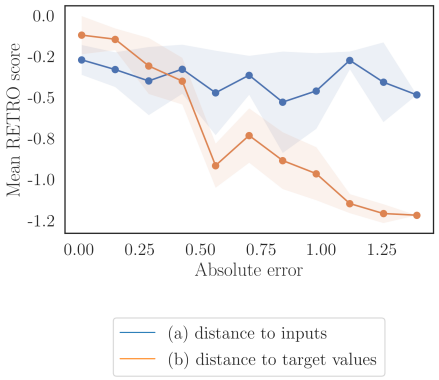
For distributional shift, we see that indeed, component d_1 or the blue line has a larger magnitude and correlates with absolute error better. In contrast, for model over- and underfit, we see that component d_2 or the orange line has a larger magnitude and correlates more strongly with error. These results were also observed across other datasets and model settings. This confirms that the components of RETRO interact as hypothesized in this section.



(a) **Distributional shift** (MLP). The blue line, or component d_1, has a larger magnitude: erroneous instances lie far away from their neighbors.



(b) **Overfit** (GP). The orange line, or component d_2, has a larger magnitude: predictions for erroneous points do not resemble the ground-truth target value for the neighbors.



(c) **Underfit** (MLP-SH). The orange line, or component d_2, has a larger magnitude. See Overfit.

Figure 5.5: The relative strength of the two components of the unnormalized RETRO-score across the different causes of error (Autompg dataset). The blue lines represent the average distance between a new instance and its neighbors (d_1). The orange lines represent the distance between the instance's predicted target value and the ground-truth target value of its neighbors (d_2). We look at the mean RETRO-score versus the absolute error in predictions (the shaded region represents one standard deviation).

5.2 RQ2: Performance across Model Architectures, Data Dimensionalities and Errors

As shown in Section 5.1, the RETRO-scores negatively correlate with error in algorithmic predictions, as desired. However, it is also clear that there is variation in the results. We aim to understand in which situations RETRO tends to correlate strongly with error and under which conditions it does not. In this section, we compare performance across model architectures, data dimensionalities as well as error causes and magnitudes, which allows us to answer **RQ2**. Note that this part of the evaluation uses the same experimental results that were obtained in Experiments A, B and C (as described in the previous section), but compares them explicitly along the specified dimensions. Where applicable, we also provide analysis to explain the observed divergence in performance.

5.2.1 Analyzing performance across model architectures

We first assess the performance of the RETRO-score across different model architectures. While RETRO can theoretically be applied to any regressor, we observe that in practice, the method works better for some model types than for others. In particular, we observe an inferior performance on tree-based models. In Figure 5.6, we summarize the results across model types. For the full results, we refer to Table 5.1.

We obtain the best performance for predictions generated by an overfitting Gaussian Process (mean ρ-correlation of $\mathbf{-0.899}$), the second-best performance for the underfit Multi-Layer Perceptron ($\mathbf{-0.682}$), followed by the non-tree distributional shift models (mean of $\mathbf{-0.627}$ for Linear Regression, $\mathbf{-0.524}$ for SVR and $\mathbf{-0.614}$ for Multi-Layer Perceptron). In contrast, across all causes of error, we observe that the method performs less well on predictions resulting from tree-based models than from other types of models. A closer inspection of the components of the RETRO-score and the way tree-based algorithms work reveals why this might be the case.

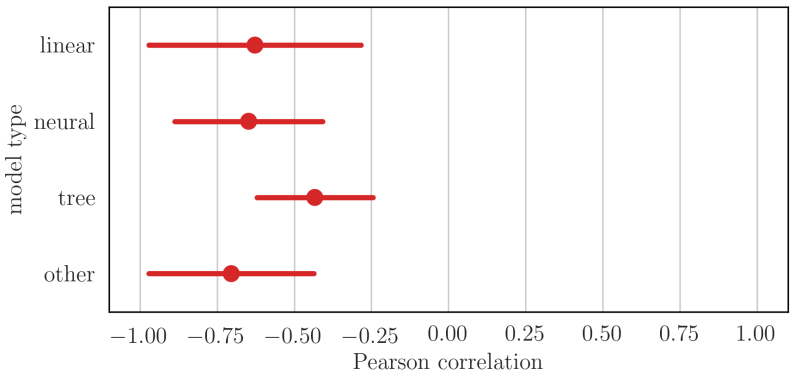


Figure 5.6: The average Pearson correlation coefficient ρ across Experiments A, B and C for each model type. The error bars equal one standard deviation from the mean. See Table 5.1 for the full results.

Tree-based regression models A tree-based regression algorithm divides the feature space into subspace, where each subspace receives a constant prediction (i.e. one output value is assigned to all inputs that fall into a subspace). This is visualized in Figure 5.7, where each rectangle represents a subspace. Splits are determined on the basis of the train set, where the splits are made such that the error on the train set is reduced. When the test data falls outside the range of the train data, the splits still happen at the same values as determined on the train set: the subspaces are 'expanded' to the range of the test data. In this way, the predictions on the test set are bounded by the target values in the train set. This means that predictions on the test set will never be far away from the ground-truth target values of the train set. For the RETRO-score, component d_2 which measures the distance between the predicted target value and the ground-truth target values of the neighbors is therefore not very predictive of error, as this distance will always be small. Instead, RETRO now mostly relies on its component d_1, which measures the distance between the nearest neighbors and the new instance (also see the end of Section 5.1).

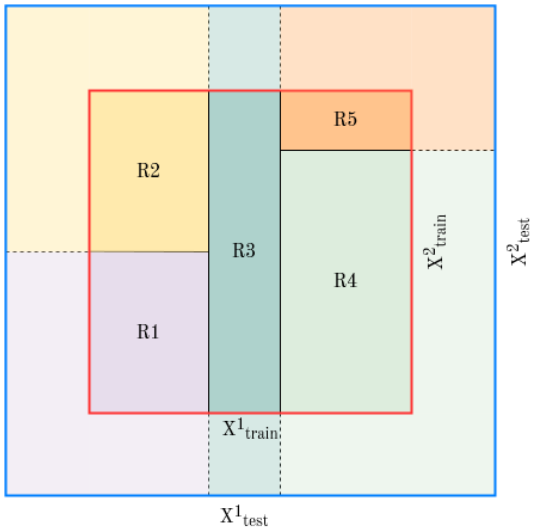


Figure 5.7: Visualization of the partition of a two-dimensional feature space by a Decision Tree algorithm. Training points all lie inside the red square, while the test feature space extends as wide as the blue square. Both train and test inputs receive one of five predicted target labels, i.e. $\hat{y} \in (R_1 ... R_5)$.

Other types of models We contrast the above with a Gaussian Process Regressor. As visualized in Figure 5.8, the Gaussian Process Regressor fits a predictive function to the training data.[1] In regions where there are many training examples, the predictive function fits the data closely. In contrast, in regions where no training data exists, the model may suddenly produce predictions that are far outside the range of the training data's targets. In other words, $\hat{Y}$ for data close to and far away from the training data do not necessarily overlap. Again, for tree-based models, this does not hold: $\hat{Y}$ always comes from a constrained set that is based on the training data. Returning to the RETRO-score, this means that the algorithm can optimally leverage both of its components when used with predictions resulting from Gaussian Process Regressors. Erroneous predictions are now likely to have a large distance to the ground-truth target values of the neighbors in the train set, which is what component d_2 responds to. Note that this also holds for neural architectures and linear regression models, which also produce predictions on the test data that are not bound to the range of the target values of the train set.

[1]Note that in reality, we fit a Gaussian Process kernel to the data and the predictive function represents the mean associated with that kernel. For details about Gaussian Processes, we refer to Bishop [2006] and to *scikit-learn* (`https://scikit-learn.org/stable/`) for implementation details.

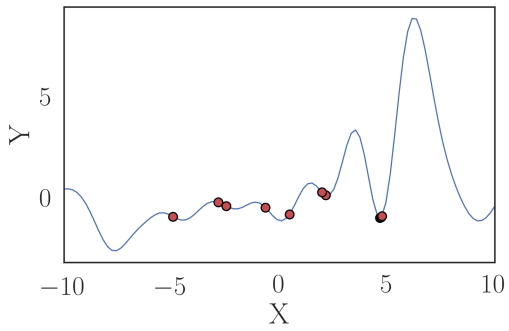


Figure 5.8: Visualization of a Gaussian Progress Regressor. The red dots represent the training data (with one input dimension X and target Y), whereas the blue line represents the fitted (mean) predictive function. In some cases (especially around $X = 6$), the blue line moves very far away from the training data, meaning that the model is likely to produce unreasonable predictions in these regions. The Gaussian Process Regressor is implemented according to the default *scikit-learn* settings and uses an RBF kernel.

5.2.2 Analyzing performance across data dimensionalities

In Figure 5.9, we visualize the results across datasets. The Pearson correlation coefficient ρ is averaged across all experiments (i.e. error causes) and models. For the full results, we refer to Table 5.1. These datasets have different dimensionalities: the number of independent features ranges from 4 to 100. We observe that there is little variation in the performance of RETRO across the dimensionalities.

The RETRO-algorithm relies on the k-Nearest-Neighbors (kNN) algorithm, which is known to perform poorly in high-dimensional spaces [Beyer et al., 1999]. Therefore, it may be expected that the performance of RETRO is worse on datasets with a larger number of features. However, recall from Section 4.3 that the largest datasets are reduced in dimensionality (*Superconductor* and *Communities* with 81 and 100 features respectively). Figure 5.9 shows that RETRO does not perform worse (or better) for these large and dimensionality-reduced datasets than for the other, smaller datasets. This indicates that the dimensionality reduction has helped to overcome the curse of dimensionality typically observed in kNN architectures.

While our experiments show a stable performance across dataset sizes for dimensionalities up to 100, we foresee that reducing the dimensionality can lead to a worse performance for much larger feature spaces. For larger dimensionalities, it is increasingly more difficult to represent the data in fewer features in a meaningful way: the larger the reduction in features, the larger the information loss. Note that the amount of information loss depends on the intrinsic dimensionality of the data: the more features correlate, the less information is lost during dimensionality reduction. In general, we expect a larger information loss to lead to a degraded performance of RETRO.

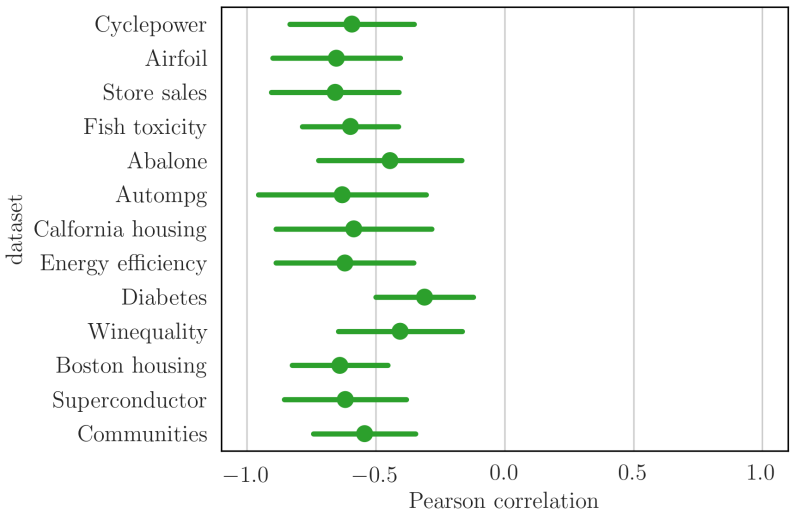


Figure 5.9: The average Pearson correlation ρ across Experiments A, B and C for each dataset. The datasets are ordered by the number of variables (in ascending order from top to bottom). The error bars equal one standard deviation from the mean. Full results can be found in Table 5.1.

5.2.3 Analyzing performance across error causes and severities

We evaluate performance across errors in two ways: we compare performance of RETRO across the of *cause* and the *magnitude* of the error.

Performance across causes of error We first compare the performance of RETRO across the causes of algorithmic error. We summarize the Pearson correlation ρ for each error cause in Figure 5.10, in which we average the correlation across all datasets and models. For the full results, we refer to Table 5.1. As can be observed, the performance of RETRO is similar across error causes. In Figure 5.5 in the previous section, we discussed how the two components of RETRO respond to different causes of error in different ways. The results in Figure 5.10 indicate that together, the RETRO components result in an algorithm that performs well across different error causes. We conclude that there does not seem to be a clear relationship between the cause of error and the performance of RETRO.

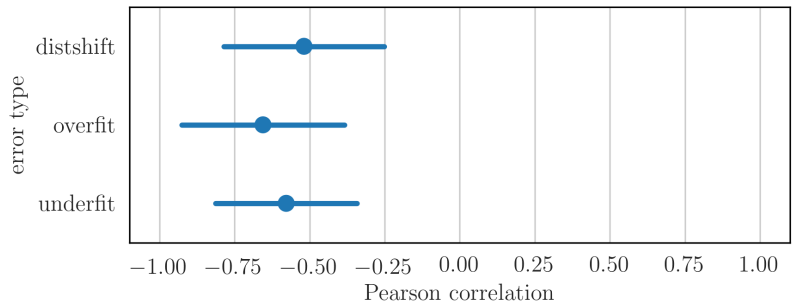


Figure 5.10: The mean Pearson correlation ρ across error causes. Results are averaged across datasets and model types. The error bars equal one standard deviation from the mean. Full results can be found in Table 5.1.

Performance depending on error magnitude We now analyze whether the performance of RETRO depends on the magnitude of the error in algorithmic predictions. To assess whether the performance of RETRO depends on the magnitude of the predictive error, we visualize results across models, datasets and error causes in Figure 5.11. Each dot in the figure represents a trained model for a particular dataset and model type. As such, there is one dot for each of the 117 experimental settings as shown in e.g. Table 5.1. Dots are grouped such that one window of the plot contains dots produced by one model type. We plot the Pearson correlation coefficient ρ between the error and the RETRO-score versus the Root Mean Squared Error (RMSE) as reported on the test set (see Appendix A). From this plot, we can draw two conclusions:

1. **Some model types induce a much larger error than others**. In particular, the Gaussian Process Regressor (GP) leads to very large errors (up to a log-RMSE of > 5 on one occasion). In contrast, the tree-based model architectures tend to result in the smallest error overall. As mentioned in Section 5.2.1, this divergence stems from the characteristics of these models.

2. **Larger errors lead to a better performance of RETRO.** We observe that a larger RMSE is linked to a larger Pearson correlation coefficient. In all windows, except that for tree-based models, we observe that dots lower on the plot (i.e. higher ρ) tend to lie to the right (i.e. larger log-RMSE). We conclude that RETRO performs better when models produce predictive errors with a large magnitude than when a model produces small errors.

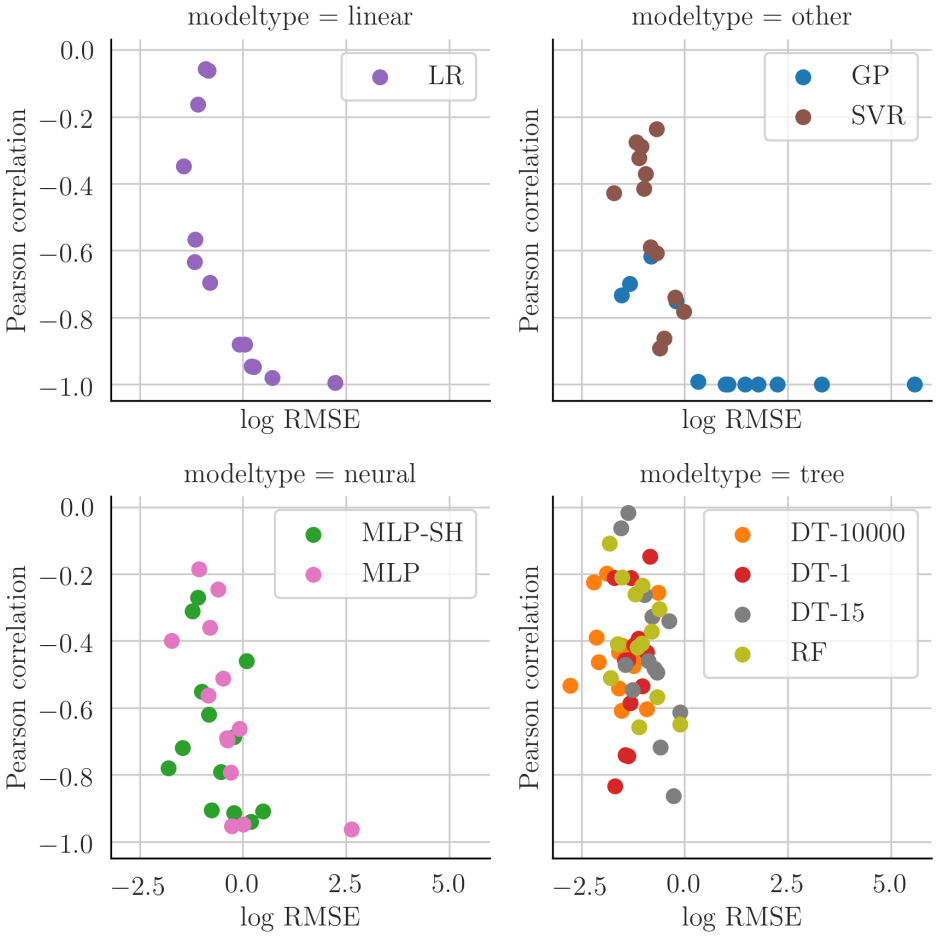


Figure 5.11: Pearson correlation coefficient for a particular model/dataset combination versus the logged Root Mean Squared Error (RMSE) as reported on the test set. Each dot represents a dataset, trained on the model type that its color indicates. Each window contains a subgroup of model types. We observe that (1) some models produce a much larger error than others; (2) RETRO tends to correlate with error more when the error is large.

Chapter 6

Evaluating the VIZ-plots

In this chapter, we describe the experimental design for evaluating the VIZ-plots, which is intended to answer **RQ3**. RETRO-VIZ consists of two parts: a numeric assessment of trustworthiness (RETRO) as well as a visual explanation for the estimated trustworthiness (VIZ). While the RETRO-scores are evaluated as described in Chapter 4 and Chapter 5, the visual explanations for the RETRO-scores, VIZ, can only be meaningfully evaluated with users. The goal of RETRO-VIZ is to improve algorithm-in-the-loop decision-making by providing users tools to recognize and understand potential error in predictions. Therefore, to assess the quality of the method, evaluation with real human users in a user study is essential.

Set-up of the user study The user study consists of two parts. Firstly, we aim to understand whether VIZ-plots help humans to recognize untrustworthy predictions *objectively*. Secondly, we assess whether users *subjectively* experience VIZ-plots as valuable - if this is not the case, the method will not be adopted by users and the theoretical benefits are not realized in practice.

The user study was completed by 41 participants, all of whom were analysts or data scientists working for one of the Ahold Delhaize brands. Participants were invited by email and completed a web-based user study in their own environment. The general set-up of the user study is as follows:

1. The participants are shown an introduction to RETRO-VIZ, in which both the RETRO-scores and VIZ-plots are explained to them.

2. An objective evaluation is performed to answer **RQ3a** and **RQ3b**. See Section 6.1 for a detailed description.

3. A subjective evaluation is performed to answer **RQ3c**. See Section 6.2 for a detailed description.

The full user study can be found in Appendix B.

6.1 RQ3a/b: Objective User Evaluation

Doshi-Velez and Kim [2017] argue that preferably, evaluation of AI explanations must either be application-grounded or human-grounded. Application-grounded evaluation means that the method is evaluated with domain experts on the goal task directly. For example, for Ahold Delhaize analysts, a goal task could be 'accurately forecasting sales'. The question application-grounded evaluations aim to answer is therefore not whether the explanations themselves are understood, but whether they help improve performance of the algorithm-in-the-loop system as a whole.

Human-grounded evaluation In human-grounded evaluation, the explanation method is instead tested on a somewhat simplified task. The advantage of this is that it is possible to control for other factors than the quality of the explanation alone, such as the difficulty of the task, which is not possible in application-grounded evaluation. Moreover, the simplified task can serve as a proxy for a multitude of tasks where RETRO-VIZ can be applied. Rather than a single potential use case, many settings exist within Ahold Delhaize in which having a better grasp on potentially erroneous predictions could be useful. Therefore, we design a human-grounded evaluation for the user study, where the task is derived from the day-to-day activities of the participants in our study.

The simplified task we design for the user study is the following: given a simple model to predict monthly sales for stores of one of the Ahold Delhaize brands, as well as several predictions generated by this simple model, participants are tasked with assessing the trustworthiness of the predictions. To do so, they can make use of VIZ explanations. Before beginning the task, the simple model as well as the RETRO-VIZ method are explained. In this explanation, it is stressed that the simple model is overall of a high quality (because it has a high r^2), but that it may still produce erroneous predictions sometimes.

Goals of the evaluation The objective evaluation aims to answer two questions: on the one hand, we want to know if the explanations allow the user to identify *whether* a prediction is trustworthy (**RQ3a**); on the other hand, we want to know if the user can identify *why* a particular prediction is trustworthy or not (**RQ3b**). Each is assessed through a separate task.

6.1.1 RQ3a: Recognizing trustworthiness of predictions

To assess whether users can distinguish trustworthy from untrustworthy predictions, we ask participants to complete three *pairwise comparisons*. In each, two VIZ-plots are shown side by side and the participant is asked to indicate which prediction she believes to be more trustworthy. The VIZ-plots are shown in pairs, because trustworthiness is not an objective, binary attribute: what meets the threshold for trustworthiness for one participant may not be trustworthy to someone else. By asking users which prediction is perceived as *more* trustworthy, we prevent having to account for these interpersonal differences [Saaty, 2008]. Note that the RETRO-scores are not

shown, so that the user has to assess trustworthiness based on the visual explanation alone. The correctness of answers is evaluated on the basis of the RETRO-scores of the two predictions.

6.1.2 RQ3b: Recognizing reasons for trustworthiness of predictions

Beyond helping users understand *whether* a prediction is trustworthy, the VIZ explanations are designed to show *why* a particular prediction is trustworthy or not. There are three main categories of VIZ-plots that can be distinguished:

1. For a trustworthy prediction, we expect both the neighbors of the new instance as well as their target values to *lie close* to the new instance and its predicted target. This corresponds to the situation in Figure 3.8.

2. For an untrustworthy prediction, it could be that (some of) the *input features lie outside the training data*, so that the 'closest' neighbors as shown on the VIZ-plot still lie far away. This corresponds to the situation in Figure 3.9.

3. A prediction is also untrustworthy if the *predicted value lies far away* from the target value of the closest neighbors. This corresponds to the situation in Figure 3.10.

To assess whether participants recognize these reasons for (un)trustworthiness on the basis of the VIZ-plots, we show them three single plots in succession, one for each of the categories. Participants are asked to identify whether they believe a prediction to be trustworthy or not and to provide a reason for their assessment. Recognizing the limitation of asking for trustworthiness on a single plot as alluded to above, we select examples with relatively extreme values: the predictions are either very untrustworthy or very trustworthy. The reason for the perceived (lack of) trustworthiness is an open-ended question: users are asked to fill out a text box with their response. We manually inspect users' answers to assess whether users correctly identified the reasons for (not) trusting a prediction. Answers are considered correct if they mention (1) close alignment between instances, (2) deviation in features and (3) deviation in targets respectively.

6.2 RQ3c: Subjective User Evaluation

Besides an objective evaluation, we want to understand whether users subjectively experience RETRO-VIZ as valuable (**RQ3c**). As Ribeiro et al. [2016] point out, this is essential: "if the users do not trust [a method], they will not use it" and any theoretical gains from implementing a method will not be realized in practice.

Hoffman et al. [2018] develop an Explanation Satisfaction Scale, which attempts to standardize existing approaches to subjective evaluation of XAI methods. The scale measures "the degree to which users feel that they understand [...] the process being explained to them". The questionnaire is based on an extensive literature review and has been tested for validity and internal consistency.

While a typical XAI explanation aims to provide insight into how a particular algorithmic prediction was generated, our explanations have a different goal. Instead, VIZ-plots explain whether and why a particular prediction is trustworthy or not. Therefore, we slightly adapt the questions from the Explanation Satisfaction Scale [Hoffman et al., 2018] so that they fit our context better.

The adapted Explanation Satisfaction Scale questions as they were posed to the participants in our user study are listed below. Answers are provided on a 5-point Likert scale with as answer options 'Strongly agree', 'Somewhat agree', 'Neutral', 'Somewhat disagree' and 'Strongly disagree'.

Adjusted Explanation Satisfaction Scale

Q1. The visualization shows me how **accurate** the prediction is.

Q2. The visualization lets me judge **when I should trust or not trust** the prediction.

Q3. From the visualization, I understand **why** the algorithmic prediction is trustworthy.

Q4. The visualization of whether the prediction should be trusted is **satisfying**: I understand what it is showing.

Q5. The visualization of whether the prediction should be trusted seems **complete**: it gives me all the information I need to assess the trustworthiness of a prediction.

Q6. I think the visualization of whether the prediction should be trusted could be **useful in business settings**.

Chapter 7

Results of the VIZ Evaluation

As discussed in Chapter 6, we evaluate the VIZ explanations with a user study. This allows us to answer **RQ3**.

7.1 RQ3a/b: Objective User Evaluation

In the first part of the user study, we aim to identify whether VIZ-plots help users identify whether (**RQ3a**) and why (**RQ3b**) algorithmic predictions are trustworthy. Table 7.1 shows the percentage of correct answers to each question. The pairwise selection questions are indicated with SB (*Select Best*) and the questions where participants had to provide a reason for (not) trusting a prediction are indicated with IR (*Identify Reason*). As can be observed, the SB-questions were answered correctly by almost all participants (95.1% on average). Similarly, all IR-questions were answered correctly by a majority of participants (75.6% on average). A χ^2 test was performed to compare the number of participants that answered each question correctly to the number of participants that did not. Results are provided in Table 7.1 and indicate that each question was answered correctly significantly more often than not ($p \leq 0.05$).

Analysis of objective evaluation The results show that participants can reason successfully about the VIZ-plots: each question was answered correctly by a significant majority of participants (at least 68.3%; $p \leq 0.05$). This shows that VIZ-plots successfully help users identify *whether* a prediction is trustworthy (SB1-3), which answers **RQ3a**. VIZ-plots also help users identify *why* a prediction is (un)trustworthy (IR1-3), which answers **RQ3b**.

We observe that more participants answered questions SB1-3 correctly than questions IR1-3. This is not unexpected, because answering IR1-3 requires more effort from the participant. Users were asked to write down a text explaining their reasons for (not) trusting the plot, which requires more effort than simply ticking a box (as in the preceding SB-questions). When the written answer

		Correct	χ^2
Select Best	***SB1***	95.1%	33.390**
	SB2	97.6%	37.098**
	SB3	92.7%	29.878**
Identify Reason	***IR1***	68.3%	5.488*
	IR2	82.9%	26.561**
	IR3	75.6%	10.756**

Table 7.1: Percentage of Correct responses to the objective questions. Questions SB1 to SB3 consisted of selecting the most trustworthy prediction out of a pair; questions IR1 to IR3 asked participants to identify the reason for their (lack of) trust in a single prediction. Significant differences are denoted using * ($p \leq 0.05$) and ** ($p \leq 0.01$).

did not refer to the relevant components (such as a deviation in features or a deviation in predicted values), the answer was marked as wrong. Due to the additional effort required to answer these questions, we expect a larger proportion of wrong answers in the first place, which might explain part of why a larger proportion of participants answered SB1-3 correctly as compared to IR1-3.

7.2 RQ3c: Subjective User Evaluation

After the objective evaluation, users are asked to indicate their opinion on the *Adjusted Explanation Satisfaction Scale*, which we adopt from Hoffman et al. [2018]. The questions in the scale are listed in Section 6.2. Answers are provided on a 5-point Likert scale, with as answer options 'Strongly agree', 'Somewhat agree', 'Neutral', 'Somewhat disagree' and 'Strongly disagree'. For our analysis, we aggregate answers into two categories, '[Strongly] Agree' and '[Strongly] Disagree'. Neutral answers are omitted from the analysis [Norman, 2010]. Table 7.2 summarizes the answers of participants to the Explanation Satisfaction Scale. Note that because neutral answers are omitted, the percentages do not add up to 100%. We perform a χ^2 test to assess whether the proportion of participants that evaluate VIZ positively on a particular dimension ('[Strongly] Agree') is larger than the proportion of participants evaluating the method negatively ('[Strongly] Disagree').

As can be observed, for five out of six questions, the proportion of participants that evaluate VIZ positively is significantly larger than the proportion that evaluates the method negatively ($p \leq 0.05$). We now discuss each of the questions in turn.

Q1 - The visualization shows me how accurate the prediction is The percentage of participants agreeing that VIZ-plots show them the accuracy of a prediction (63.4%) is significantly larger than the percentage of participants who do not agree with this (29.3%; $p \leq 0.05$). Therefore, we conclude that participants find VIZ-plots helpful to assess the accuracy of a prediction. On the one hand, this is justified: as shown in Chapter 4, RETRO-scores (and therefore the VIZ-

	(Strongly) Agree	(Strongly) Disagree	χ^2
Q1	63.4%	29.3%	5.16*
Q2	87.8%	7.3%	27.92**
Q3	78.0%	19.5%	14.40**
Q4	87.8%	4.9%	30.42**
Q5	34.1%	43.9%	0.50
Q6	87.8%	4.9%	30.42**

Table 7.2: Number of users who (strongly) Agree or (strongly) Disagree with each statement. As neutral answers are omitted, percentages do not add up to 100%. Significant differences are denoted using * ($p \leq 0.05$) and ** ($p \leq 0.01$). See Section 6.2 for the full questions.

explanations) generally correlate with accuracy. However, on the other hand, the correlation is not perfect and trustworthiness as measured by RETRO does not fully correspond to accuracy. This shows that it is important that users of RETRO-VIZ are made aware of this distinction.

Q2 - The visualization lets me judge when I should trust or not trust the prediction In Q2, a significant majority of participants (87.8%; $p \leq 0.01$) indicate that VIZ-plots help them identify whether they should trust or not trust a prediction. This confirms our findings for **RQ3a** and the objective evaluation.

Q3 - From the visualization, I understand why the algorithmic prediction is trustworthy A majority of 78.0% of participants indicate that they understand why a prediction is (un)trustworthy (Q3), which is again significant ($p \leq 0.01$). Again, this confirms our findings for **RQ3b** and the objective evaluation.

Q4 - The visualization of whether the prediction should be trusted is satisfying: I understand what it is showing Participants indicate that they find the VIZ-plots satisfying in that they understand what these plots are showing (87.8% of participants; $p \leq 0.01$). This is confirmed in the first part of the survey, where indeed each question was answered correctly by a significant majority of participants. However, it is surprising that this percentage is larger than the percentage of participants that answered question IR1-3 correctly (at most 82.9%). This could indicate overconfidence of the participants or could point to the large amount of effort required to answer questions IR1-3 as discussed in the previous section.

Q5 - The visualization of whether the prediction should be trusted seems complete: it gives me all the information I need to assess the trustworthiness of a prediction 43.9% of participants indicate that they believe the VIZ-plots are **not** complete: the plots do not provide all the information that is needed to assess the trustworthiness of a prediction. While the result is not significant, this is larger than the percentage of participants that indicate that

VIZ-plots *are* complete (34.1%). This is reasonable and it is also what we anticipated in the design of the plots (see Section 3.2). To determine whether deviations in either features or predictions are problematic and to decide what actions should be taken, users must rely on their domain expertise and may leverage other sources of information within their organization.

Q6 - I think the visualization of whether the prediction should be trusted could be useful in business settings A significant majority of participants indicate that they believe the VIZ-plots to be useful in business settings (87.8% of participants; $p \leq 0.01$). This indicates that VIZ-plots are indeed perceived as a valuable indicator of the trustworthiness of algorithmic predictions which facilitates further investigation of risky predictions by domain experts.

Chapter 8

Discussion and Conclusion

We introduced RETRO-VIZ, a method for identifying and explaining the trustworthiness of algorithmic predictions from regression models. RETRO-VIZ aims to fill an important gap in the XAI literature by focusing on explaining the trustworthiness of predictions for which the true error is unknown. On the one hand, existing interpretability methods focus on explaining which factors contributed to an algorithmic prediction, without awareness of whether a prediction is correct. On the other hand, current uncertainty estimation techniques provide a quantitative evaluation of the likelihood that an algorithmic prediction is correct, but do not explain why a prediction is (not) trustworthy. The goal of the method is to improve decision-making in algorithm-in-the-loop settings where humans are expected to prevent failure by correcting algorithmic mistakes.

In this chapter, we provide an overview of the main findings of this research. We also discuss the potential implications of this work, as well as limitations of the method and avenues for future research.

8.1 Answers to Research Questions

We provide an overview of the answers to the research questions as defined in Section 1.1. For a detailed description of the evaluation framework and the results, we refer to Chapters 4 to 7.

RQ1. Do the estimates of trustworthiness that RETRO produces correlate with the errors in algorithmic predictions, and if so, how? We assess the Pearson correlation coefficient ρ between the absolute error in algorithmic predictions and the RETRO-score. Across all 117 experimental settings, spanning various datasets, causes of error and model architectures, we find that ρ is negative, with a mean value of -0.561. This shows that RETRO-scores are capable of distinguishing erroneous predictions: on average, larger errors will result in lower RETRO-scores. In addition, we assess the overlap between the set of test instances with the 50% lowest RETRO-

scores and the 50% largest absolute errors. Ideally, these sets are identical: instances with a more severe error should receive a lower score. We find that on average, 73.1% of these sets overlap.

RQ2. Under which conditions does RETRO perform best, given different (a) model architectures, (b) data dimensionalities and (c) causes and magnitudes of error? We study the conditions under which RETRO performs best by comparing the Pearson correlation coefficient ρ across 117 experimental settings in total. Firstly, we find that RETRO performs worse for tree-based models than for other model architectures. Secondly, we find that performance of the method does not differ across the tested data dimensionalities (4 to 100). Lastly, we find that performance does not depend on the type of the predictive errors and that the method is better at capturing errors of large magnitude than at capturing small errors.

RQ3. To what degree do VIZ-plots *objectively* and *subjectively* provide insight into algorithmic performance in a way that aids human-algorithm co-operation? We answer this research question through a user study and specifically answer three subquestions as listed below. The user study was performed at Ahold Delhaize with 41 participants, all data scientists or passive users of algorithmically-generated predictions.

RQ3a. To what extent do VIZ-plots *objectively* help users distinguish trustworthy algorithmic predictions from untrustworthy predictions? To answer this question, we ask participants to select the most trustworthy prediction from a pair of VIZ-plots, for three pairs. We find that VIZ-plots help users to successfully distinguish trustworthy and untrustworthy algorithmic predictions, as each question was answered correctly by a significant majority of participants ($p \leq 0.05$).

RQ3b. To what extent do VIZ-plots *objectively* help users assess the reason for the (lack of) trustworthiness of a prediction? For three single VIZ-plots, we ask users to describe why they believe the prediction is (not) trustworthy. We find that VIZ-plots help users to successfully assess the reasons for the (un)trustworthiness of algorithmic predictions: again, each question was answered correctly by a significant majority of participants ($p \leq 0.05$).

RQ3c. To what extent do users *subjectively* experience VIZ-plots as valuable for assessing the trustworthiness of algorithmic predictions? We assess this question through the *Adjusted Explanation Satisfaction Scale*, partially adopted from Hoffman et al. [2018]. We find that participants in the user study experience VIZ-plots as a helpful tool to discover whether algorithmic predictions are trustworthy and that they believe VIZ-plots could be valuable in their work. However, participants also indicate that VIZ-plots alone are not sufficient for decision-making and that they require further tools and analysis to make a final judgment about whether to trust or adapt a prediction.

8.2 Impact of this Research

In this section, we discuss the impact of the present research. We consider the ways in which the contribution of RETRO-VIZ could positively and negatively affect algorithmic decision-making. In particular, this section considers RETRO-VIZ from a user perspective: we aim to take the lens of the algorithmic decision-maker and that of the subject of such decisions rather than the academic viewpoint of the AI researcher.

8.2.1 Potential benefits

There are several operational benefits from RETRO-VIZ. Here, we highlight three of those benefits. Note that these show much overlap with the motivations for the development of explainable AI methods, which we described in Section 2.2.1.

Better end-to-end decision-making The main motivation for this research is to improve decision-making in algorithm-in-the-loop systems. In such systems, humans serve as a 'safety net' to prevent large algorithmic errors from being made [Elish, 2019; Green and Chen, 2019b], such that the system as a whole can perform better than when either operates alone [Kamar, 2016]. RETRO-VIZ equips humans inside such systems with a tool to recognize when algorithmic errors occur, which could lead to better end-to-end decision-making.

Acceptance of complex methods by decision-makers In Section 1.2, we describe the experience of analysts at Ahold Delhaize, who expressed a hesitation to implement complex machine learning methods due to the difficulty of interpreting such models. They specifically expressed a concern that it would be more difficult to detect when such complex methods fail. As RETRO-VIZ provides insight into the potential failure of complex models and offers a way to mitigate the risks of such potential failure, it may help to make such models more acceptable to organizational decision-makers.

Acceptance of complex methods by the subjects of algorithmic decisions Studies have found that people are often not comfortable with being the subject of automated decisions that they do not understand [Binns et al., 2018; Eslami et al., 2015]. In addition, the GDPR, an European Union-wide regulatory framework, requires that affected individuals are provided 'meaningful information about the logic' behind algorithmic decisions that are applied to them [Wachter et al., 2017a]. RETRO-VIZ could help to provide insight into algorithmic decisions by showing users that a particular prediction is grounded in the training data and in this way could help to make such predictions more acceptable. However, note that this assumes that VIZ-plots are understood and evaluated positively by non-experts, which we have not assessed in this work.

8.2.2 Potential risks

We highlight two ways in which RETRO-VIZ could have a negative impact on algorithmic decision-making, viewed from the perspective of algorithmic decision-makers and subjects of such decisions.

Misguided trust While RETRO-scores correlate with error to some extent, the correlation is not perfect. As a result, predictions that contain a large error could mistakenly receive a high RETRO-score. In this way, RETRO-VIZ could contribute to increasing the trust in a prediction when this is in fact misguided. This concern has been raised in previous work by Lakkaraju and Bastani [2020] in the context of a feature attribution method, but equally applies to RETRO-VIZ. To overcome this, users must be made aware of the limitations of RETRO-VIZ and of the fact that RETRO-scores do not correlate with error perfectly, but must be seen as an indication of the trustworthiness of a prediction. Especially in safety-critical applications, RETRO-scores are not sufficient as a sole mechanism to prevent large errors from being made.

Human bias through feature exposure Explanations are often framed as a tool to prevent bias in algorithmic decisions, as humans can recognize when such bias occurs [Gilpin et al., 2018]. However, while explanations may be used to reduce *algorithmic* bias, they also offer a venue for *human* bias to be introduced. In particular, by explicitly exposing feature values to humans and inviting humans to alter the algorithmic prediction based on their 'domain knowledge', RETRO-VIZ could increase the extent to which decisions are influenced by human bias. Green and Chen [2019b] find that in algorithm-in-the-loop settings, humans indeed make biased alterations to algorithmic decisions when they can view the feature values of an instance and it could well be the case that RETRO-VIZ induces similar behavior.

8.3 Limitations

In this section, we discuss the limitations of RETRO-VIZ from a technical perspective.

Training data availability RETRO-VIZ requires that the training data is (1) available and (2) can be shown to users in VIZ-plots. There are numerous settings in which this is not the case, for example when there are privacy concerns. In such situations, RETRO-VIZ does not provide a suitable solution.

Non-quantitative features RETRO-VIZ requires that all features used are quantitative, which is not always the case. It might be relatively straightforward to convert some features to a quantitative equivalent. For example, features with multiple levels might be turned into a set of binary features (indicating whether a particular level applies [1] or not [0]) when there are relatively few

levels. However, this is not always possible and the best solution in this situation might be to exclude such variables from the RETRO-VIZ procedure.

Visualizing large numbers of features VIZ-plots aim to provide insight into the potential deviation of a new instance versus the most similar points in the training data. This is done by visualizing the features on a set of parallel axes, which is problematic when the number of features becomes very large. Although some solutions have been proposed to reduce the number of features (see Section 3.2.2), the issue remains that not showing some features may lead to a reduced usefulness of the VIZ-plots for high-dimensional data.

Computational complexity The calculation of the RETRO-score relies on calculating the kNN-distance between the new instance and all instances in the train set. This is necessary to select those training instances that are the closest to the new instance. When the train set is very large, this is a computationally expensive step. As described in Chapter 3, a potential solution is to take a random sample from the train set and to find the closest neighbors within this sample. However, this is still a limitation: information is inevitably lost in the sampling process and the true closest neighbors may or may not be included in the sample.

8.4 Future Directions

In this section, we discuss potential avenues for extending the present research.

Better parameter search and understanding In our experiments, we used parameter settings that we generally found to work sufficiently well. However, we did not perform an extensive search of the best possible settings. On the one hand, such a search could improve the performance of the method. On the other, it could lead to an increased understanding of how the optimal parameters depend on particular experimental settings (datasets and models) and how parameters interact with each other. For example, it is a known issue of kNN algorithms that there is no general procedure for selecting an appropriate value of K and that the optimal value is highly dataset-dependent [Hastie et al., 2009]. In addition, it is not clear how the distance between instances (d_1) and the distance between predicted target values (d_2) should be balanced, which can be controlled through parameter β. A parameter search could provide more insight in this regard.

Improving performance for tree-based models We found that RETRO-VIZ performs well on a large set of models, but that the correlation of RETRO with error is worse for tree-based models such as Decision Trees and Random Forests. This could be explained by the properties of tree-based models: they exclusively predict from a set of values that is based on the training data, such that component d_2 of the RETRO-score is always small, in contrast to other models, whose

predictions are not constrained in this way and can take more extreme values (see Section 5.2.1). Future research could investigate how to make RETRO-VIZ more applicable to tree-based models.

Extension to time series In discussions with Ahold Delhaize colleagues, we found that many of the problems relevant in the retail context are time series regression problems. RETRO-VIZ is not specifically equipped to capture temporal aspects, so a useful next step would be to extend the method in this direction.

End-to-end, application-grounded evaluation While our user study has confirmed that RETRO-VIZ helps to identify the trustworthiness of algorithmic predictions and the reasons for (a lack of) trustworthiness, we have not explicitly evaluated whether our method leads to an increased end-to-end system performance. This type of evaluation is characterized by Doshi-Velez and Kim [2017] as 'application-grounded evaluation'.

Usefulness of VIZ-plots versus RETRO-scores In the user study, we explain the existence and the functioning of the RETRO-score to the participants, but we evaluate whether they can recognize (the reasons for) trustworthiness of predictions on the basis of the VIZ-plots alone. Interestingly, the VIZ-plots alone were sufficient for most users to answer the questions correctly. This raises the question to what extent the RETRO-scores provide an additional value. Moreover, future research could investigate in which settings RETRO-scores are useful, when VIZ-plots should be used in isolation and in which scenarios they should both play a role. For example, perhaps RETRO-scores could be used to automatically alert users when a prediction seems untrustworthy and a VIZ-plot can be shown when this happens.

8.5 Open-Source Availability and Reproducibility

Where relevant, we have pointed to the availability of source code for RETRO-VIZ throughout this thesis. To facilitate exploration of these resources, we provide an overview here.

- Code to reproduce the experiments performed in this study is available at `https://www.github.com/kimdebie/retroviz-tutorial`.
- RETRO-VIZ is available as a Python package, `retroviz`. Details can be found at `https://www.pypi.org/project/retroviz/`.
- A live VIZ-plot is available at `https://www.kimdebie.nl/pages/retroviz.html`.

Bibliography

A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, 7(1):39–59, 1994.

D. Alvarez-Melis and T. Jaakkola. Towards robust interpretability with self-explaining neural networks. In *Advances in Neural Information Processing Systems*, pages 7775–7784, 2018.

D. Alvarez-Melis, H. Daumé III, J. W. Vaughan, and H. Wallach. Weight of evidence as a basis for human-oriented explanations. *arXiv preprint arXiv:1910.13503*, 2019.

D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

J. O. Awoyemi, A. O. Adetunmbi, and S. A. Oluwadare. Credit card fraud detection using machine learning techniques: A comparative analysis. In *2017 International Conference on Computing Networking and Informatics (ICCNI)*, pages 1–9. IEEE, 2017.

G. Bansal, B. Nushi, E. Kamar, W. S. Lasecki, D. S. Weld, and E. Horvitz. Beyond accuracy: The role of mental models in human-ai team performance. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 7(1), pages 2–11, 2019a.

G. Bansal, B. Nushi, E. Kamar, D. S. Weld, W. S. Lasecki, and E. Horvitz. Updates in human-ai teams: Understanding and addressing the performance/compatibility tradeoff. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 2429–2437, 2019b.

K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is "nearest neighbor" meaningful? In *International conference on database theory*, pages 217–235. Springer, 1999.

U. Bhatt, A. Xiang, S. Sharma, A. Weller, A. Taly, Y. Jia, J. Ghosh, R. Puri, J. M. Moura, and P. Eckersley. Explainable machine learning in deployment. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 648–657, 2020.

R. Binns, M. Van Kleek, M. Veale, U. Lyngs, J. Zhao, and N. Shadbolt. 'it's reducing a human being to a percentage' perceptions of justice in algorithmic decisions. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2018.

O. Biran and C. Cotton. Explanation and justification in machine learning: A survey. In *IJCAI-17 workshop on explainable AI (XAI)*, volume 8(1), 2017.

C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

J. Q. Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. *Dataset shift in machine learning*. MIT Press, 2009.

B. J. Dietvorst, J. P. Simmons, and C. Massey. Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General*, 144(1):114, 2015.

B. J. Dietvorst, J. P. Simmons, and C. Massey. Overcoming algorithm aversion: People will use imperfect algorithms if they can (even slightly) modify them. *Management Science*, 64(3): 1155–1170, 2018.

F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

F. Doshi-Velez, R. Budish, and M. Kortz. The role of explanation in algorithmic trust. Technical report, Technical report, Artificial Intelligence and Interpretability Working Group ..., 2017.

D. Dua and C. Graff. UCI machine learning repository, 2020. URL `http://archive.ics.uci.edu/ml`.

M. C. Elish. Moral crumple zones: Cautionary tales in human-robot interaction. *Engaging Science, Technology, and Society*, 5:40–60, 2019.

A. Elmalech, D. Sarne, A. Rosenfeld, and E. S. Erez. When suboptimal rules. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

M. Eslami, A. Rickman, K. Vaccaro, A. Aleyasen, A. Vuong, K. Karahalios, K. Hamilton, and C. Sandvig. "i always assumed that i wasn't really that close to [her]." reasoning about invisible algorithms in news feeds. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*, pages 153–162, 2015.

FAA. *Advanced Avionics Handbook*. Federal Aviation Authority, 2017. URL `https://www.faa.gov/regulations_policies/handbooks_manuals/aviation/advanced_avionics_handbook/`.

Y. Gal. Uncertainty in deep learning. *University of Cambridge*, 2016.

Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE, 2018.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

D. Granberg and T. A. Brown. The monty hall dilemma. *Personality and social psychology bulletin*, 1995.

R. M. Grath, L. Costabello, C. L. Van, P. Sweeney, F. Kamiab, Z. Shen, and F. Lecue. Interpretable credit application predictions with counterfactual explanations. *arXiv preprint arXiv:1811.05245*, 2018.

B. Green and Y. Chen. Disparate interactions: An algorithm-in-the-loop analysis of fairness in risk assessments. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 90–99, 2019a.

B. Green and Y. Chen. The principles and limits of algorithm-in-the-loop decision making. *Proceedings of the ACM on Human-Computer Interaction*, 3(CSCW):1–24, 2019b.

R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1321–1330. JMLR. org, 2017.

W. Guo, S. Huang, Y. Tao, X. Xing, and L. Lin. Explaining deep learning models–a bayesian nonparametric approach. In *Advances in Neural Information Processing Systems*, pages 4514–4524, 2018.

T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

R. R. Hoffman, S. T. Mueller, G. Klein, and J. Litman. Metrics for explainable ai: Challenges and prospects. *arXiv preprint arXiv:1812.04608*, 2018.

A. Inselberg. The plane with parallel coordinates. *The visual computer*, 1(2):69–91, 1985.

H. Jiang, B. Kim, M. Guan, and M. Gupta. To trust or not to trust a classifier. In *Advances in neural information processing systems*, pages 5541–5552, 2018.

E. Kamar. Directions in hybrid intelligence: Complementing ai systems with human intelligence. In *IJCAI*, pages 4070–4073, 2016.

H. Kaur, H. Nori, S. Jenkins, R. Caruana, H. Wallach, and J. Wortman Vaughan. Interpreting interpretability: Understanding data scientists' use of interpretability tools for machine learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2020.

A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017.

E. M. Kenny and M. T. Keane. Twin-systems to explain artificial neural networks using case-based reasoning: comparative tests of feature-weighting methods in ann-cbr twins for xai. In *Proceedings of the International Joint Conference on Artifical Intelligence*, pages 2708–2715, 2019.

B. Kim, R. Khanna, and O. O. Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in neural information processing systems*, pages 2280–2288, 2016.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

R. Kocielnik, S. Amershi, and P. N. Bennett. Will you accept an imperfect ai? exploring designs for adjusting end-user expectations of ai systems. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2019.

V. Lai and C. Tan. On human predictions with explanations and predictions of machine learning models: A case study on deception detection. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 29–38, 2019.

V. Lai, S. Carton, and C. Tan. Harnessing explanations to bridge ai and humans. *arXiv preprint arXiv:2003.07370*, 2020.

H. Lakkaraju and O. Bastani. " how do i fool you?" manipulating user trust via misleading black box explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 79–85, 2020.

B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pages 6402–6413, 2017.

J. D. Lee and K. A. See. Trust in automation: Designing for appropriate reliance. *Human factors*, 46(1):50–80, 2004.

M. K. Lee. Understanding perception of algorithmic decisions: Fairness, trust, and emotion in response to algorithmic management. *Big Data & Society*, 5(1):2053951718756684, 2018.

O. Li, H. Liu, C. Chen, and C. Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

Z. C. Lipton. The mythos of model interpretability. *Queue*, 16(3):31–57, 2018.

A. Lucic, H. Haned, and M. de Rijke. Why does my model fail? contrastive local explanations for retail forecasting. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 90–98, 2020.

S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774, 2017.

S. Massie, S. Craw, and N. Wiratunga. Visualisation of case-base reasoning for explanation. In *Proceedings of the ECCBR*, pages 135–144, 2004.

S. M. McKinney, M. Sieniek, V. Godbole, J. Godwin, N. Antropova, H. Ashrafian, T. Back, M. Chesus, G. C. Corrado, A. Darzi, et al. International evaluation of an ai system for breast cancer screening. *Nature*, 577(7788):89–94, 2020.

T. Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

T. Miller, P. Howe, and L. Sonenberg. Explainable ai: Beware of inmates running the asylum or: How i learnt to stop worrying and love the social and behavioural sciences. *arXiv preprint arXiv:1712.00547*, 2017.

B. Mittelstadt, C. Russell, and S. Wachter. Explaining explanations in ai. In *Proceedings of the conference on fairness, accountability, and transparency*, pages 279–288, 2019.

M. Narayanan, E. Chen, J. He, B. Kim, S. Gershman, and F. Doshi-Velez. How do humans understand explanations from machine learning systems? an evaluation of the human-interpretability of explanation. *arXiv preprint arXiv:1802.00682*, 2018.

G. Norman. Likert scales, levels of measurement and the "laws" of statistics. *Advances in health sciences education*, 15(5):625–632, 2010.

M. Nourani, S. Kabir, S. Mohseni, and E. D. Ragan. The effects of meaningful and meaningless explanations on trust and perceived system accuracy in intelligent systems. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 7(1), pages 97–105, 2019.

B. Nushi, E. Kamar, and E. Horvitz. Towards accountable ai: Hybrid human-machine analyses for characterizing system failure. In *Sixth AAAI Conference on Human Computation and Crowdsourcing*, 2018.

N. Papernot and P. McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018.

F. Poursabzi-Sangdeh, D. G. Goldstein, J. M. Hofman, J. W. Vaughan, and H. Wallach. Manipulating and measuring model interpretability. *arXiv preprint arXiv:1802.07810*, 2018.

V. Rajendran and W. LeVine. Accurate layerwise interpretable competence estimation. In *Advances in Neural Information Processing Systems*, pages 13981–13991, 2019.

M. T. Ribeiro, S. Singh, and C. Guestrin. "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

C. Ross and I. Swetlitz. Ibm's watson supercomputer recommended 'unsafe and incorrect' cancer treatments, internal documents show. *Statnews*, Jul 2018. URL `https://www.statnews.com/2018/07/25/ibm-watson-recommended-unsafe-incorrect-treatments/`.

C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

T. L. Saaty. Relative measurement and its generalization in decision making why pairwise comparisons are central in mathematics for the measurement of intangible factors the analytic hierarchy/network process. *RACSAM-Revista de la Real Academia de Ciencias Exactas, Fisicas y Naturales. Serie A. Matematicas*, 102(2):251–318, 2008.

J. Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3145–3153. JMLR. org, 2017.

A. Springer, V. Hollis, and S. Whittaker. Dice in the black box: User experiences with an inscrutable algorithm. In *2017 AAAI Spring Symposium Series*, 2017.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

S. Tan, J. Adebayo, K. Inkpen, and E. Kamar. Investigating human + machine complementarity for recidivism predictions. *arXiv preprint arXiv:1808.09123*, 2018a.

S. Tan, R. Caruana, G. Hooker, P. Koch, and A. Gordo. Learning global additive explanations for neural nets using model distillation. *arXiv preprint arXiv:1801.08640*, 2018b.

G. Tolomei, F. Silvestri, A. Haines, and M. Lalmas. Interpretable predictions of tree-based ensembles via actionable feature tweaking. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 465–474, 2017.

E. Toreini, M. Aitken, K. Coopamootoo, K. Elliott, C. G. Zelaya, and A. van Moorsel. The relationship between trust in ai and trustworthy machine learning technologies. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, page 272–283, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450369367. doi: 10.1145/3351095.3372834. URL `https://doi.org/10.1145/3351095.3372834`.

N. Virani, N. Iyer, and Z. Yang. Justification-based reliability in machine learning. *arXiv preprint arXiv:1911.07391*, 2019.

S. Wachter, B. Mittelstadt, and L. Floridi. Why a right to explanation of automated decision-making does not exist in the general data protection regulation. *International Data Privacy Law*, 7(2):76–99, 2017a.

S. Wachter, B. Mittelstadt, and C. Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017b.

J. Wortman Vaughan and H. Wallach. The inescapability of uncertainty: Ai, uncertainty, and why you should vote no matter what predictions say, 2016. URL `https://society.net/uncertainty-edd5caf8981b`.

M. Yin, J. Wortman Vaughan, and H. Wallach. Understanding the effect of accuracy on trust in machine learning models. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2019.

Appendices

Appendix A

Model fit

In this Appendix, we present the performance of the trained models as measured by the Root Mean Squared Error (RMSE) on the train and test sets.

A.1 Distributional Shift

For distributional shift, we expect good performance on the train set, and a poor performance on the test set.

	LR		*SVR*		*MLP*		*DT-15*		*RF*	
	Train	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
Cycle power	0.120	1.292	0.109	0.387	0.103	0.680	0.033	0.455	0.032	0.449
Airfoil	0.251	1.051	0.196	0.792	0.190	0.687	0.006	0.511	0.036	0.512
Store sales	0.170	0.429	0.149	0.988	0.135	0.914	0.072	0.896	0.032	0.899
Fish toxicity	0.199	0.451	0.187	0.437	0.180	0.432	0.037	0.414	0.076	0.318
Abalone	0.161	0.310	0.156	0.180	0.152	0.177	0.032	0.215	0.059	0.162
Autompg	0.226	0.313	0.194	0.548	0.175	0.759	0.141	0.255	0.083	0.219
Cal. housing	0.296	9.305	0.283	0.504	0.243	13.870	0.093	0.378	0.078	0.300
Energy eff.	0.171	2.042	0.153	0.375	0.068	1.004	0.000	0.237	0.036	0.198
Diabetes	0.326	0.405	0.313	0.505	0.226	0.545	0.002	0.686	0.138	0.539
Winequality	0.250	0.337	0.240	0.351	0.209	0.613	0.092	0.479	0.073	0.359
Boston housing	0.208	0.239	0.161	0.609	0.133	0.745	0.009	0.764	0.059	0.330
Superconductor	0.189	1.226	0.166	0.310	0.136	0.447	0.067	0.284	0.055	0.167
Communities	0.255	0.913	0.248	0.331	0.056	0.346	0.038	0.552	0.102	0.352

Table A.1: Root Mean Squared Error (on the normalized features) on the train and test set for the distributional shift models - Linear Regression (LR), Support Vector Regressor (SVR), Multi-Layer Perceptron (MLP), Decision Tree with depth 15 (DT-15) and a Random Forest (RF).

A.2 Overfit

For overfit models, we expect a near-zero error on the train set, and a poor performance on the test set.

	GP		*DT-10000*	
	Train	**Test**	**Train**	**Test**
Cycle power	0.093	0.106	0.000	0.118
Airfoil	0.053	4.355	0.000	0.152
Store sales	0.053	2.861	0.000	0.062
Fish toxicity	0.043	9.475	0.320	0.220
Abalone	0.107	2.681	0.000	0.217
Autompg	0.000	1.394	0.000	0.202
Cal. housing	0.146	262.248	0.000	0.292
Energy eff.	0.000	0.216	0.000	0.110
Diabetes	0.000	0.811	0.000	0.525
Winequality	0.001	5.975	0.000	0.285
Boston housing	0.000	0.266	0.000	0.203
Superconductor	0.040	27.870	0.045	0.124
Communities	0.000	0.443	0.000	0.401

Table A.2: Root Mean Squared Error (on the normalized features) on the train and test set for the overfit models - the Gaussian Process (GP) and the Decision Tree with depth 10,000 (DT-10000).

A.3 Underfit

For underfit models, we expect a poor performance on both the train and test set.

	MLP-SH		*DT-1*	
	Train	**Test**	**Train**	**Test**
Cycle power	0.466	0.468	0.238	0.238
Airfoil	1.587	1.632	0.338	0.325
Store sales	0.163	0.164	0.180	0.184
Fish toxicity	0.582	0.582	0.254	0.255
Abalone	0.374	0.367	0.198	0.182
Autompg	0.854	0.807	0.269	0.267
Cal. housing	0.330	0.336	0.394	0.399
Energy eff.	1.180	1.207	0.226	0.234
Diabetes	1.022	1.084	0.400	0.431
Winequality	0.287	0.293	0.269	0.272
Boston housing	0.850	0.807	0.305	0.296
Superconductor	0.232	0.230	0.253	0.254
Communities	0.443	0.436	0.356	0.357

Table A.3: Root Mean Squared Error (on the normalized features) on the train and test set for the underfit models - the shallow MLP (MLP-SH) and the Decision Tree with depth 1 (DT-1).

Appendix B

User Study

This Appendix contains the full user study as it has been distributed to data scientists within Ahold Delhaize.

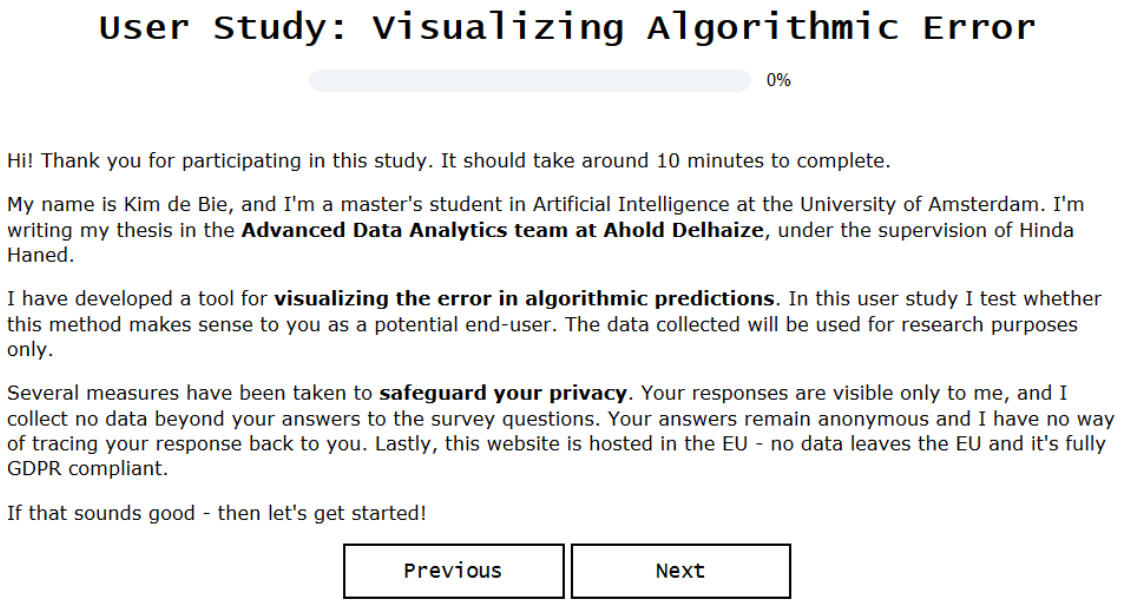


Figure B.1: Introduction.

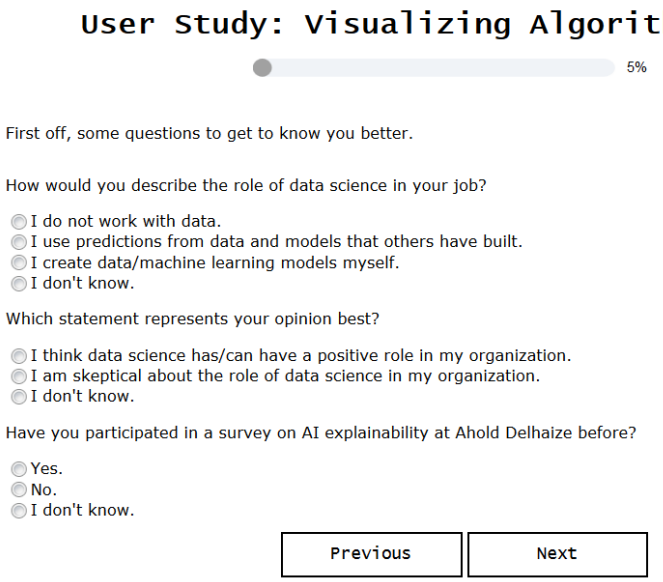


Figure B.2: Understanding participant characteristics.

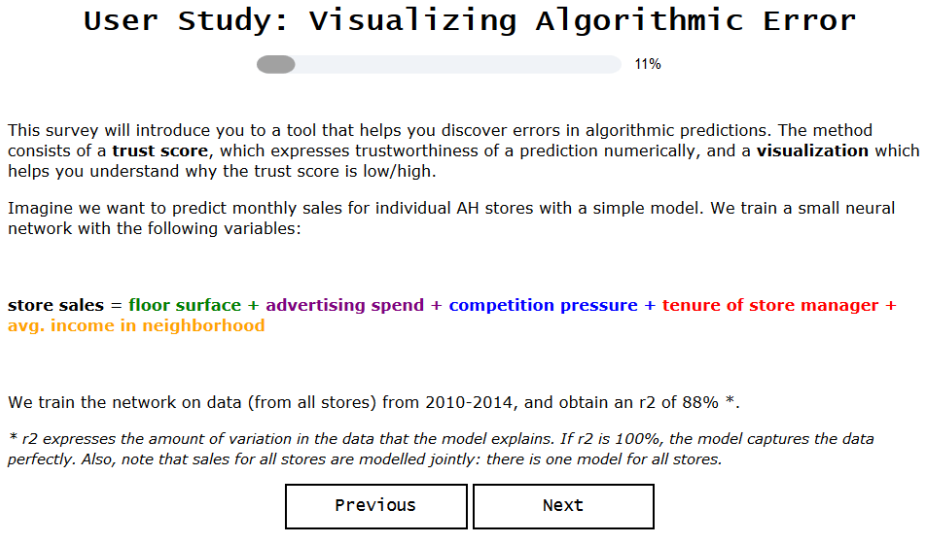


Figure B.3: Introduction to the method, part 1.

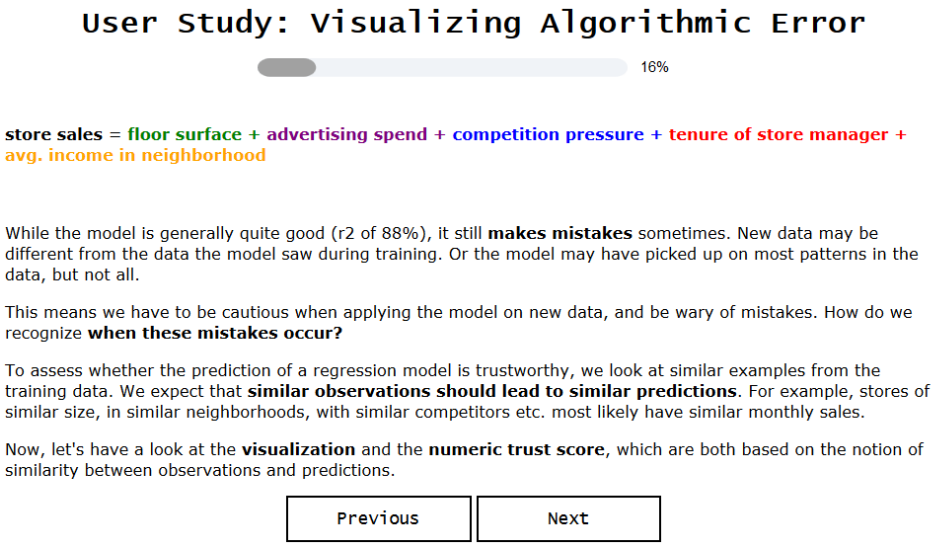


Figure B.4: Introduction to the method, part 2.

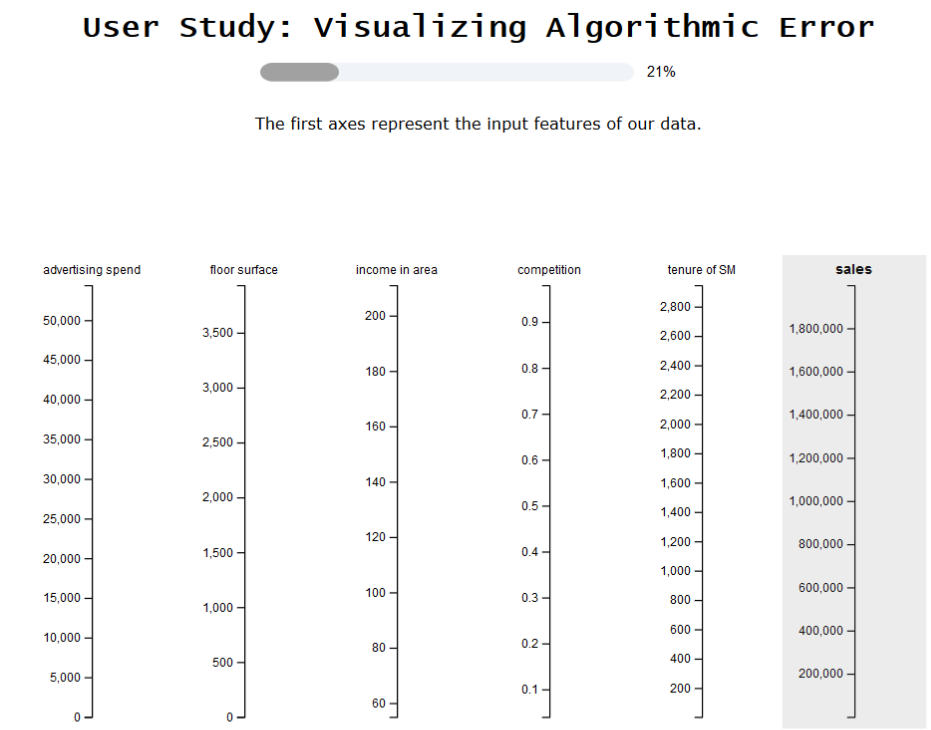


Figure B.5: Introduction to the method, part 3.

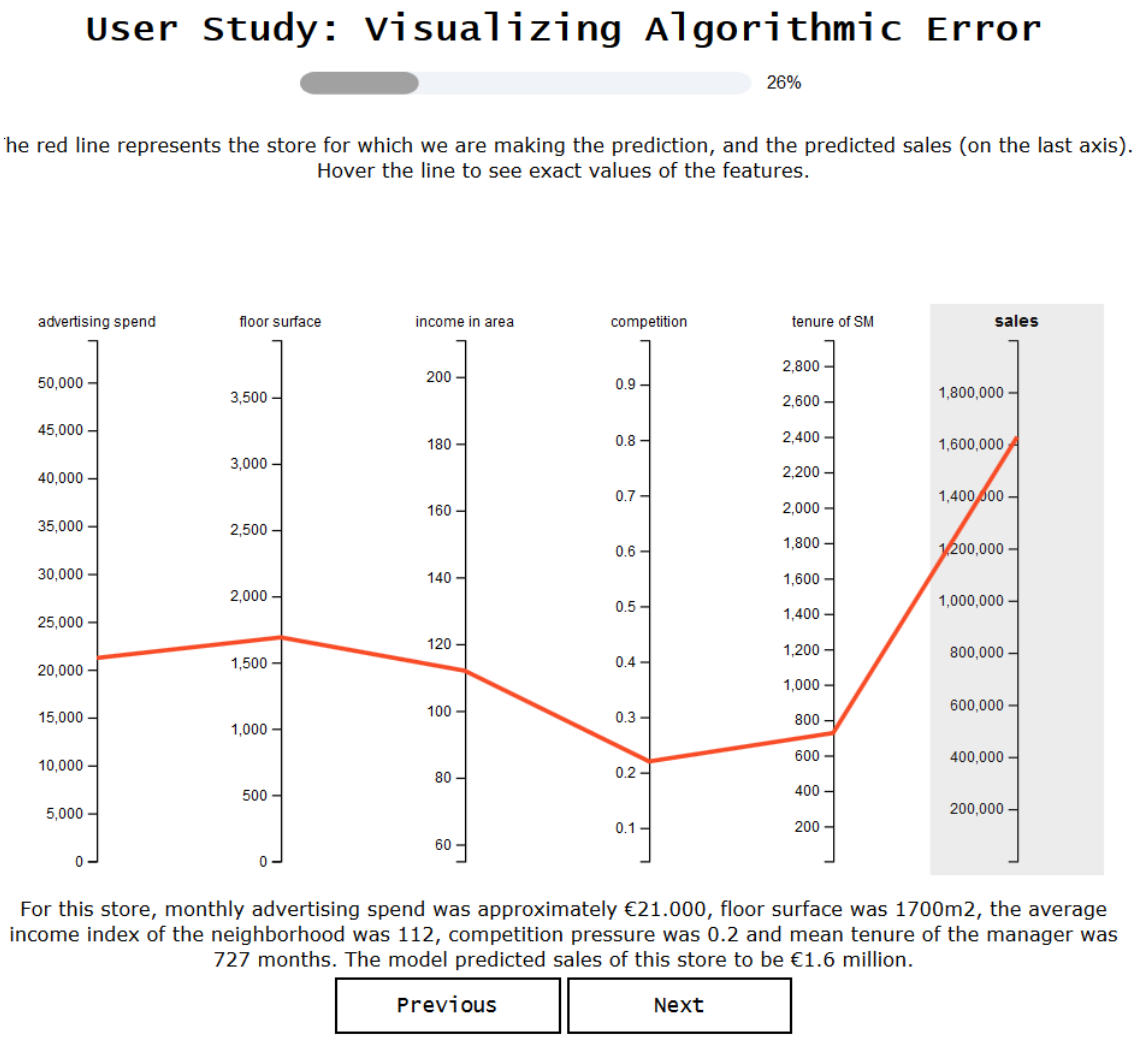


Figure B.6: Introduction to the method, part 4.

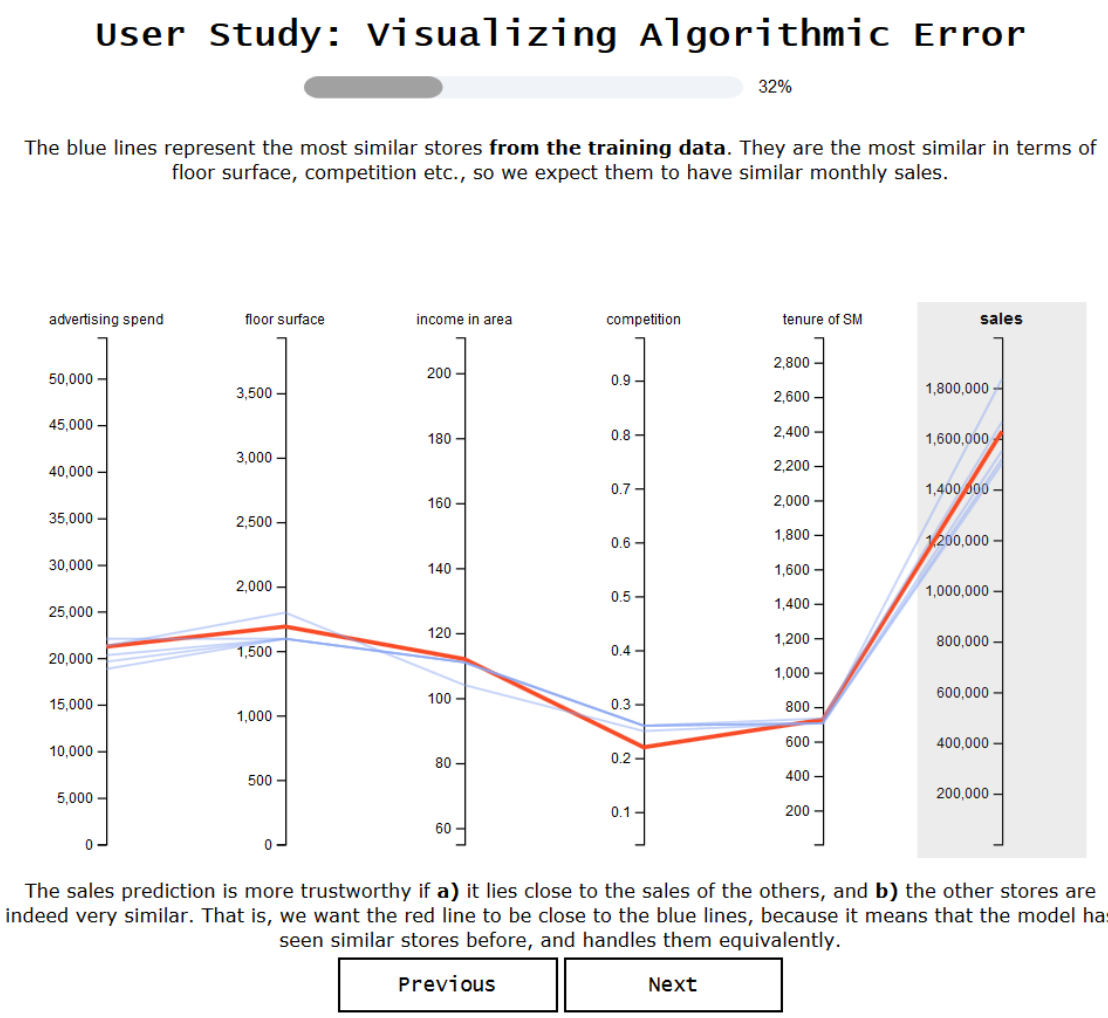


Figure B.7: Introduction to the method, part 5.

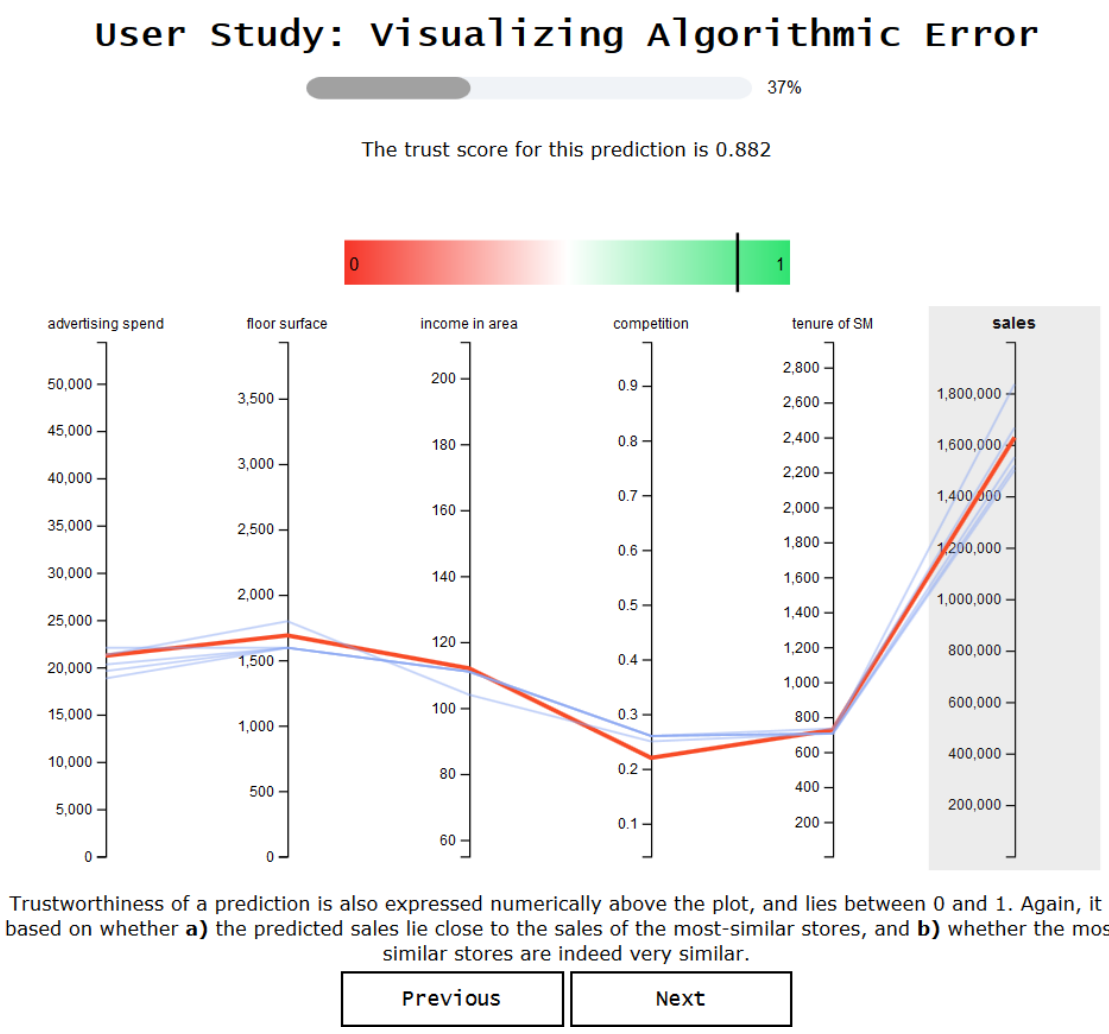


Figure B.8: Introduction to the method, part 6.

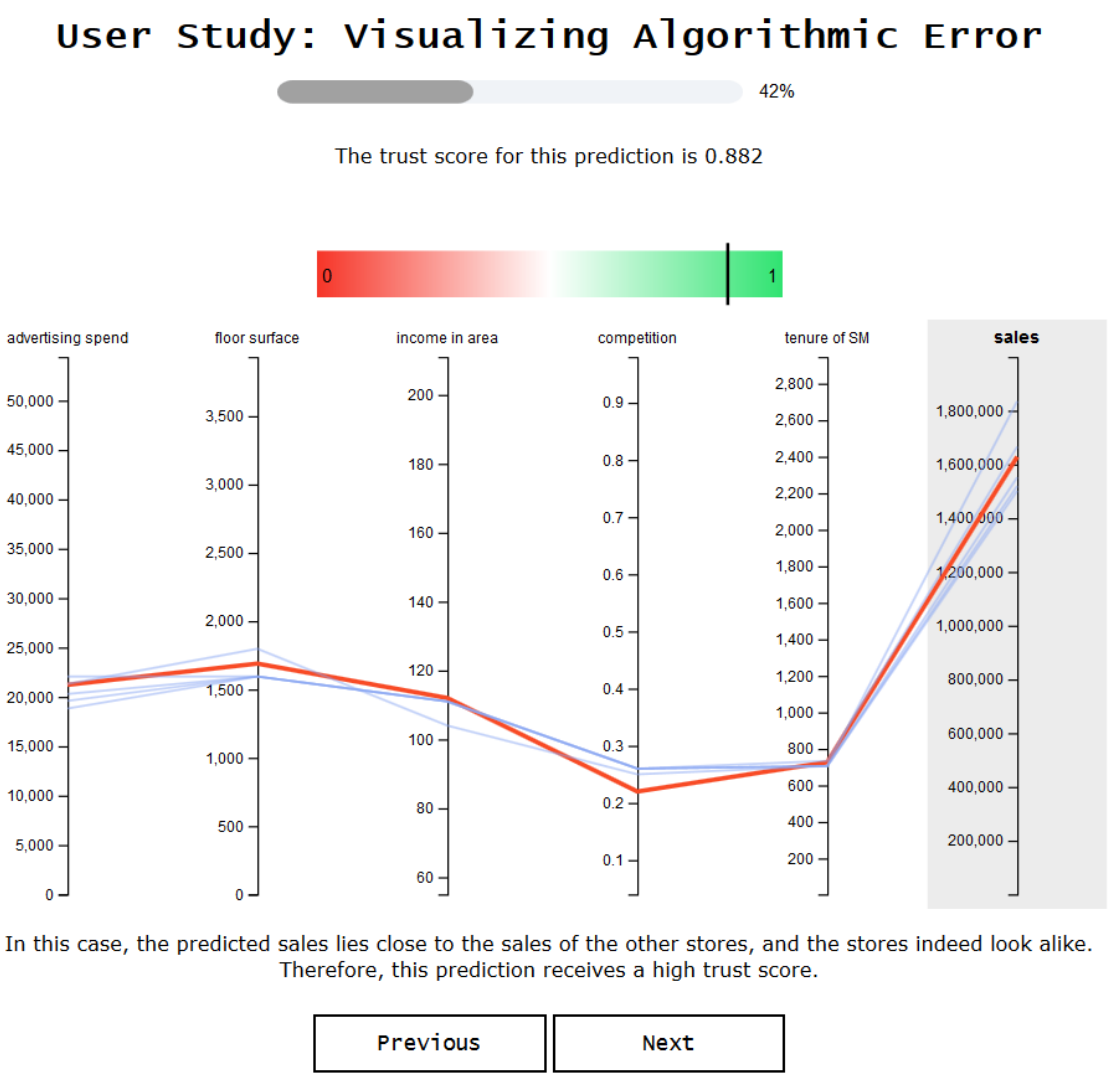


Figure B.9: Introduction to the method, part 7.

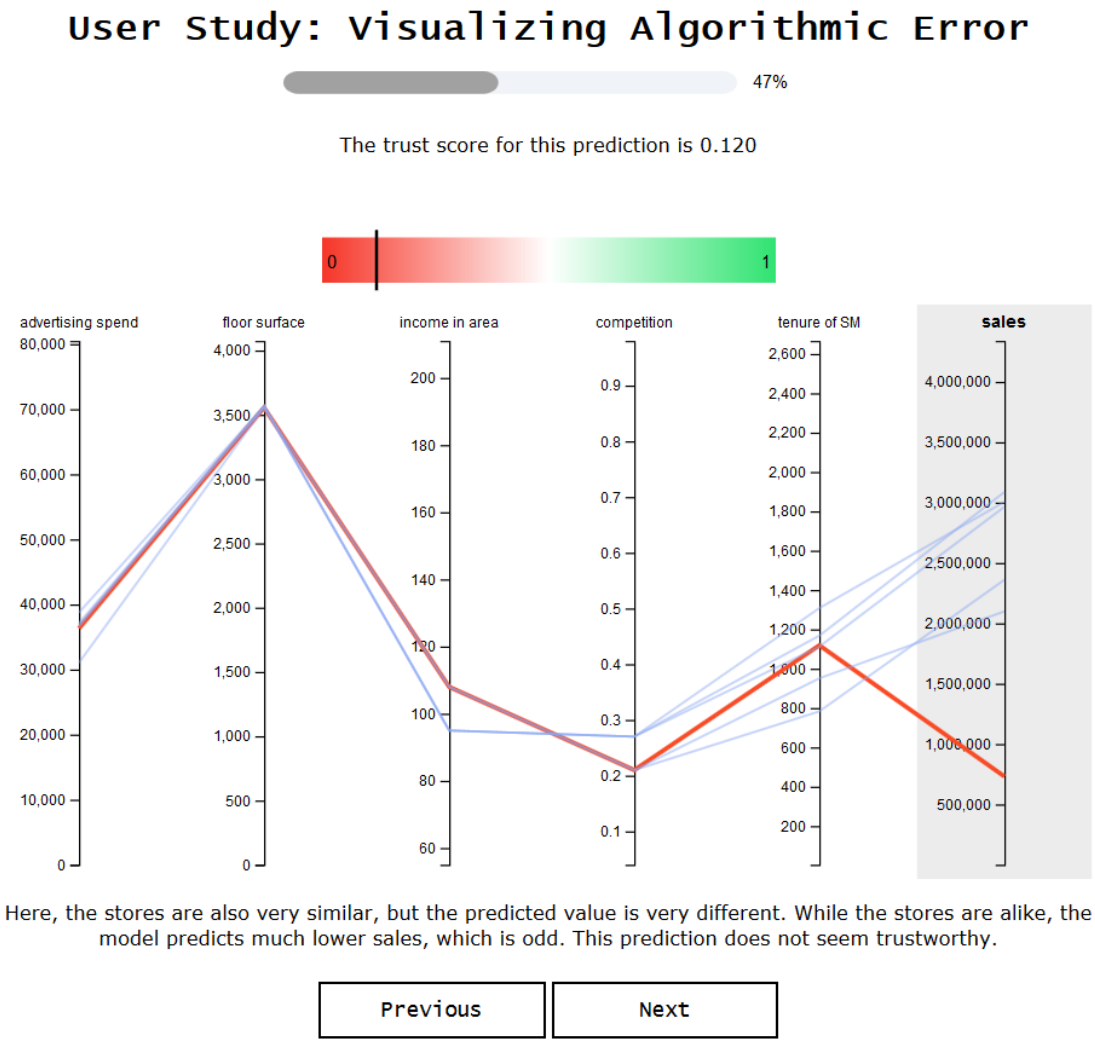


Figure B.10: Introduction to the method, part 8.

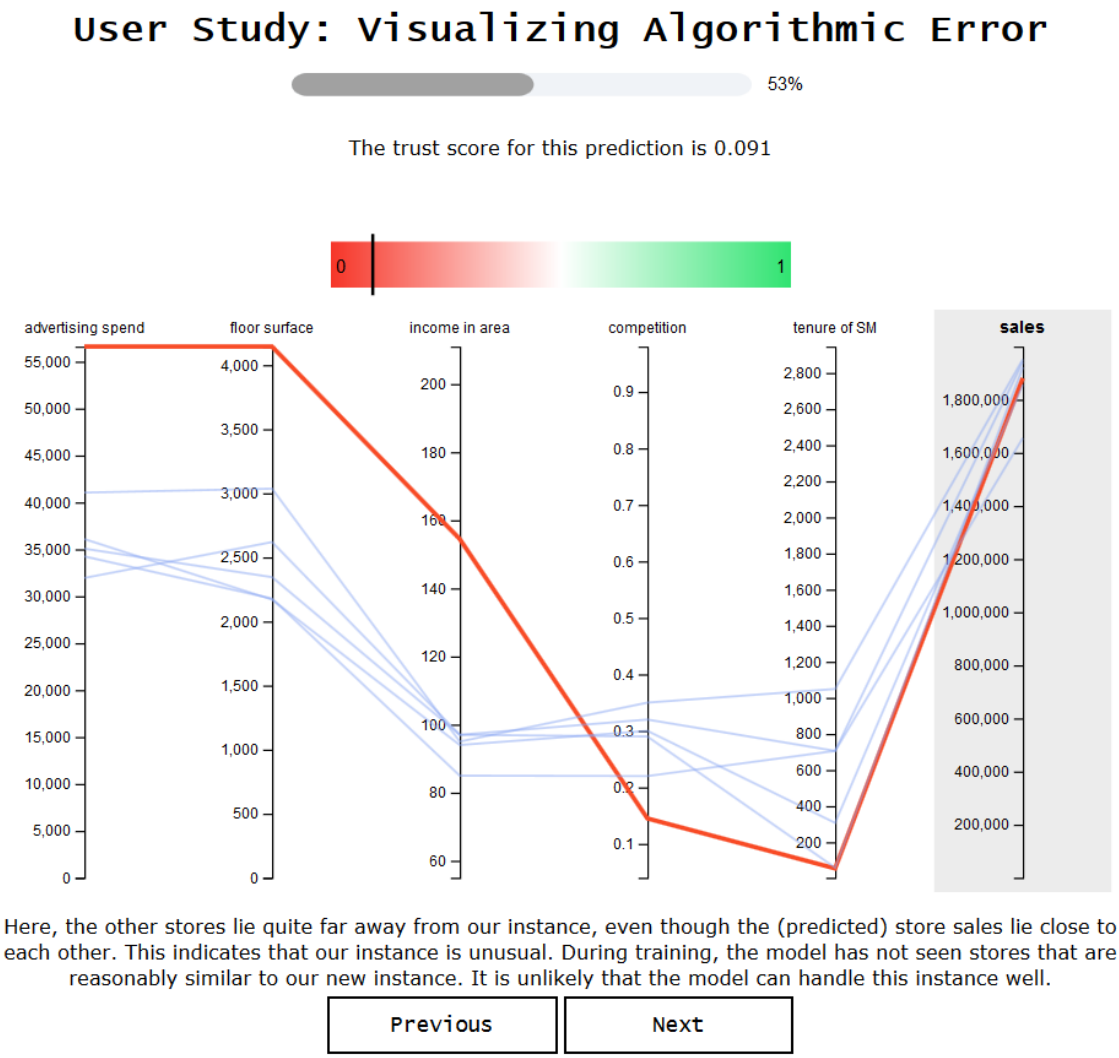


Figure B.11: Introduction.

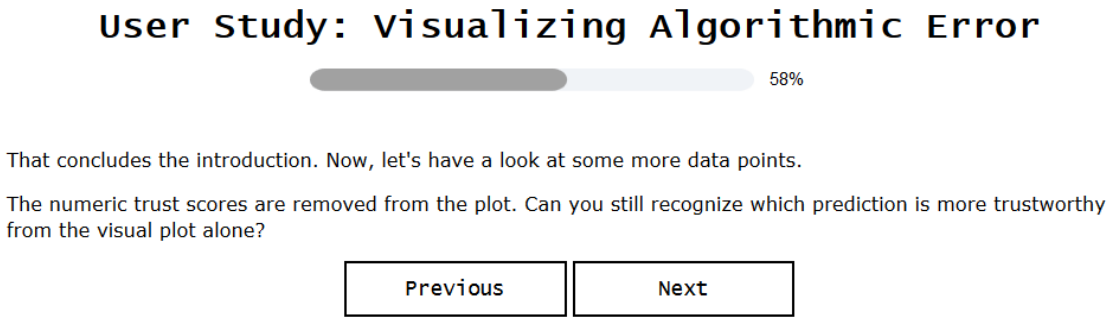


Figure B.12: Transition to user questions.

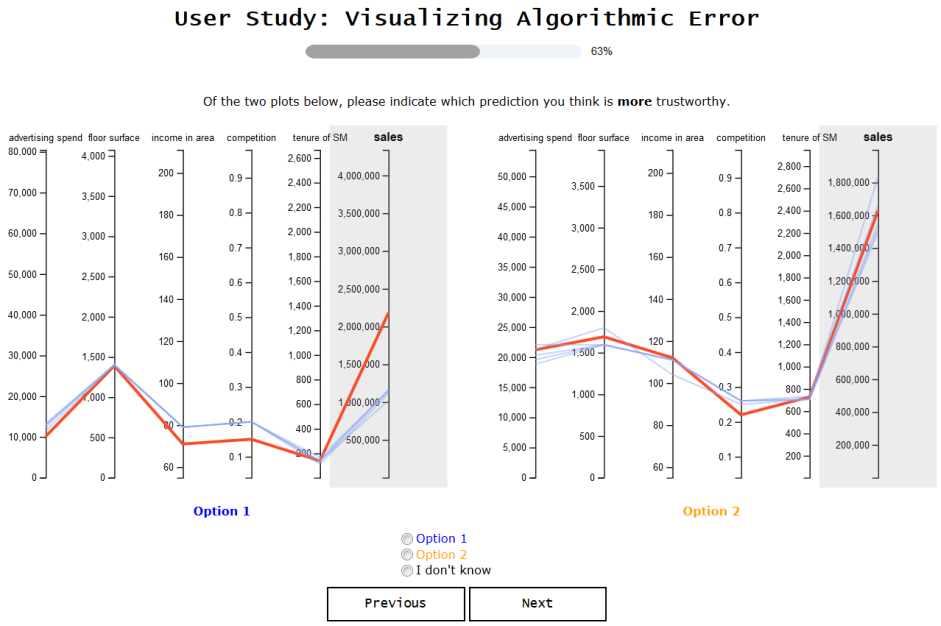


Figure B.13: Pairwise comparison, part 1.

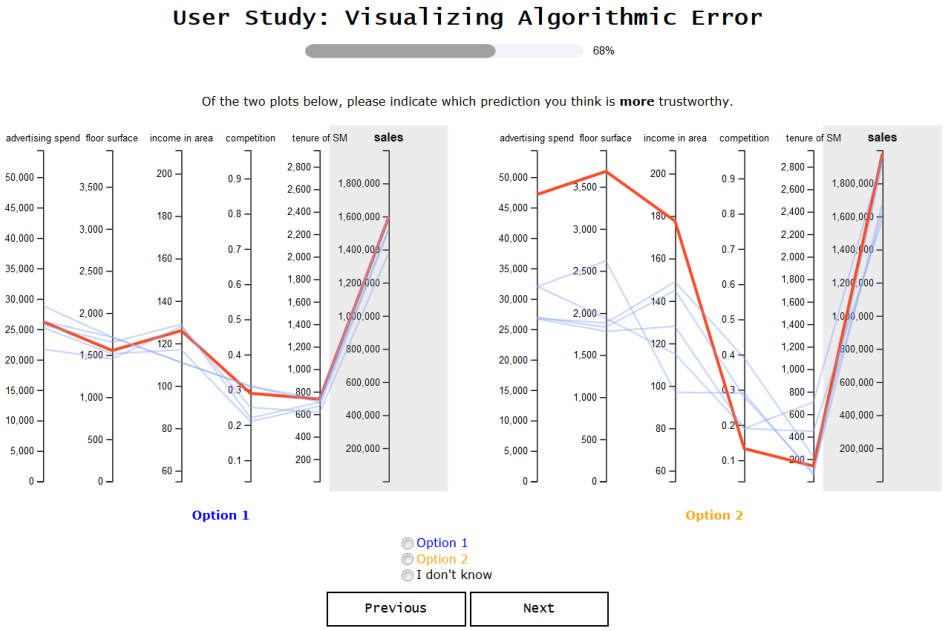


Figure B.14: Pairwise comparison, part 2.

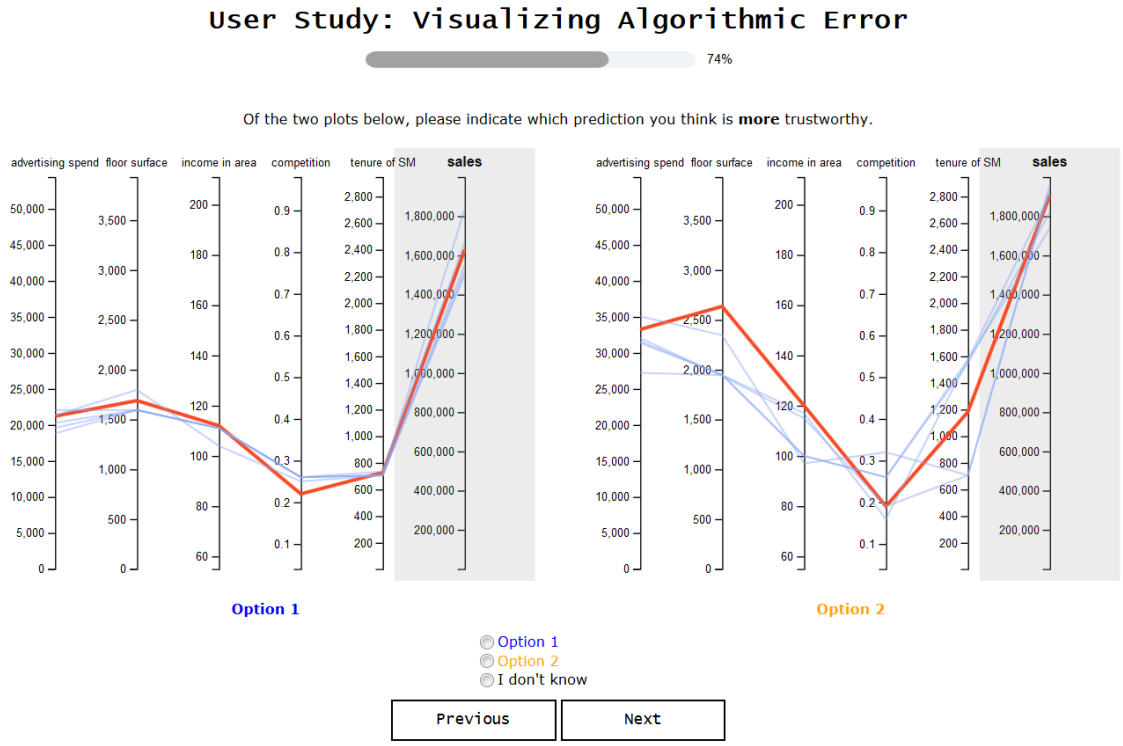


Figure B.15: Pairwise comparison, part 3.

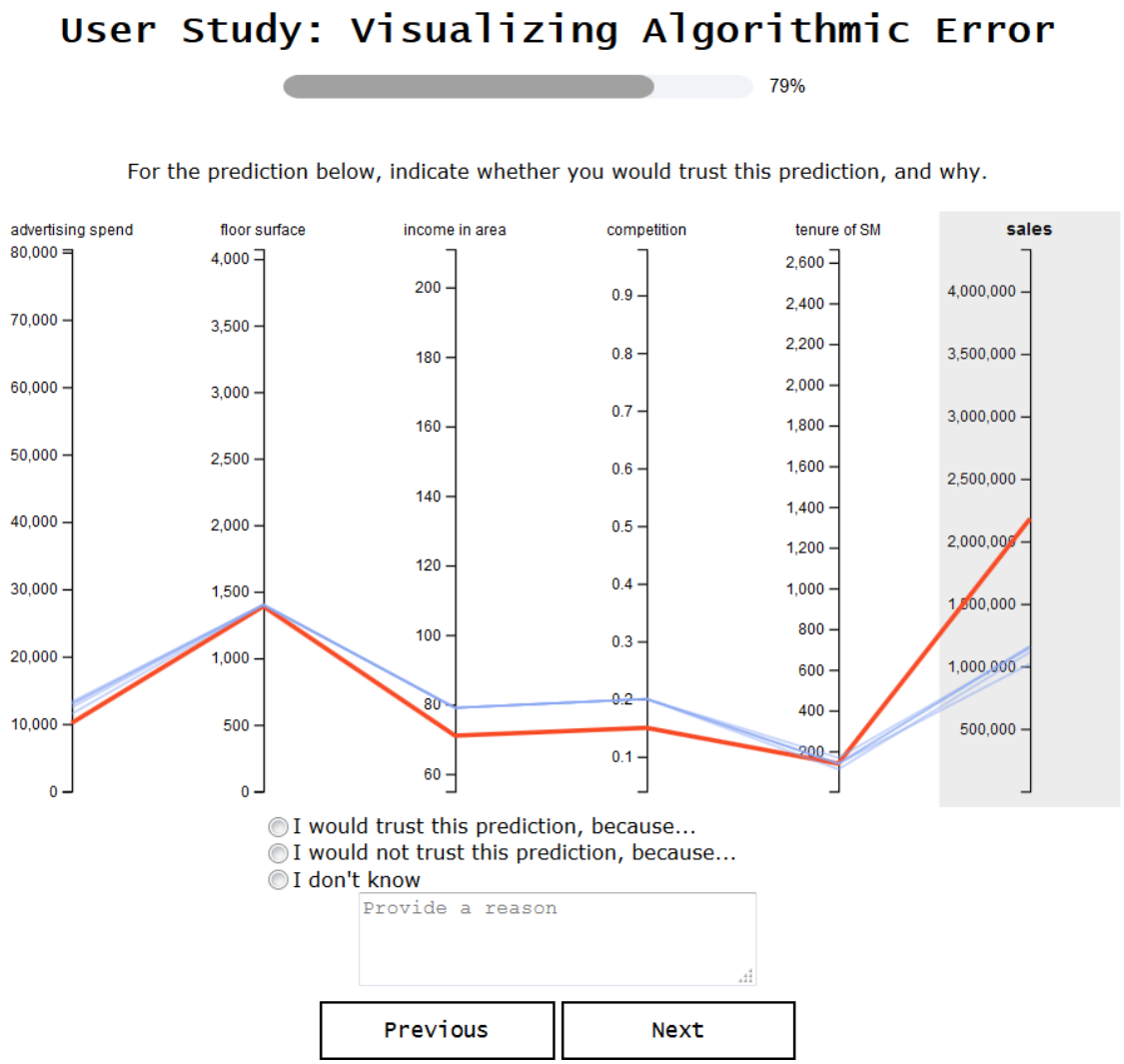


Figure B.16: Single evaluation, part 1.

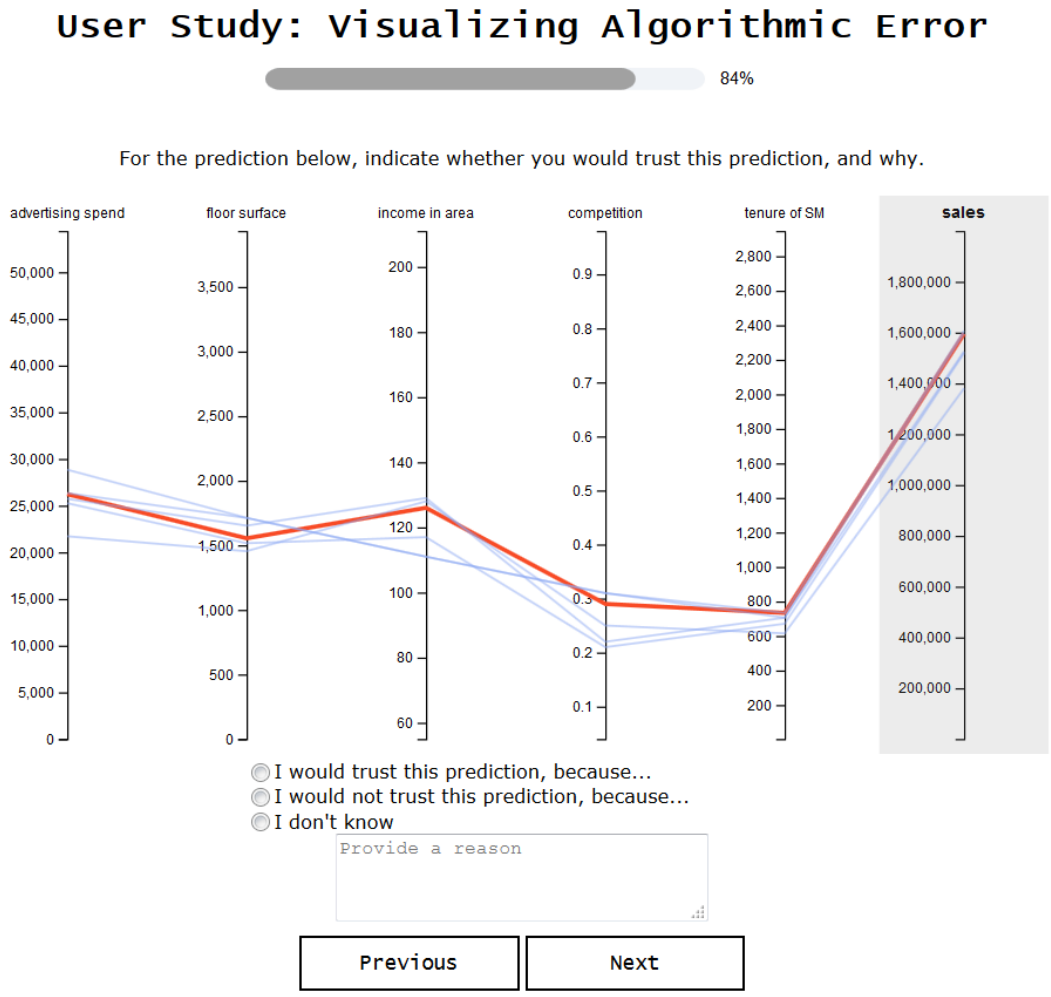


Figure B.17: Single evaluation, part 2.

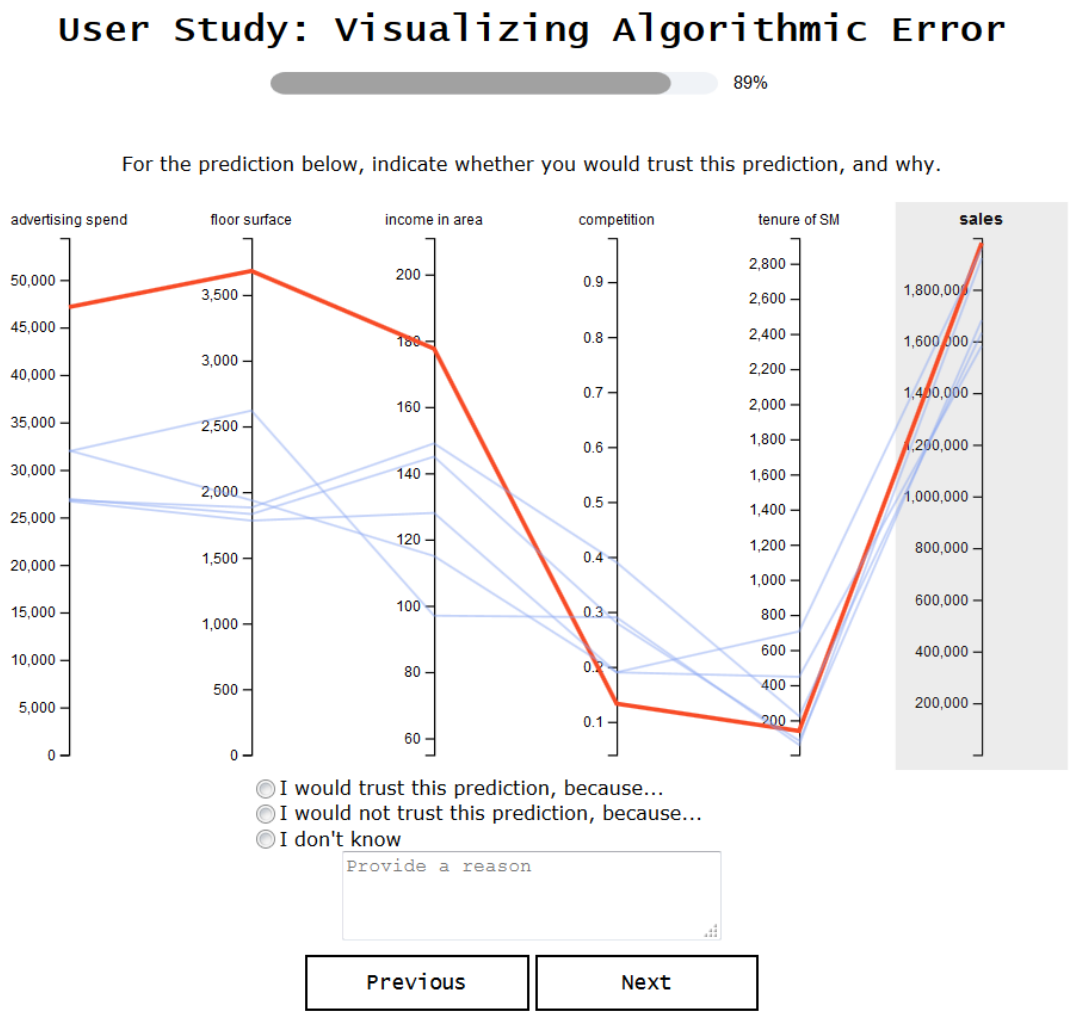


Figure B.18: Single evaluation, part 3.

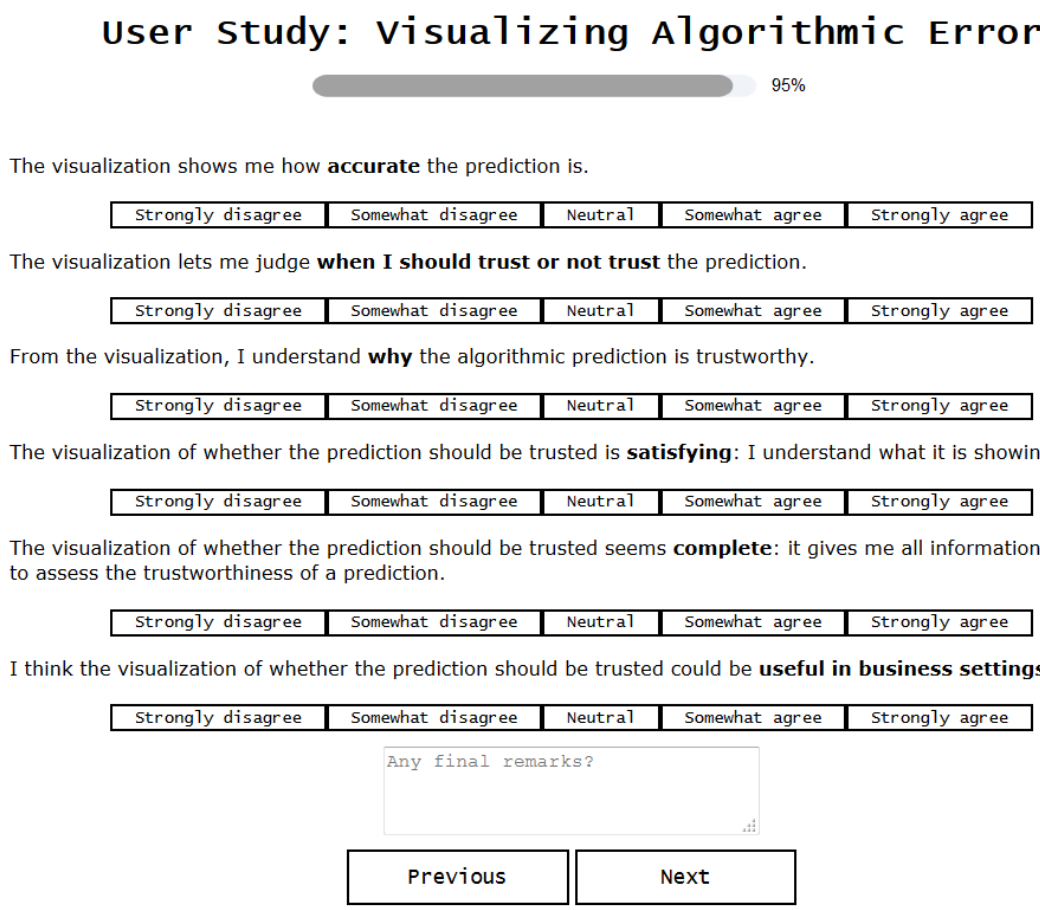


Figure B.19: Qualitative questions.

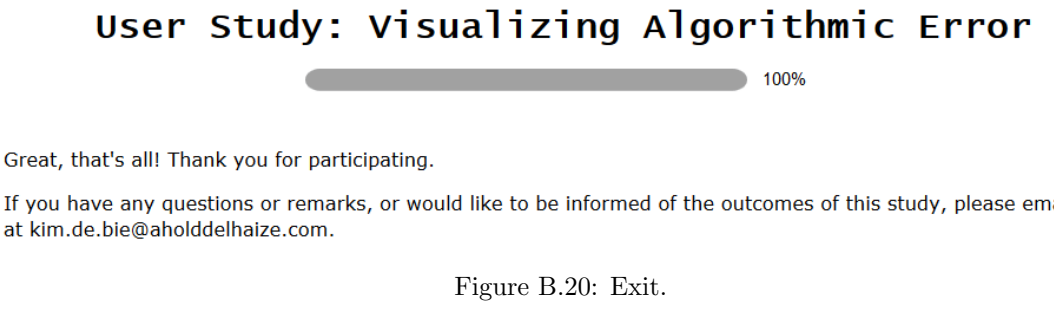


Figure B.20: Exit.