Journal of Artificial Intelligence Research 76 (2023) 829-881 Submitted 03/2022; published 03/2023

Visualizing the Implicit Model Selection Tradeoff

Zezhen (Dawn) He ZEZHEN.HE@SIMON.ROCHESTER.EDU
Simon Business School, University of Rochester
Rochester, NY 14627

Yaron Shaposhnik YARON.SHAPOSHNIK@SIMON.ROCHESTER.EDU
Simon Business School, University of Rochester
Rochester, NY 14627

Abstract

The recent rise of machine learning (ML) has been leveraged by practitioners and researchers to provide new solutions to an ever growing number of business problems. As with other ML applications, these solutions rely on model selection, which is typically achieved by evaluating certain metrics on models separately and selecting the model whose evaluations (i.e., accuracy-related loss and/or certain interpretability measures) are optimal. However, empirical evidence suggests that, in practice, multiple models often attain competitive results. Therefore, while models' overall performance could be similar, they could operate quite differently. This results in an implicit tradeoff in models' performance throughout the feature space which resolving requires new model selection tools.

This paper explores methods for comparing predictive models in an interpretable manner to uncover the tradeoff and help resolve it. To this end, we propose various methods that synthesize ideas from supervised learning, unsupervised learning, dimensionality reduction, and visualization to demonstrate how they can be used to inform model developers about the model selection process. Using various datasets and a simple Python interface, we demonstrate how practitioners and researchers could benefit from applying these approaches to better understand the broader impact of their model selection choices.

1. Introduction

In recent years, there has been a surge in the application of machine learning (ML) algorithms to solve core business problems in information systems (Bose & Mahapatra, 2001; Fu, Huang, & Singh, 2021; Shin et al., 2020; Xia Liu, Li, & Xu, 2021), operations (Choi, Wallace, & Wang, 2018; Mišić & Perakis, 2020; Qi, Mak, & Shen, 2020), marketing (Brei et al., 2020; Ma & Sun, 2020), accounting (Bertomeu, 2020), and finance (Dixon, Halperin, & Bilokon, 2020; Emerson, Kennedy, O'Shea, & O'Brien, 2019; Rundo, Trenta, di Stallo, & Battiato, 2019). These include problems in pricing, revenue management, and supply chain management. An important step in the development of predictive models is model selection, which refers to processes and techniques for choosing the best model from a set of candidate models, often with the goal of optimizing prediction accuracy. This can be done for the purposes of selecting a model among different classes, such as between classification trees or support vector classification models (Friedman, Hastie, Tibshirani, et al., 2001), and for hyper-parameter tuning (i.e., selecting the best model within a family of models, such as decision tree classifiers with different tree depths or linear models with different number of linear coefficients). In both cases, model selection is usually performed by optimizing quantifiable metrics that act as a proxy to or bound the generalization error, which is the

measure of how well a model trained on one dataset (the training set) predicts the outcome on a new dataset (the validation or test sets). This is achieved with techniques such as cross validation (CV) or by using information criteria such as the Akaike information criterion (AIC) or Bayesian information criterion (BIC) criteria (see Claeskens & Hjort, 2008).

Traditionally, the common wisdom was that since interpretability tradeoffs flexibility, as a consequence it also tradeoffs performance. Figure 1 represents this view (James, Witten, Hastie, & Tibshirani, 2013). This figure contains a collection of models that vary in their interpretability and flexibility. The simpler models (e.g., Lasso) appear on the top left (low flexibility and high interpretability), and the increasingly more complex and less interpretable models appear towards the bottom right. The figure conveys the intuition that flexbility and therefore performance comes at a cost–when flexibility improves, interpretability worsens.

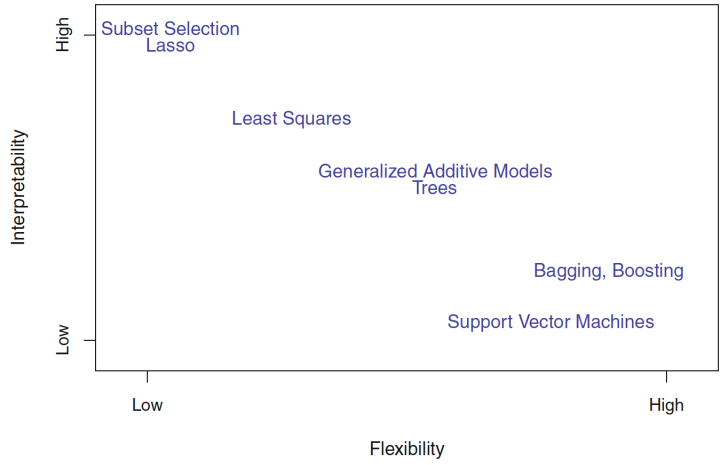


Figure 1: A typical illustration of the interpretability–flexibility tradeoff (James et al., 2013).

However, recent studies (e.g., Dong & Rudin, 2020; Fisher, Rudin, & Dominici, 2019; Semenova, Rudin, & Parr, 2020) have challenged this view, suggesting that there are often times multiple models that can be trained that are comparable in their overall performance. This was termed the *Rashomon set phenomenon* by Breiman (2001). As an example, consider Figure 2, which compares the prediction accuracy on four datasets (described in Appendix B). The figure is partitioned into four subplots corresponding to the datasets: in each subplot, the X-axis represents models (detailed in Appendix A), while the Y-axis represents the mean and standard deviation of the CV accuracy. For each dataset, 15 models are compared, 10 of which are standard models (shown in red) and five are more modern interpretable models (shown in blue). We observe that the performance of the best models for each of the datasets is often quite similar in terms of accuracy. Therefore, simply computing the CV accuracy is not informative for deciding how to choose among the models.

(a) Adult dataset

(b) Bank Marketing dataset

(c) Mushroom dataset

(d) FICO (binary) dataset

Figure 2: The CV accuracy of 10 standard box models (red, 10 leftmost models along the X-axis) and five interpretable models (blue, five rightmost models along the X-axis). The intervals represent the mean ± one standard deviation.

Note that while the models displayed in Figure 2 are comparable in terms of their accuracy, they are of different functional forms. For example, for the FICO (binary) dataset, the linear (logistic regression), generalized additive (EBM), non-parametric (SVM), and neural network (MLP) models achieve similar accuracy, yet operate in fundamentally different ways. This is illustrated in Figure 3, which shows the pairwise agreements between predictions of different models on the Bank Marketing dataset. The X-axis and Y-axis list a collection of different models, and the numbers in the table indicate the percentage of the agreement on each pair of models' predictions. Specifically, using the test data, we make predictions for all models, and for each pair of models, which corresponds to an entry in the matrix, we compute the percentage of predictions on which the models predict the same labels. For example, the value 0.98 in the cell corresponding to the models LR and MLP means that on 98% of the test set observations, the model LR agrees with the model MLP in terms of their predictions. The figure also shows (on the left column and top row) the test accuracy of each model by comparing the models to the actual labels (denoted as Y). The level of agreement varies across pairs of models, and multiple models (ADB, LR, LSVC, MLP, EBM) achieve the highest test accuracy of 74% (on the left column). These models are similar in terms of their accuracy, but are otherwise quite different—tree ensembles (ADB), linear models (LR and LSVC), neural networks (MLP), and generalized additive models (EBM) each describe a different functional relation between input and output vari-

ables. We note in passing that a similar intuition is conveyed by computing the mutual information (MacKay, 2003) between pairs of models (see Figure 24 in Appendix F).

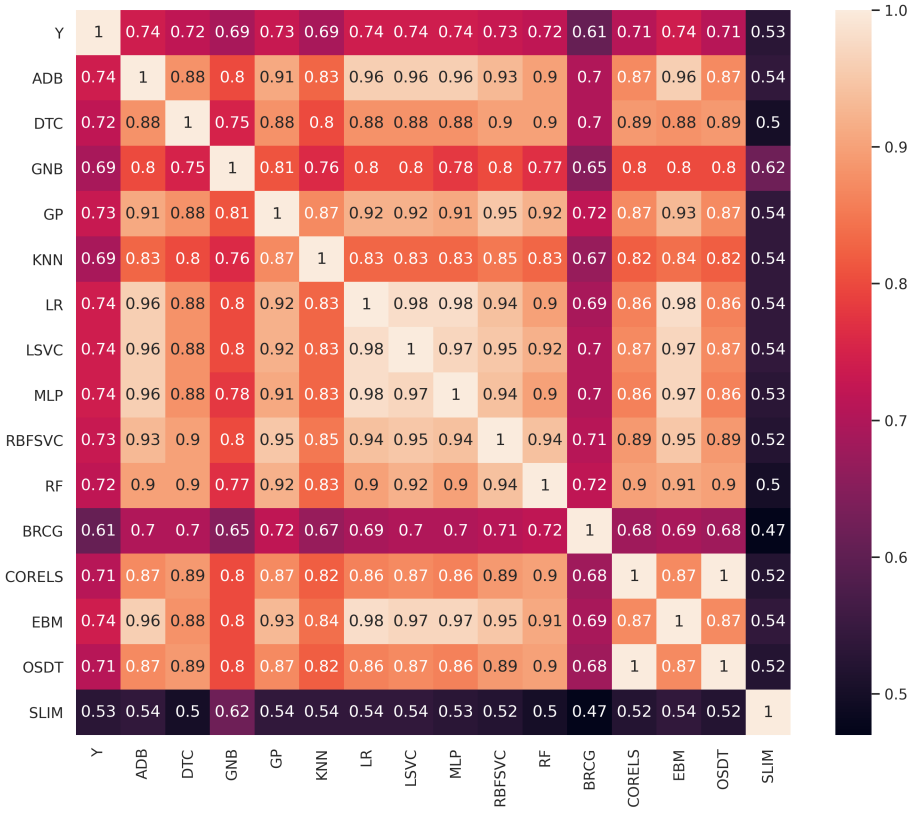


Figure 3: A heatmap showing the model prediction agreement level for the FICO (binary) dataset. A value of 0.98 in the cell (LR, MLP) indicates that the models LR and MLP agree on 98% of their predictions on the test data.

This observation raises the following question: how should one select a model when several models are equivalent in terms of performance, but are actually different in terms of their inner working? The latter implies that these models tradeoff prediction accuracy across various parts of the feature space. As an example of the tradeoff, consider Figure 4, which illustrates two different linear models that achieve the same prediction accuracy, each misclassifying two observations. Model 1 predicts "+" in regions A and D, and "-" in regions B and C. Model 2 predicts "+" in regions A and B, and "-" in regions C and D. We see that Model 1 and Model 2 are identical in regions A and C, but make opposite predictions in regions B and D. Therefore, they err differently in different parts of the feature space. While this difference does not affect the overall accuracy, it could have a dramatic impact on individuals and on how different parts of the population are affected by the models. As an example, for estimating credit risk, there could be two models that are identical in their overall performance and in their predictions on 90% of the population. However, on two sub-populations (regions B and D) the models predict differently, and therefore make different types of errors. As another example, consider medical diagnosis. Two models

could be similar in their overall performance, but each of the models could make different types of errors (false positive and false negative) on different groups of patients. Unveiling the trade-off when choosing between models is important to understand the broader impact of model selection decisions; that is, to clarify how the predictive models manage risk across the population, and ideally, to identify their societal effects prior to deployment.

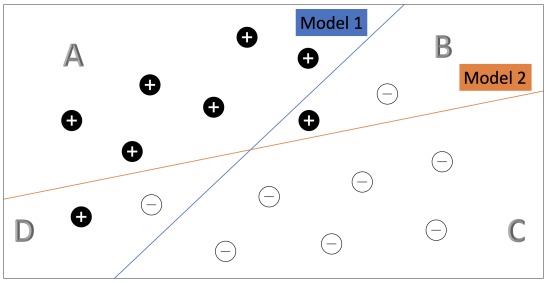


Figure 4: Two linear models with similar accuracy that predict differently throughout the feature space.

More specifically, we ask and attempt to answer the following research questions: *How and whether comparing predictive models can be done in an interpretable manner to expose the implicit model selection tradeoff?* In the example of linear models, one could represent the model differences using the polyhedra that define the respective parts of the feature space. However, can this be done in the general case of non-linear, possibly non-parametric, models? Can this be done when models are not explicitly given, but only their predictions? Moreover, can this be done in an interpretable manner? If the answers to the above questions is yes, then we might be able to make better model selection decisions by understanding how predictive models affect the population differently and how risk is managed.

Results and contributions. Our paper studies a fundamental problem in ML that greatly impacts practices in management sciences, and develops a practical and comprehensive tool-set to solve it. Specifically,

1. We introduce a new methodology for comparing models, which we call *interpretable comparative meta-models*. Given the predictions of two models, it produces a meta-model that describes the parts of the feature space in which the models make similar and different predictions.

2. We develop new visualization methods for comparing two or more models. For example, the *visual model landscapes* plot places on a two dimensional plane a collection of models that show which models are most similar to each other. To provide a comprehensive treatment of the problem, we also discuss and name some straightforward methods which we find useful (to avoid repetitions, we elaborate on each of the methods in the body of the work in dedicated sections). The methods are based on visualization techniques (e.g., scatter plots, heatmaps, and dendrograms), supervised learning algorithms (in particular, interpretable ML models), unsupervised learning methods (e.g., clustering and dimensionality reduction), as well as other standard ML methods such as feature importance computation and confusion matrices.

3. We demonstrate how these methods can be used for various datasets from the UCI ML Repository (Dua & Graff, 2017). We also conduct a case study in which we apply the methods in the context of recidivism risk prediction to show how the methods we developed can be used to understand the implications of using different models.

4. We make the code available online to facilitate exploration and adoption (see Appendix E).

Organization. The rest of the paper is structured as follows. In Section 2, we review related work. In Section 3, we introduce notation that will be helpful to describe the methods. We then continue to discuss each of the methods: interpretable comparative meta-models (Section 4), visual model landscapes (Section 5), visual confusion matrices (Section 6), and visual comparative matrices (Section 7). In Section 8, we present a case study using the COMPAS dataset (Larson, Mattu, Kirchner, & Angwin, 2016) on criminal recidivism in which we show how our methods can be used to explain differences between models. Finally, we conclude the paper in Section 9 with a discussion and suggestions for future work.

2. Literature Review

There are four main bodies of work related to our paper: analytics as a domain area of ML, model selection as we study a general model selection problem, the Rashomon set theory (which is concerned with the existence of multiple top-performing models), and interpretable ML (whose methods we employ).

Analytics. Analytics can be defined as the "... science of using data to build models that add value to decisions made by companies, insitutions and individuals." (Bertsimas, Allison, & Pulleyblank, 2016). In many business problems, the goal is often to optimize decisions using data or ML models as primitives (Ban, El Karoui, & Lim, 2018; A. Elmachtoub, Liang, & McNellis, 2020; A. N. Elmachtoub & Grigas, 2021; Ferreira, Lee, & Simchi-Levi, 2016; Guo, Grushka-Cockayne, & De Reyck, 2021; J. Sun, Zhang, Hu, & Van Mieghem, 2021; K. Wang et al., 2021). However, in many business problems, prediction itself is the main objective. For example, the prediction of waiting time (Ibrahim, 2018; Kuo et al., 2020; Y. Zhang, Nguyen, & Zhang, 2013), performance prediction (Bertsimas, Brynjolfsson, Reichman, & Silberholz, 2015; Melançon, Grangier, Prescott-Gagnon, Sabourin, & Rousseau, 2021), medical outcomes (Bertsimas, O'Hair, Relyea, & Silberholz, 2016; Center, 2021; Lee et al., 2016; Shah et al., 2019), demand forecasts (Bertsimas, Pauphilet, Stevens, & Tandon, 2021; Lin et al., 2020), etc. In the latter examples, and especially when prediction is an end and not a primitive of an optimization problem, understanding the implicit model selection tradeoff is important. This could potentially help uncover performance issues and biases while improving the overall quality of the resulting model. See Choi et al. (2018), Qi et al. (2020), and Mišić and Perakis (2020) for recent surveys on operations analytics literature.

Model selection. The literature on model selection studies methods for selecting a model from a candidate set of models. This is typically conducted by balancing a loss function (e.g., log-likelihood) with an additional term for complexity, where the complexity term is added for the purposes of avoiding overfitting and improving interpretability. Claeskens and

Hjort (2008) present an overview of selection criteria such as AIC, BIC, the deviance information criterion (DIC), the focused information criterion (FIC), the likelihood ratio test, etc. Naturally, different selection criteria lead to training different models as the definition of optimal solution varies. Rust, Simester, Brodie, and Nilikant (1995) argued that combining multiple criteria does not outperform a single criterion. The authors recommended using simpler, less computationally intensive criteria, such as the Schwarz criterion, for model selection. Shen and Ye (2002) proposed an adaptive model selection criterion that employs a data-driven complexity penalty instead of traditional non-adaptive selection criteria. Busemeyer and Wang (2000) offered a "generalization criterion" for model selection that tends to select simpler models that generalize better. This method consider training and validation samples from completely different experimental designs, instead of the same design for all folds as in cross-validation, in order to assess the generalizability of models, with the assumption of parameter invariance. In addition to regular and robust model selection criteria, Rao, Wu, Konishi, and Mukerjee (2001) surveyed various model selection methods in the literature including hypothesis testing, minimization of prediction error, cross-validation, bootstrapping, the Bayesian approach, etc. Kadane and Lazarand (2004) reviewed traditional methods for choosing among models that are related to feature importance analysis. For example, computing the impact on the SSE (sum of squared errors) when excluding some variables in the original model; by using forward selection that starts from an empty set and adds variables with the highest importance one by one from the variable set without replacement; by using backward elimination that starts from a full set and subtracts variables with the lowest importance; by using step-wise selection process that combines forward and backward methods; or by selecting a set of variables from the feature space based on some other criteria and using these features to compare competing models.

More recently, researchers developed interactive visual tools to aid users with model selection decisions. Ren, Amershi, Lee, Suh, and Williams (2016) proposed Squares, which is a visual tool to compare multiclass classifiers based on their predictions of each class. D. Sun et al. (2020) proposed DFSeer, an interactive tool that ranks models using adjustable weights of various performance metrics and allows drilling-down to observe model performance at the cluster or item level (the information available on each item is a time series). Their primarily goal is to perform stacking (Wolpert, 1992), that is, to decide how to partition the feature space in order to assign a suitable model to each partition. J. Zhang, Wang, Molino, Li, and Ebert (2018) introduced Manifold, a tool that consists of model comparison plots that compare the scores predicted by each model for each label. The predictions are organized in a confusion matrix based on whether the models agree or disagree on the predictions. Additional plots provide information about the distribution of feature attribution by each model. Finally, Narkar, Zhang, Liao, Wang, and Weisz (2021) discussed a tool called Model LineUpper, which compares models based on summary statistics of their performance, distribution of feature importance, and prediction scores.

While our work utilizes the idea of pairwise comparison, our techniques are conceptually and technically different from that of the previous studies. The guiding principle of our solutions attempts to elicit how models differ throughout the feature space. To our knowledge, our comparative meta-model and model landscape plots are novel, as well as other types of model comparison plots that synthesize dimensionality reduction and confusion matrices.

Finally, we briefly note that the topic of debugging machine learning models has some bearing to our work as it deals with model inspection (see, e.g., Cadamuro, Gilad-Bachrach, & Zhu, 2016; Chakarov, Nori, Rajamani, Sen, & Vijaykeerthy, 2016; Lourenço, Freire, & Shasha, 2019; X. Zhang, Zhu, & Wright, 2018). The focus in this line of work, however, is quite different, as it aims to understand when a model makes wrong predictions and how to correct it. For example, by identifying the set of training samples that are responsible for prediction errors and improving the model performance by altering this subset. Our paper, on the other hand, aims to determine which sub-populations do two models trade off in terms of their predictions.

Rashomon set theory. Breiman (2001) coined the term *Rashomon set* as a set of predictive models that perform equally well. Recently, researchers developed the Rashomon set theory to better understand this phenomenon, which challenges the common view that complexity and performance must be traded-off with interpretability. They showed that there is often not a single model that clearly dominates the rest of the hypothesis space; rather, there could be multiple near-optimal candidate models. Building on existing measures of variable importance and the fact that different models place different importance on variables of interest, Dong and Rudin (2020) proposed a way to evaluate Rashomon sets by introducing *variable importance clouds* which are mappings of variable importance to models within the Rashomon set to provide further comparison measures of models that perform equally well. Fisher et al. (2019) proposed the *model class reliance* (MCR) which returns a range (upper and lower bound) for the variable importance in near-optimal models within a Rashomon set as a novel approach to providing a comprehensive explanation. Semenova et al. (2020) introduced the Rashomon ratio (the fraction of models from the hypothesis space that are in the Rashomon set) as a measure for the simplicity of models in the hypothesis space. According to the authors, a high Rashomon ratio (i.e., a comparatively large Rashomon set) indicates that the candidate set contains many well-performing simple models with low loss. The authors propose to generate the Rashomon curve from the Rashomon ratio, and use it to indicate the optimal balance of fit and complexity.

Our work is related to the Rashomon set phenomenon, but our focus is different. Rather than understanding why it happens and how to take advantage of it when dealing with a specific hypothesis class, we develop methods to compare arbitrary models in an interpretable manner.

Interpretable machine learning. The field of interpretable ML seeks to develop transparent and simple predictive models that can be understood and used by relevant stakeholders. According to Rudin et al. (2021), "an interpretable machine learning model obeys a domain-specific set of constraints to allow it to be more easily understood by humans. These constraints can differ dramatically depending on the domain." One of our methods for model comparison utilizes interpretable ML models. For our application, it is important to be able to interpret the inner working of ML models that are used to infer how comparable models differ. Examples for some of the recent work on interpretable ML models can be found in the work of Bennetot et al. (2021); Doshi-Velez and Kim (2017, 2018); Du, Liu, and Hu (2019); Holzinger, Saranti, Molnar, Biecek, and Samek (2022); Molnar (2020); Murdoch, Singh, Kumbier, Abbasi-Asl, and Yu (2019).

To our knowledge, this paper is the first work to develop methods for interpretable comparison of predictive models.

3. Problem Formulation and Experimental Setup

In this section, we introduce notation that we use to formalize the problem and describe the experimental setup.

3.1 Notation and Problem Formulation

Denote by $\mathbf{X}$ a data matrix consisting of N observations $\{\mathbf{x}_i\}_{i=1}^N$ that are characterized by P features (that is, $\mathbf{x}_i \in \mathcal{R}^P$), and a corresponding vector of labels $\mathbf{Y}$. We consider binary classification problems where each label y_i in $\mathbf{Y}$ satisfies $y_i \in \{0, 1\}$ (later in Section 8, we demonstrate how to apply methods for comparison of classification models to regression models). We are given vectors with predictions $\mathbf{Y}^k$ of M classification models $\{h^k\}_{k=1}^M$, where $h^k : \mathcal{R}^P \to \{0, 1\}$, and the prediction of model h^k for observation $\mathbf{x}_i$ is denoted as $\mathbf{y}_i^k = h^k(\mathbf{x}_i)$. To simplify the exposition, we will at times simply refer to model h^k as k when it is clear from the context.

Our goal is to derive insights into how the models $\{h^k\}_{k=1}^M$, and in particular how every pair of models h^j and h^k, relate to each other—that is, when they predict similarly and differently. To this end, we define the *"joint prediction"* as the combined predictions by two compared models (i.e., $\{00, 01, 10, 11\}$), and define the four joint labels Y_{00}^{jk}, Y_{01}^{jk}, Y_{10}^{jk}, and Y_{11}^{jk}. The vector $\mathbf{Y}_{00}^{jk}$ represents labels of observations for which both models j and k predict 0; that is, element i of $\mathbf{Y}_{00}^{jk}$ is defined as $\mathbb{1}[y_i^j = 0 \text{ AND } y_i^k = 0]$. Similarly, the vector $\mathbf{Y}_{01}^{jk}$ represents labels of observations where model j predicts 0 and model k predicts 1.

Let $T(\cdot)$ denote the matrix transpose operator, and $[M]$ the set $\{1, 2, \ldots, M\}$. We define the following matrices to facilitate the discussion on the application of various ML algorithms:

- The prediction matrix $\mathbf{Y}^S$. For a set of models $S \subset [M]$, define the *prediction matrix* $\mathbf{Y}^S$ as the matrix of predictions by the respective models. That is, the dimensions of $\mathbf{Y}^S$ are N by $|S|$, and $\mathbf{Y}_{i,k}^S$ is equal to $h^k(\mathbf{x}_i)$.

- Dimensionality reduction transformation $\text{DR}_{\mathcal{ALG}}(\mathbf{X})$. Dimensionality reduction (DR) algorithms transform data matrices into a lower-dimensional representation that preserves some structure in the data. Given a DR algorithm $\mathcal{ALG}$, we denote by $\text{DR}_{\mathcal{ALG}}(\mathbf{X})$ the 2-dimensional output of the algorithm when applied to the data matrix $\mathbf{X}$. Note that any matrix may be used as an input to the algorithm, including $\mathbf{Y}^S$.

- $\text{PW}_{\text{acc}}(\mathbf{Y}^S)$ is an operator that, given a matrix of predictions $\mathbf{Y}^S$, returns the similarities in the predictions between pairs of models. $\text{PW}_{\text{acc}}(\mathbf{Y}^S)_{k,l}$ is equal to the percentage of observations for which the predictions of models h^k and h^l agree: $\text{PW}_{\text{acc}}(\mathbf{Y}^S)_{k,l} = \frac{|\{i \in [N] : h^k(\mathbf{x}_i) = h^l(\mathbf{x}_i)\}|}{N}$.

3.2 Numerical Experiments

We next describe the datasets, training process, classification models, hyperparameters, and the DR methods used in our experiments.

Datasets. We use datasets from the UCI Machine Learning Repository (Asuncion & Newman, 2007) and FICO's Explainable Machine Learning Challenge (FICO, 2018). The description of the specific datasets used in this paper can be found in Appendix B.

Training process. We apply a standard model training and evaluation process of randomly partitioning each of the datasets into a 80% training set and a 20% test set. We evaluate the training error using five-fold CV on the training set and evaluate the test error on the test set.

Preprocessing methods. We tested different preprocessing methods such as the min-max scaler, standardization, and KBins discretizer (Pedregosa et al., 2011) and compared the CV and test accuracy of the models attained using each method (the results are presented and discussed in later sections). The missing values in our datasets only affected categorical variables. These were binarized, and additional binary features were added to indicate that the corresponding feature value is missing (this is commonly done when missing values are informative and dropping them may result in performance loss). Some of our models (see Appendix A) require binarized datasets, while others work with continuous and categorical variables. In general, different models and scaling methods work better for different datasets, and there is no single dominating scaling method. For simplicity, in training the 10 standard models (ADB, DTC, GNB, GP, KNN, LR, LSVC, MLP, RBFSVC, RF) and one interpretable model (EBM), we used unscaled datasets which mostly works well for these models. For training the other four interpretable models (BRCG, CORELS, OSDT, and SLIM) that require discretized datasets, we created a decision tree discretizer that for each feature individually trains a decision tree classifier to predict the label and use the resulting tree structure to determine the intervals of each feature which define the respective binary features (see also Chen et al., 2022; Galli, 2021; Niculescu-Mizil et al., 2009, for examples of this technique).

Hyperparameter tunning. We apply hypterparameter tuning using five-fold CV to determine the configuration with the best prediction accuracy for each model. We tune the typical hyperparameters of each model. The specific values used as hyperparameters for each model are described in Appendix C.

Performance measures. We collect information on the average and standard deviation of the CV accuracy, recall, precision, prediction, prediction confidence scores (e.g., probability for a certain class), and confusion matrix of each model.

Dimensionality reduction visualization methods. We experimented with the following DR methods: (Linear) principal component analysis (PCA) (Hotelling, 1933), (Nonlinear) isometric mapping (Isomap) (Tenenbaum, De Silva, & Langford, 2000), uniform manifold approximation and projection (UMAP) (McInnes, Healy, & Melville, 2018), and a DR method based on triplets (Trimap) (Amid & Warmuth, 2019). However, we found that PCA works better for our datasets, so we mainly used it for visualization. We note in passing that, in general, other DR methods may also be employed, including specialized

methods that were designed primarily for visualization (e.g., PaCMAP, Y. Wang, Huang, Rudin, & Shaposhnik, 2021). Even general purpose auto-encoders (e.g., the Variational Autoencoder of Kingma & Welling, 2013) can also be used for dimensionality reduction. Finally, we note that the DR methods were applied to the data matrices but also to some of the auxiliary matrices defined in the previous section.

With the above definitions in mind, we introduce four approaches to visualizing model comparison, with each presented in a separate section.

4. Interpretable Comparative Meta-models

We call the first approach interpretable comparative meta-models (ICMs). The general idea behind it is that for a pair of models, we can train a meta-model, which is simply a model trained on the joint predictions of the compared models. The interpretation of the meta-model can be used to infer when the compared models make similar or different predictions. This approach relies on supervised learning, and in particular, on interpretable ML models.

Formally, given two models h^j and h^k, we compute one of the joint labels (e.g., Y_{01}^{jk}), and train an interpretable ML model to predict it. Assuming that the meta-model is sufficiently accurate, it can then be used to explain when two models make a certain joint prediction (e.g., when model h^j predicts 0 and model h^k predicts 1). Note that one can train any type of model to learn any of the joint labels, or more generally, a Boolean function of the two labels.

In the following experiments that illustrate this approach, we trained 15 ML models using various datasets and computed the models' predictions both for the training set and on the test set. Since all the datasets have binary labels, the joint labels are denoted by 00, 01, 10, and 11. We then further defined binary classification tasks based on the joint labels. For example, we denote as "01-mismatch vs. rest" the classification task of characterizing the joint prediction 01. The term "mismatch" indicates that Model's 1 prediction does not match the prediction of Model 2, and that we are interested in understanding when the particular mismatch in which Model 1 predicts 0 and Model 2 predicts 1 occurs. Similarly, we define the classification tasks: "00-match vs. rest" (to identify when both models agree on predicting 0), "10-mismatch vs. rest" (to identify when Model 1 predicts 1 and Model 2 predicts 0), "11-match vs. rest" (to identify when both models predict 1), and "match vs. mismatch" (to understand when the two models predict the same). Based on these binary labels, we trained various interpretable ML models to predict them.

We provide examples of two interpretable models, CORELS and BRCG, and present two additional models, OSDT and EBM, in Appendix G.1.

Example 1: CORELS

CORELS (Certifiable Optimal Rule Lists; Angelino, Larus-Stone, Alabi, Seltzer, & Rudin, 2017) is an interpretable rule list model. We train it on the FICO dataset to compare the models BRCG and OSDT to explain the 01-mismatch. CORELS is used to explain when BRCG predicts 0 (low risk of customer defaulting on a loan) and OSDT predicts 1 (high risk). The rules describing the comparative model and its associated confusion matrix are provided in Figure 5.

	Predicted rest	Predicted 01-mismatch
Actual rest	1608	15
Actual 01-mismatch	0	352

RULELIST:
if *(External Risk Estimate* $\geq$ *71)* ***and*** *(Delinquency)* ***then predict*** *"01-mismatch"*
else predict *rest*

Figure 5: CORELS confusion matrix and rulelist for BRCG vs. OSDT "01-mismatch" on the FICO dataset. "Delinquency" indicates that the information about the number of months since the most recent delinquency is not missing.

The confusion matrix shows that the rule list model serves as an accurate proxy for the 01-mismatch prediction task. We can therefore infer that the model BRCG predicts a low-risk of defaulting and OSDT predicts a high-risk of defaulting whenever the feature ExternalRiskEstimate is greater than 71 and the feature MonthsSinceMostRecentDelinquency is positive. Since these features are interpretable (for more information, see FICO, 2018), this approximate explanation provides some insights about when the two models predict differently.

Similarly, a CORELS model can be trained to predict the 10-mismatch between the two models, as shown in Figure 6.

	Predicted rest	Predicted 10-mismatch
Actual rest	1692	0
Actual 10-mismatch	76	207

RULELIST:
if *(External Risk Estimate* $<$ *71)* ***and*** *(No Delinquencies)* ***then predict*** *"10-mismatch"*
else predict *rest*

Figure 6: CORELS confusion matrix and rulelist for BRCG vs. OSDT "10-mismatch" on the FICO dataset.

Example 2: BRCG

BRCG (Boolean Decision Rules via Column Generation; Dash, Günlük, & Wei, 2018) generates Boolean rules in either disjunctive normal form (DNF, OR-of-ANDs, equivalent to decision rule sets) or conjunctive normal form (CNF, AND-of-ORs) for classification tasks. The comparative model that explains the 01-mismatch between the models LSVC and SLIM for the FICO dataset and its corresponding confusion matrix can be seen in Figure 7.

We see that the meta-model is quite accurate, implying that it provides the approximate conditions under which the 01-mismatch occurs for the two models. Specifically, when any

	Predicted rest	Predicted 01-mismatch
Actual rest	1069	122
Actual 01-mismatch	75	709

Predict "01-mismatch" if ANY of the following rules are satisfied:
(External Risk Estimate ≤ 71) ***and*** *(Months Since Most Recent Inquiry excluding 7 days ≤ 23)*
(Average Months in File ≤ 49) ***and*** *(Months Since Most Recent Inquiry excluding 7 days ≤ 1)*

Figure 7: BRCG confusion matrix and rules for LSVC vs. SLIM "01-mismatch" on the FICO dataset.

of the specified conditions are met, LSVC predicts 0 and SLIM predicts 1, and in most other cases a different outcome occurs. Similarly, the following BRCG model in Figure 8 provides an explanation for 10-mismatch.

	Predicted rest	Predicted 10-mismatch
Actual rest	1827	20
Actual 10-mismatch	47	81

Predict "10-mismatch" if ANY of the following rules are satisfied:
(Percent Trades with Balance ≤ 74) ***and*** *(No Delinquencies)* ***and*** *(No Inquiries)*

Figure 8: BRCG confusion matrix and rulelist for LSVC vs. SLIM "10-mismatch" on the FICO dataset.

By identifying which model works better in different parts of the feature space, ICMs could potentially be used for stacking (Wolpert, 1992) (i.e., for combining multiple models together in a way that is superior to each of the models). This, however, might result in a combination of models that is overly complex and could even lead to overfitting if not carefully executed. We leave this direction for future work, as the scope of this paper is limited to model selection.

5. Visual Model Landscape

We call the second approach visual model landscape (VML). This type of visualization shows how multiple models relate to each other in a two-dimensional or hierarchical space. These methods utilize various unsupervised learning methods, such as dimensionality reduction and clustering, and plots such as scatter plots or tree visualizations. We provide two examples that convey the main ideas and delay the discussion on three additional examples to Appendix G.3.

Example 1: Comparing Models of Different Hypothesis Classes

Figure 9 shows multiple models trained on the Adult dataset. The green square marker Y represents the true labels, the orange circle markers represent standard models, and the blue diamond markers represent interpretable models. Beside each model is its test accuracy. Models tend to cluster according to their performance and the similarity of their predictions, and models with similar structure tend to be grouped together. For example, EBM and ADB are both boosting methods and they are located within the same cluster. Similarly, RBFSVC and KNN are both non-parametric classifiers that happen to be close to each other. Figure 9 also helps identifying interpretable models that could substitute standard models.

Figure 9 was created by computing $\mathbf{Y}^S$ (the matrix of models' predictions) and applying a DR method (PCA in this case) to the transposed prediction matrix (i.e., the matrix where rows correspond to models and columns to predictions, meaning that there are S rows and N columns in the matrix). The numeric values of the PCA transformation presented in Figure 9 can be found in Table 1 of Appendix G.2. Comparing with the heatmap in Figure 3 which presents the complete pairwise comparisons of model predictions, the VML represents the models graphically and uses Euclidean distance as a proxy for the similarity between models.

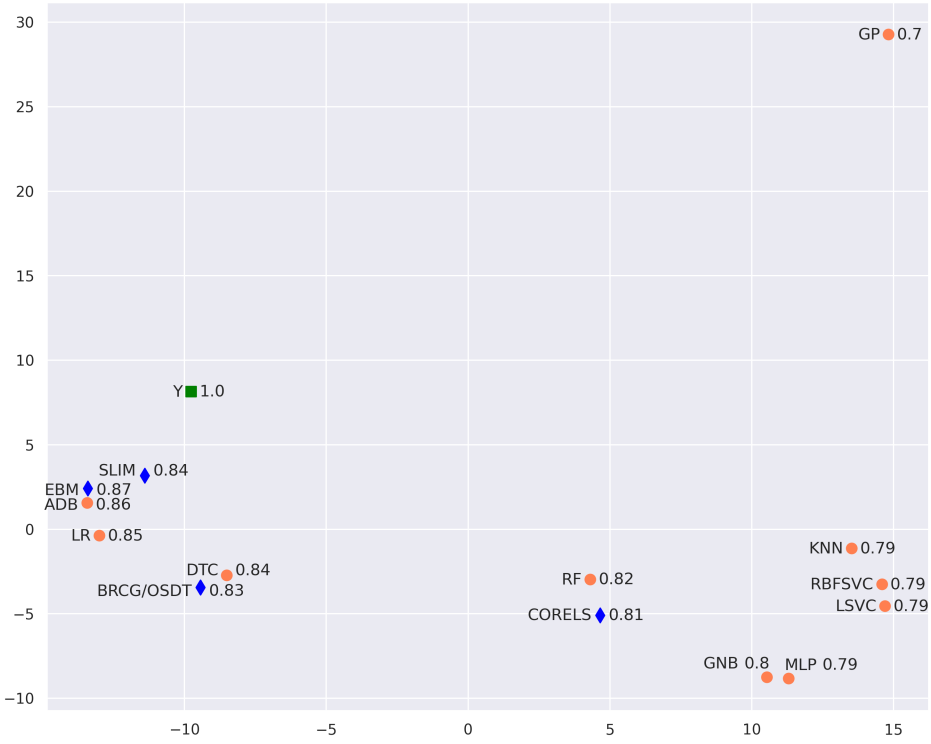


Figure 9: A VML scatter plot on a model obtained using PCA on the Adult dataset; the models BRCG and OSDT coincide.

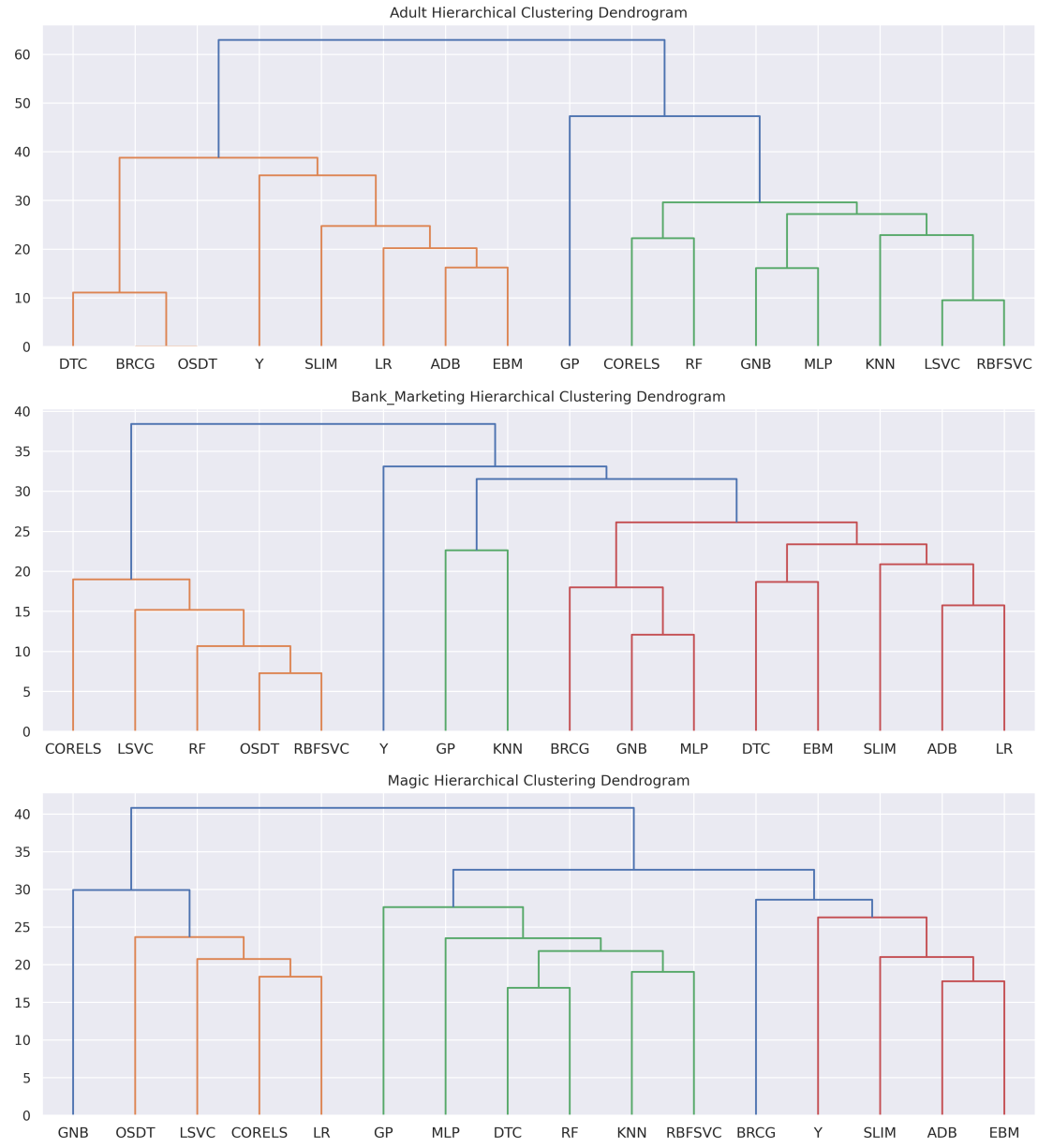


Figure 10: Dendrograms of VMLs on the Adult, Bank Marketing, and Magic datasets.

Example 2: VML Denrograms

We now present VML using hierarchies. Specifically, we apply hierarchical clustering to model predictions and display the resulting dendrograms.

Figure 10 shows a dendrogram created using the matrix $T(\mathbf{Y}^S)$. It compares the same 15 models and shows which models are most similar to each other and could therefore potentially be substituted for one another. For example, in the top subplot of Figure 10, we see that BRCG and OSDT (both interpretable logical models) are clustered together

with DTC (a decision tree generated using CART). It is also clear that SLIM (linear sparse model) is quite similar to LR, EBM, and even ADB. Of course, this depends on the specific dataset as well, which is why different hierarchies are generated from different datasets.

We briefly note that we can also cluster the models based on summary statistics, such as confusion matrices or discretized ROC curves, rather than on the complete prediction matrix (Appendix H). This could improve the running time and more importantly, generate different insights about the comparison between the overall performance of models according to certain metrics rather than the raw similarity of their predictions.

6. Visual Confusion Matrices

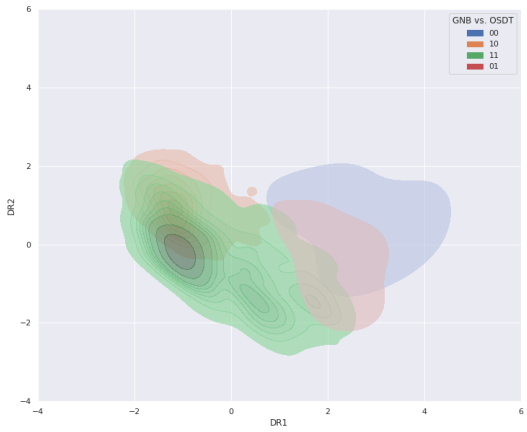
We denote by visual confusion matrices (VCMs) plots that, similarly to the confusion matrix, separate joint predictions. The plot itself could be a density plot (presented below), scatter plot, or biplot. All of these variants make use of dimensionality reduction on the data matrix, and the third approach is specific to PCA. For brevity, we present the density VCM plot, which conveys most of the ideas, and delay the discussion about the two variants to Appendix G.4.

Example 1: Density VCM Plots

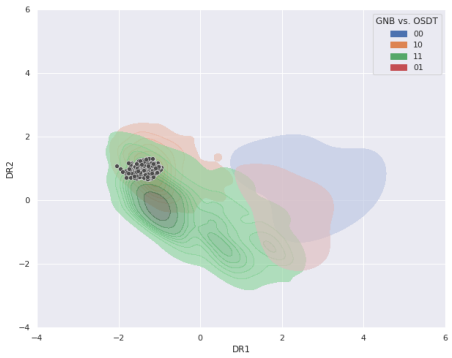
The density VCM plot extends the idea of confusion matrix by visualizing the density of data points aggregated according to the joint predictions (e.g., 00, 01, 10, and 11) of the compared models in the reduced-feature space. This is done by first applying a DR method (e.g., PCA) to the data matrix and then creating a density plot (e.g., KDE, the kernel density estimate; Parzen, 1962) to highlight the distribution of the joint predictions.

Figure 11 shows Density VCM plots created using PCA that compare two models (GNB and OSDT) based on the binary-labeled Magic dataset. In Figure 11a, we see four "clouds" corresponding to the joint predictions. For example, the bottom right red cloud corresponds to the joint prediction 01. For the group of observations that mostly lie in the 01 cloud, GNB tends to predict 0 and OSDT tends to predict 1. The implicit model selection tradeoff is presented in this plot (and similarly in other VCM plots) by the two clouds corresponding to the predictions 01 and 10. The particular meaning of each cloud provides intuition about the set of observations affected by the selection of one model over other.

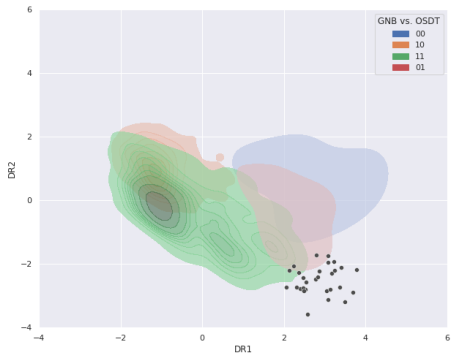
We can further use the plot to illustrate how this tradeoff affects clusters (i.e., similar subsets of observations). Clusters identification can be done manually, using standard clustering algorithms or by leveraging the interpretation of a particular DR method. This can be used to determine the tendency for making certain types of errors (e.g., 01 or 10) based on where the cluster is located. Two examples are shown in Figures 11b and 11c, which illustrate observations belonging to two clusters on top of the density plot. The cluster in Figure 11b lies at the orange and green regions where 10 and 11 overlap. Here, Model 1 (GNB) mostly predicts 1, and there is some ambiguity about the prediction of Model 2 (OSDT). On the other hand, the cluster shown in Figure 11c is located at a region close to where the clouds 11 and 01 overlap. In this case, Model 1 is ambiguous and Model 2 mostly predicts 1. This type of analysis improves our understanding of how model selection affects particular groups of observations.



(a) Four density plots based on the DR-transformed data matrix by each joint prediction.



(b) A cluster located at the 00 and 01 "clouds."



(c) A cluster located at the 01 and 11 "clouds."

Figure 11: (a) Illustration of a Density VCM plot obtained using PCA for the models GNB vs. OSDT on the Magic dataset; (b) the same Density VCM plot with observations belonging to a cluster where the predictions of Model 1 (GNB) are mostly consistent and the predictions of Model 2 (OSDT) are ambiguous; (c) the same Density VCM plot with observations belonging to a cluster where the predictions of Model 1 are ambiguous and the predictions of Model 2 are mostly consistent.

Note that the density plot is not perfect—clouds may overlap and observations may reside outside of the clouds. Nevertheless, it provides useful information about the relation between different models in the resolution of the entire dataset, particular clusters, or observations.

7. Visual Comparative Matrices

We next introduce the final approach, which we call visual comparative matrices (VCXs). While this approach is straightforward and has been part of existing analyses (see, e.g., J. Zhang et al., 2018), we discuss it here for completeness and due to its apparent effectiveness in comparing models. One example of this approach is given in Figure 3 in Section 1 where we compared different models in terms of the overall prediction similarity (additional examples of other datasets can be found in Appendix F).

The general idea of this approach is to simply create tables where one dimension corresponds to models that we wish to compare, the other dimension correspond to a property of interest, and the values in the tables represent an important metric. The tabular representation allows us to simultaneously compare multiple models and observe how the property affects the metric. Alternatively, both dimensions could correspond to models and the values represent quantified relations between them. We use heatmaps to highlight the relative magnitude of values in the table.

We next present an example in which we use VCXs for cluster analysis. We delay the discussion of another example that compares preprocessing methods to Appendix G.5.

Example 1: Cluster Analysis Using a VCX

Figure 12 shows how a VCX can be used for cluster analysis. It contains four heatmaps, each corresponding to different metrics: true positive rate (TPR), false positive rate (FPR), precision, and recall. In each heatmap, rows correspond to models, and columns correspond to 10 clusters obtained by K-Means clustering. The values in the tables correspond to the metric (i.e., subplot), model (i.e., row), and cluster (i.e., column). For example, we see that some clusters (e.g., Cluster 5) are difficult to classify for most models while other clusters (e.g., Cluster 2) are easier. The implicit model selection tradeoff is observed through the quality of prediction of models on different clusters. For example, in comparing KNN and LR, we observe that the recall is decreasing in cluster 4 while it is increasing for cluster 9. This allows us to compare multiple models and understand how subsets of the populations are impacted by the model selection decision.

Note that such an analysis relies on a meaningful application of a clustering algorithm (which are widely and effectively used in practice; see Punj and Stewart (1983) for examples in the marketing literature).

8. Case Study

In previous sections, we described different methods for model comparison and demonstrated them using various datasets. In this section, we focus on a particularly important application of predictive modeling routinely used to guide high-stake decisions. The Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) algorithm by Northpointe, Inc. is one such example. Such algorithms are commonly used in U.S. criminal sentencing to assess criminal defendant's likelihood of recidivism. Larson et al. (2016) raised the concern that the COMPAS algorithm is biased in evaluating certain populations to be at a higher risk than they actually are. Such bias could result in high-risk defendants being

Figure 12: Illustration of a VCX for cluster analysis based on the Magic dataset.

scored as low-risk, and vice versa. Understanding the implicit model selection tradeoff in this context is important for determining the impact of model selection decisions on people.

We apply our methods using the COMPAS dataset (Larson et al., 2016), which contains data collected between 2013–2014, which has information about criminal defendants in Broward County, Florida. The COMPAS algorithm was used to evaluate their "General Recidivism Risk" and "Violent Recidivism Risk" based on 137 features, as decile scores ranging from 1 to 10, with 10 indicating the highest risk level. These scores were considered in deciding about the release or detention of defendants. In general, a score between 1–4 is considered low risk, a score between 5–7 medium risk, and a score of 8–10 high risk (Northpointe, 2019).

To demonstrate the advantages of our methods, we train classification and regression models that predict risk, and then apply our methods to compare the resulting models and show how our methods highlight certain issues. Specifically, we train classification and regression models using the dataset provided by Coker, Rudin, and King (2021), which we

further preprocess by deleting and discretizing certain features. The dataset contains 6215 observations, each representing a defendant, and contains 14 features which include defendant's gender, age (18–20, 21–22, 23–25, 26–45, and > 45), a binary indicator of whether the most recent charge prior to filling the questionnaire is a felony, a binary indicator of juvenile felonies, binary indicator of juvenile misdemeanors, a binary indicator of juvenile crimes, and the number of prior charges (0, 1, 2–3, and > 3).

We note that we do not take any stance on this issue, but rather we demonstrate how our methodologies could potentially be used within this important context.

8.1 Comparing Classification Models

To compare classification models, we train models in which the target is a binary variable indicating recidivism within two years. Figure 13 shows the test accuracy of the models we used; we see that ADB, MLP, LSVC, LR, RF, KNN are the top-performing models, achieving test accuracies of 0.66 or higher. Figure 14 shows the agreement among models regarding the prediction; while some models appear to be similar, others differ by up to 5% of the test set population. We then train interpretable comparative meta-models that compare these top-performing models.

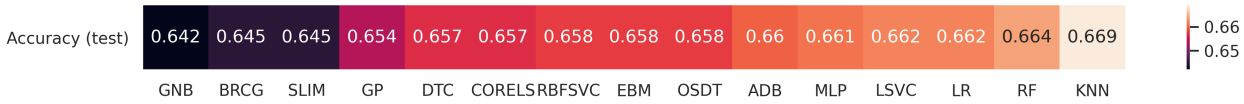


Figure 13: Performance of models based on test accuracy on the COMPAS dataset (predicting two year recidivism).

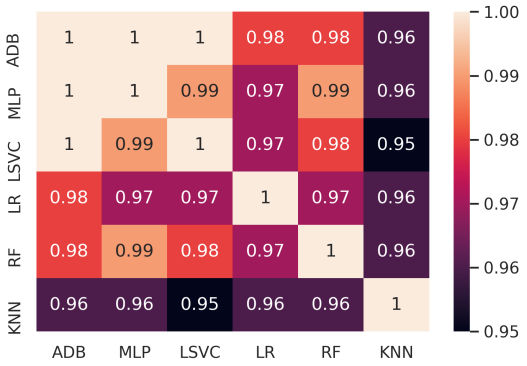


Figure 14: Model agreement level of top-performing models for the COMPAS dataset (predicting two year recidivism).

Figure 15 presents a CORELS rulelist for comparison of the pair ADB vs. KNN on classification of "01-mismatch vs. rest." The meta-model asserts that ADB tends to predict 0, while KNN predicts 1 whenever the most recent charge on file is a felony, the age is between 21 to 22, and the number of prior charges is 1.

	Predicted rest	Predicted 01-mismatch
Actual rest	1213	4
Actual 01-mismatch	8	19

RULELIST:
if *(c_charge_degree_F)* ***and*** *(age not between 21 and 22)* ***then predict*** *rest*
else if *(c_charge_degree_F)* ***and*** *(priors = 1)* ***then predict*** *"01-mismatch"*
else predict *rest*

Figure 15: CORELS confusion matrix and rulelist for ADB vs. KNN "01-mismatch" on the COMPAS dataset. The feature"c_charge_degree_F" indicates that the most recent charge on file is a felony.

Figure 16 shows a meta-model based on BRCG boolean rule for comparing the pair MLP vs. KNN on "01-mismatch vs. rest." We see that MLP tends predicts 0 while KNN predicts 1 in two cases. First, in the case of males, when the defendant's age is between 21 to 22, the most recent charge on file is a felony, and the number of prior charges is 1. Second, when the defendant is a female whose age is between 21 to 25, the most recent charge on file is *not* a felony, and the number of prior charges is 3 or less.

	Predicted rest	Predicted 01-mismatch
Actual rest	1217	2
Actual 01-mismatch	6	19

Predict "01-mismatch" if ANY of the following rules are satisfied:
'(age $\leq$ 22) ***and*** *(age $\geq$ 21)* ***and*** *(c_charge_degree_F)* ***and*** *(priors = 1)* ***and*** *(sex is Male)'*
'(age $\leq$ 25) ***and*** *(age $\geq$ 21)* ***and*** *(not c_charge_degree_F)* ***and*** *(priors $\leq$ 3)* ***and*** *(sex not Male)'*

Figure 16: BRCG confusion matrix and rules for MLP vs. KNN "01-mismatch" on the COMPAS dataset.

Finally, Figure 17 presents an optimal sparse decision tree meta-model for comparing the models ADB with LR, with the goal of understanding 10-mismatch in prediction. The leaf nodes of the tree show the predictions, the number of samples, and the training accuracy in each leaf. There are two cases in which mismatches occur. The first case is when the defendant is younger than 45 but not 23–25, his or her number of prior charges is different than 2 and 3, and the most recent charge on file is *not* a felony. The second case involves females older than 45 with more than three prior charges, and whose most recent charge on file is *not* a felony. The meta-model also identifies the accuracy of the explanation with 69% in the first case and 96% in the second case.

	Predicted rest	Predicted 10-mismatch
Actual rest	1210	5
Actual 10-mismatch	8	21

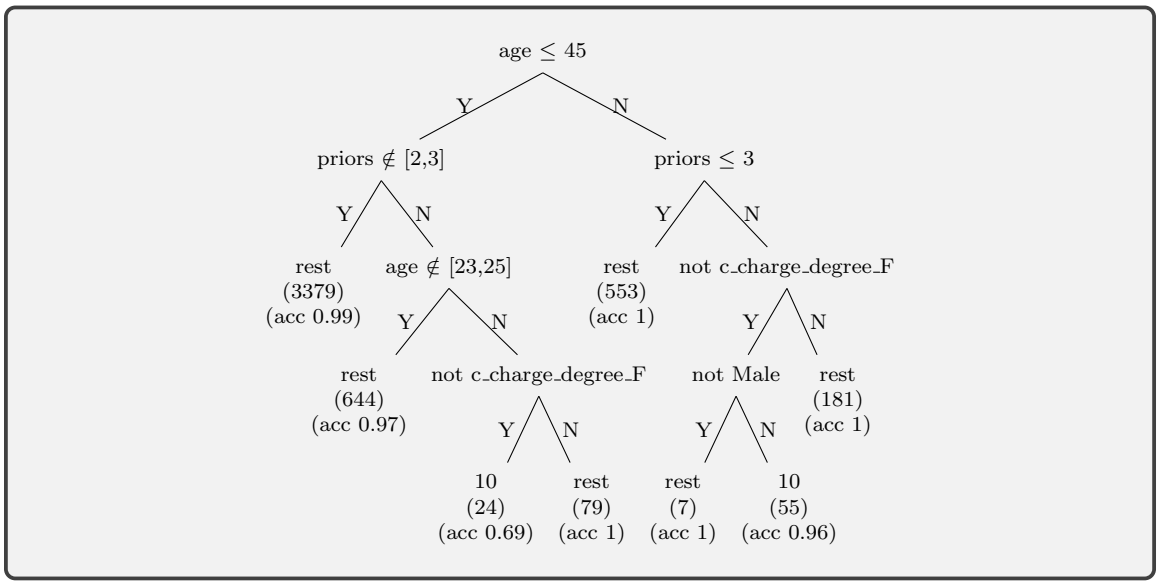


Figure 17: OSDT decision tree for ADB vs. LR "10-mismatch" on the COMPAS dataset.

8.2 Comparing Regression Models

Next we generalize our methods to compare regression models. To illustrate how to interpret the implicit model selection tradeoff for regression problems, we use regression models to predict the general recidivism risk decile scores of defendants that range from 1–10, where 10 represents the highest risk compared to other offenders in the norm group. We then define classification problems on which our methods can be directly applied. This is done by discretizing the continuous prediction into classes (Low risk (1–4), Medium risk (5–7), and High risk (8–10) based on the COMPAS Practitioner's Guide Northpointe, 2019). We briefly note that an alternative way could be to define an indicator for when the difference in the score prediction exceeds a certain threshold α. For example, we could define the label $\mathbf{Y}_{HL}^{jk}$ as $\mathbb{1}[y_i^j - y_i^k > \alpha]$, which could be used to decide whether model j's prediction is significantly higher than that of model k's.

We used the following models in this experiment: AdaBoost Regressor (ADBR), Bayesian Ridge (BR), Decision Tree Regressor (DTR), Gaussian Process Regressor (GPR), KNeighbors Regressor (KNR), Lasso, Linear Regression (LinearR), MLP Regressor (MLPR), Random Forest Regressor (RFR), Ridge, Epsilon-Support Vector Regression (SVR), Explainable Boosting Regressor (EBMR), and Supersparse Linear Integer Model Regressor (SLIMR). The hyperparameters are described in Appendix D.

The MSE scores of the regression models are shown in Figure 18. We see that LinearR, BR, Ridge, Lasso, EBMR, and MLPR achieve the smallest mean squared error. The agreement in the models' discretized predictions among the top-performing models is displayed in Figure 19. Some models are very similar in their discretized predictions (e.g.,

LinearR, BR, Ridge, and Lasso) while others (EBMR and MLPR) predict differently from other models, even though their MSEs are quite similar. Figure 36 in Appendix I shows the average predicted labels versus the actual ones for each of the models—we can see that the trained models are biased towards the average prediction. This may results from our limited access to the complete dataset (we use less than 20 features in comparison to the 137 features available in the complete dataset).

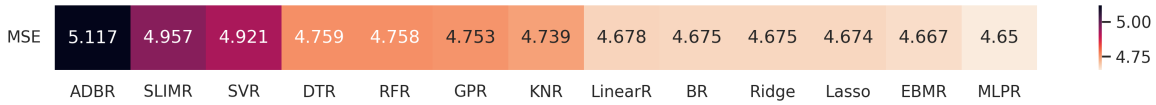


Figure 18: Performance of models based on MSE on the COMPAS dataset (regression).

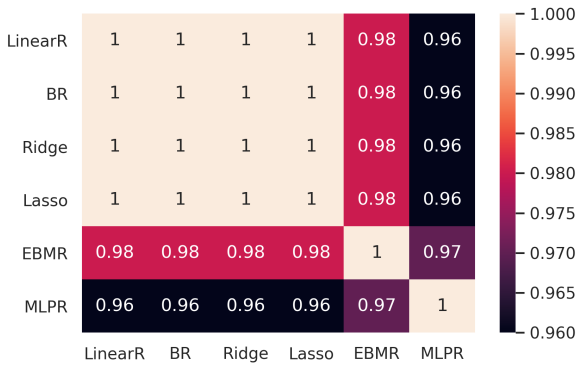


Figure 19: Model agreement levels where predictions are rounded to low, medium, and high risk on the COMPAS dataset (regression).

To train interpretable meta-models, we define the joint predictions to classify when the models agree ("LL," "MM," and "HH") or disagree ("LM," "ML," "MH," "HM," "LH," and "HL"). We did not observe the joint predictions LH and HL, which indicates that none of the models predicts too differently from the others. Therefore, understanding the implicit model selection tradeoff can be done by explaining the joint predictions "LM vs. rest," "ML vs. rest," "MH vs. rest," and "HM vs. rest."

Figure 20 shows a CORELS rulelist for comparing the models Ridge vs. MLPR on the joint prediction "LM-mismatch vs. rest." We see that Ridge (as well as LinearR and Lasso) tends to predict low risk, while MLPR predicts medium risk for people aged 23–25 with two or three prior charges when the most recent charge on file is a felony.

Figure 21 shows an example for the BRCG boolean rule as a meta-model for comparing the models BR and MLPR, to explain the LM-mismatch. BRCG predicts "LM-mismatch" in two cases: first, if the defendant's age is between 23 and 25, the most recent charge on file is a felony, the number of prior charges is less than three, and the defendant is a male; second, if the defendant is older than 26, the most recent charge on file is a felony, there are no juvenile misdemeanors, the number of prior charges is less than 3, and the defendant is male.

	Predicted rest	Predicted LM-mismatch
Actual rest	1215	5
Actual LM-mismatch	6	18

RULELIST:
if *(not c_charge_degree_F)* ***and*** *(age not between 21 and 22)* ***then predict*** *rest*
else if *(age between 23 and 25)* ***and*** *(priors = 2 or 3)* ***then predict*** *"LM-mismatch"*
else predict *rest*

Figure 20: CORELS confusion matrix and rulelist for Ridge vs. MLPR "LM-mismatch" on the COMPAS dataset (regression). "c_charge_degree_F" indicates whether the most recent charge before COMPAS score calculation is a felony.

	Predicted rest	Predicted LM-mismatch
Actual rest	1218	2
Actual LM-mismatch	6	18

Predict "LM-mismatch" if ANY of the following rules are satisfied:
'(age $\leq$ 25) ***and*** *(age $\geq$ 23)* ***and*** *(c_charge_degree_F)* ***and*** *(priors $\leq$ 3)* ***and*** *(sex is Male)'*
'(age $\geq$ 26) ***and*** *(c_charge_degree_F)* ***and*** *(juvenile misdemeanors = 0)* ***and*** *(priors $\leq$ 3)* ***and*** *(sex is Male)'*

Figure 21: BRCG confusion matrix and rules for BR vs. MLPR "LM-mismatch" on the COMPAS dataset (regression).

Finally, Figure 22 presents an optimal sparse decision tree for comparing the models LinearR and MLPR, in order to identify HM-mismatches. There are multiple cases of discrepancy, each characterized by an estimated accuracy level. For example, if the defendant has a record of juvenile misdemeanor and crime, their age is 18–20, and the number of prior charges is greater than three, then there is a chance of 78% that LinearR predicts high risk while MLPR predicts medium risk.

9. Concluding Remarks

Summary. The paper studies the implicit model selection tradeoff and proposes an array of methods to unveil it. To our knowledge, this is the first work that develops such methods for the interpretable comparison of competing models. The methods we explored rely on visualization techniques applied to outputs of supervised and unsupervised ML algorithms. We illustrate how these methods work and make them accessible by providing a simple Python interface (Appendix E).

Best practices. The paper proposes a host of methods for model comparison. To facilitate their utilization, we suggest using visual comparative matrices for preprocessing and

	Predicted rest	Predicted HM-mismatch
Actual rest	1219	2
Actual HM-mismatch	0	23

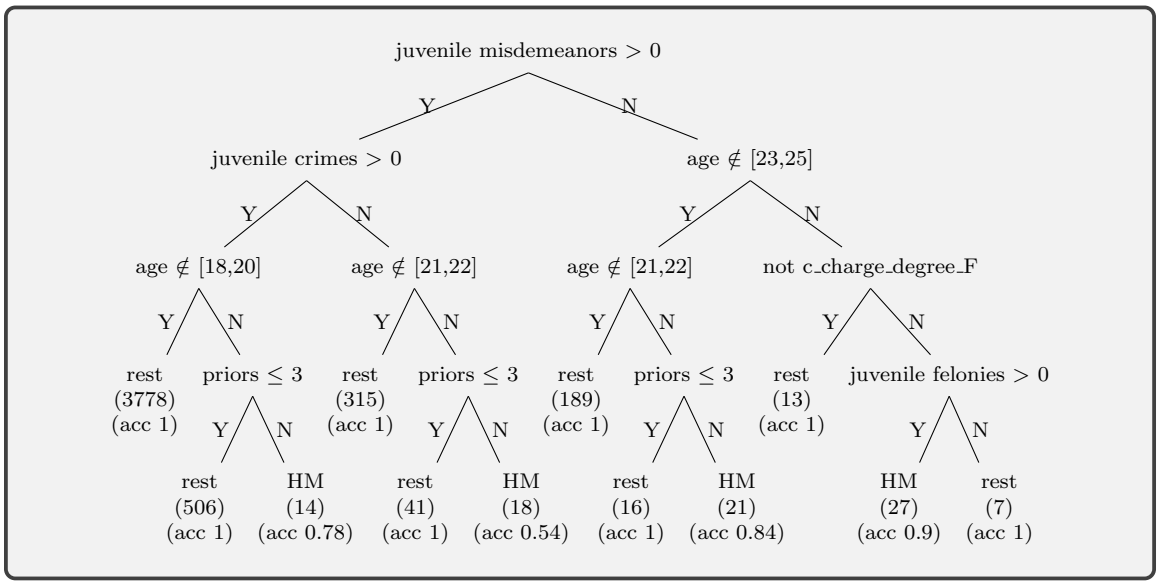


Figure 22: OSDT decision tree for LinearR vs. MLPR "HM-mismatch" on the COMPAS dataset (regression).

feature engineering, utilizing visual model landscape plots as part of the preliminary analysis to identify and eliminate candidate models; and using visual confusion matrices and comparative meta-models for final tuning and selection. The latter can also be used for validation, that is, to compare the performance of models that are based on the train and test data and to identify generalization issues.

Future work. We believe that this work could potentially lead to further research on the implicit model selection tradeoff. We describe a few broad directions that seem promising.

First, it would be interesting to develop methods that address some of the limitations of our work. For example, our sampling-based approach means that the resulting visualizations and interpretable meta-model could suffer from bias and variance. An alternative approach could take advantage of the exact structure of the compared models; however, this requires knowledge about the compared models and may only allow for the comparison of specific models. Another inevitable limitation is the reliance on DR methods that do not fully preserve structure in high dimensions and on whose interpretation some of our approaches are based. We also note that it is not always possible to succinctly describe the comparison between models due to their complex functional form. Finally, while our focus has been on structured data, it would be interesting to extend our methods to allow for a comparison of models on unstructured data, such as images or text. These, however, might require alternative methods. For example, to create a comparative meta-model, one may need to resort to more complex interpretable neural network (see, for example, Chen et al., 2018).

Second, our work could be used as a new method of interpreting models that is based on model comparison. That is, one can describe a potentially complex model using an interpretable model that approximates it, and a meta-model that describes the difference between the original model and the approximation. Such an explanation, could consist of multiple interpretable models and meta-models.

Finally, we note that information theory could potentially serve as a source of inspiration for additional ways to visualize the implicit model selection tradeoff. For example, mutual information could be used as a means to compute similarity between models (as we briefly illustrated in Figure 24 in Appendix F), which could, for example, be used to form dimension-reduced plots (such as VML scatter plots and dendrograms).

We leave these directions for future research.

Acknowledgments

We would like to thank the review team for their efficient handling of the paper, as well as for their constructive and insightful feedback which helped improve the work.

Appendix A. Classification Models

The 15 classification models used in our numerical experiments involve 10 standard models and five interpretable models. The standard models are: K-nearest Neighbors Classifier (KNN), Linear Support Vector Classification (LSVC), Radial Basis Function kernel SVC (RBFSVC), Gaussian Process Classifier (GP), Decision Tree Classifier (DTC), Random Forest Classifier (RF), Multi-layer Perceptron classifier (MLP), AdaBoost Classifier (ADB), Gaussian Naive Bayes (GNB), and Logistic Regression (LR) (Pedregosa et al., 2011). The interpretable models are: Boolean Decision Rules via Column Generation (BRCG)(Dash et al., 2018), CORELS (Angelino et al., 2017), ExplainableBoostingClassifier (EBM)(Nori, Jenkins, Koch, & Caruana, 2019), Optimal Sparse Decision Tree (OSDT)(Hu, Rudin, & Seltzer, 2019), and Supersparse Linear Integer Model (SLIM)(Singh, Nasseri, Tan, Tang, & Yu, 2021; Ustun & Rudin, 2016).

The models BRCG, CORELS, and OSDT only work with binary datasets. Therefore, we binarized the datasets using DTs by training a DT model for each feature to predict the output variable. We then used the Boolean condition of each branching node of the tree as a binary feature. We limited the parameter "min samples leaf" to 1/20 of the dataset size and limited the parameter "max leaf nodes" to 10 to restrict the total number of features to prevent overfitting.

Appendix B. Datasets

- Adult—Kohavi and Becker (1994); N=48,842, P=14; the binary labels in the data indicate whether an individual's income exceeds \$50K per year using information provided from the census data. The percentage of observations of each label: 75.9% labeled 0 (income $>$ 50K), and 24.1% labeled 1 (income $\leq 50K$).

- Bank Marketing—S. Moro and Rita (2014); N=41,188, P=17; the binary labels in the data indicate whether a customer contacted in the direct marketing campaigns would subscribe to a term deposit. The percentage of observations of each label: 88.7% labeled 0 (yes), and 11.3% labeled 1 (no).
- Magic—Bock (1998); N=19020, P=11; the binary labels in the data indicate whether Cherenkov photons are caused by primary gammas (signal) or hadronic showers (background). The percentage of observations of each label: 64.8% labeled 0 (class g), and 35.2% labeled 1 (class h).
- Mushroom—Schlimmer (1981); N=8,124, P=22; the binary labels in the data indicate whether mushrooms are edible or poisonous (or unknown). The percentage of observations of each label: 51.8% labeled 0 (class e), and 48.2% labeled 1 (class g).
- Musk (Version 2)—Chapman (1994); N=6,598, P=168; the binary labels in the data indicate whether molecules are musks or non-musks. The percentage of observations of each label: 84.6% labeled 0, and 15.4% labeled 1.
- German Credit data—Hofmann (1994); N=1000, P=24; the binary labels in the data indicate whether people have bad or good credit risks based on a number of attributes. The percentage of observations of each label: 70% labeled 0, and 30% labeled 1.
- FICO continuous data—FICO (2018); N=9,871, P=35; the binary labels in the data indicate whether customers were late on payment of HELOC loans. The percentage of observations of each label: 52.0% labeled 0, and 48.0% labeled 1.
- FICO binary data—FICO (2018); N=9,871, P=109; this is a binarized version of the FICO continuous data based on Rudin and Shaposhnik (2023).

Appendix C. Tuning Parameters for Section 4

Standard models

Model	**Hyperparameters**
ADB	n estimators: [10, 30, 50], algorithm: [SAMME, SAMME.R]
DTC	criterion: [gini, entropy], max depth: [2, 3, 4, 5]
GNB	var smoothing: [1e-5, 1e-4, 1e-3]
GP	random state: [0]
KNN	n neighbors: [1, 2, 3, 4, 5], weights: [uniform, distance], algorithm: [auto], metric: [euclidean, manhattan]
LR	penalty: [l1, l2], tol: [1e-5, 1e-4, 1e-3], solver: [newton-cg, lbfgs, liblinear, sag, saga]
LSVC	loss: [hinge, squared hinge], tol: [1e-5, 1e-4, 1e-3]
MLP	solver: [sgd, adam], alpha: [0.0001, 0.001, 0.01], activation: [logistic, relu], learning rate: [constant, invscaling, adaptive]
RBFSVC	kernel: [poly, rbf], degree: [2, 3, 4, 5], gamma: [scale, auto]
RF	n estimators: [10, 30, 50], criterion: [gini, entropy], max depth: [2, 3, 4]

Interpretable models

Model	Datasets	Hyperparameters
BRCG	Sections 5, 6, and 7 for the Adult, Bank Marketing, Magic, Mushroom, and Musk2 datasets	lambda0: [0.0005, 0.001, 0.005], lambda1: [0.0005, 0.001, 0.005], CNF: [True], iterMax: [100, 500, 1000], K: [10], D: [10], B: [5], eps: [1e-6, 1e-3]
BRCG	Sections 5, 6, and 7 for the German Credit and FICO datasets	lambda0: [1e-05, 1e-04, 1e-03], lambda1: [1e-05, 1e-04, 1e-03], CNF: [True], iterMax: [100], K: [10], D: [10], B: [5], eps: [1e-3, 1e-2]
BRCG	Section 4 for all datasets	lambda0: [0.0001, 0.0005, 0.001, 0.005, 0.01], lambda1: [0.0001, 0.0005, 0.001, 0.005, 0.01], CNF: [True], iterMax: [100, 200, 300], K: [10], D: [10], B: [5], eps: [1e-6, 1e-3]
CORELS	Sections 4, 5, 6, and 7 for the Adult, Bank Marketing, Magic, Mushroom, and Musk2 datasets	c: [0.025, 0.05, 0.075], n iter: [10000], map type: [prefix], policy: [bfs], verbosity: [[rulelist]], ablation: [0], max card: [2], min support: [0.01, 0.025]
CORELS	Sections 4, 5, 6, and 7 for the German Credit and FICO datasets	c: [1e-6, 1e-5, 5e-5, 1e-4, 5e-4, 1e-3], n iter: [10000], map type: [prefix], policy: [bfs], verbosity: [[rulelist]], ablation: [0], max card: [2], min support: [1e-6, 1e-5, 5e-5, 1e-4, 5e-4]
EBM	Sections 4, 5, 6, and 7 for all datasets	inner bags: [50, 100], outer bags: [50, 100], learning rate: [0.1], n jobs: [-2], binning: [quantile], interactions: [2], max rounds: [100], min samples leaf: [2], max leaves: [7]
OSDT	Sections 4, 5, 6, and 7 for the Adult, Bank Marketing, Magic, Mushroom, and Musk2 datasets	lamb: [0.005, 0.01, 0.05], prior metric : [curiosity], MAXDEPTH: [2, 3, 4], MAX NLEAVES: [3, 5, 7], timelimit: [1800], niter: [100]
OSDT	Sections 4, 5, 6, and 7 for the German Credit and FICO datasets	lamb: [0.0005, 0.001, 0.0025, 0.005, 0.01, 0.025], prior metric: [curiosity], MAXDEPTH: [2, 3, 4], MAX NLEAVES: [3, 5, 7], timelimit: [1800], niter: [100, 200, 300]
SLIM	Sections 4, 5, 6, and 7 for all datasets	alpha: [1e-09, 5e-09, 1e-08, 5e-08, 1e-07, 5e-07, 1e-06, 5e-06, 1e-05, 5e-05, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 1.25, 1.5, 1.75, 2, 2.25]

Appendix D. Tuning Parameters for the Section 8 Case study

D.1 Classification Problem

Standard models

Model	Hyperparameters
ADB	n estimators: [10, 15, 20, 25, 30], algorithm: [SAMME, SAMME.R]
DTC	criterion: [gini, entropy], max depth: [3, 4, 5, 6, 7, 8]
GNB	var smoothing: [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8]
GP	random state: [0]
KNN	n neighbors: [10, 20, 30, 40, 50], weights: [uniform, distance], algorithm: [auto], metric: [euclidean, manhattan]
LR	penalty: [l1, l2], tol: [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8], solver: [newton-cg, lbfgs, liblinear, sag, saga]
LSVC	loss: [hinge, squared hinge], tol: [1e-12, 1e-9, 1e-6, 1e-3]
MLP	solver: [sgd, adam], alpha: [1e-6, 1e-5, 1e-4, 1e-3, 1e-2], activation: [logistic, relu], learning rate: [constant, invscaling, adaptive]
RBFSVC	kernel: [poly, rbf], degree: [2, 3, 4, 5], gamma: [scale, auto]
RF	n estimators: [10, 20, 30, 40, 50], criterion: [gini, entropy], max depth: [3, 4, 5, 6, 7, 8]

Interpretable models

Model	Hyperparameters
BRCG	lambda0: [0.0001, 0.001, 0.01], lambda1: [0.0001, 0.001, 0.01], CNF: [False], iterMax: [100], K: [5, 10], D: [5, 10], B: [5, 10], eps: [1e-6, 1e-4, 1e-2]
CORELS	c: [1e-12, 1e-09, 1e-06, 1e-03], n iter: [10000], map type: [prefix], policy: [bfs], verbosity: [[rulelist]], ablation: [0], max card: [2], min support: [1e-12, 1e-09, 1e-06, 1e-03]
EBM	inner bags: [50, 100], outer bags: [50, 100], learning rate: [0.001, 0.01, 0.1], n jobs: [-2], binning: [uniform, quantile, quantile humanized], interactions: [1, 2], max rounds: [500], min samples leaf: [2], max leaves: [3, 5, 7]
OSDT	lamb: [1e-09, 1e-06, 1e-03], prior metric : [curiosity], MAXDEPTH: [3, 4], MAX NLEAVES: [3, 4, 5], timelimit: [1800], niter: [100, 200, 300]
SLIM	alpha: [1e-06, 5e-06, 1e-05, 5e-05, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]

D.2 Regression Problem

The new standard models we introduce here are: AdaBoost Regressor (ADBR), Bayesian Ridge (BR), Decision Tree Regressor(DTR), Gaussian Process Regressor (GPR), KNeighbors Regressor (KNR), Lasso, Linear Regression (LinearR), MLP Regressor (MLPR)), Random Forest Regressor (RFR), Ridge, and Epsilon-Support Vector Regression (SVR).

The new interpretable models are: ExplainableBoostingRegressor (EBMR) and Supersparse Linear Integer Model Regressor (SLIMR).

Standard models

Model	Hyperparameters
ADBR	n estimators: [10, 15, 20, 25], loss: [linear, square, exponential], learning rate: [0.01, 0.1, 0.2, 0.3, 0.4, 0.5]
BR	n iter: [50, 100], tol: [1e-3, 1e-2, 0.1, 0.2, 0.3], alpha1: [1e-10, 1e-8, 1e-6, 1e-4, 1e-2, 1e-1], alpha2: [1e-10, 1e-8, 1e-6, 1e-4, 1e-2, 1e-1], lambda1: [1e-10, 1e-8, 1e-6, 1e-4, 1e-2, 1e-1], lambda2: [1e-10, 1e-8, 1e-6, 1e-4, 1e-2, 1e-1]
DTR	criterion: [squared error, friedman mse, absolute error, poisson], splitter: [best, random], max depth: [3, 4, 5, 6, 7]
GPR	random state: [0]
KNR	n neighbors: [30, 50, 70, 90], weights: [uniform, distance], algorithm: [auto], metric: [euclidean, manhattan]
Lasso	alpha: [1e-06, 1e-05, 1e-04, 1e-03, 1e-02, 1e-01, 1, 1e01, 1e02, 1e03, 1e04, 1e05, 1e06], fit intercept: [True, False], normalize: [True, False], max iter:[1000, 10000], tol: [1e-08, 1e-07, 1e-06, 1e-05, 1e-04, 1e-03, 1e-02, 1e-01], random state: [0], selection: [cyclic, random]
LinearR	fit intercept: [True, False], normalize: [True, False]
MLPR	hidden layer sizes: [10, 20, 30, 40, 50, 60], solver: [sgd, adam], alpha: [0.00001, 0.0001, 0.001, 0.01], activation: [logistic, relu], learning rate: [constant, invscaling, adaptive]
RFR	n estimators: [50, 100, 150], criterion: [squared error, absolute error, poisson], max depth: [3, 6, 9, 12, 15, 18]
Ridge	alpha: [1e-06, 1e-05, 1e-04, 1e-03, 1e-02, 1e-01, 1, 1e01, 1e02, 1e03, 1e04, 1e05, 1e06], fit intercept: [True, False], normalize: [True, False], max iter:[None], tol: [1e-12, 1e-11, 1e-10, 1e-09, 1e-08, 1e-07, 1e-06, 1e-05, 1e-04, 1e-03, 1e-02, 1e-01], random state: [0], solver: [auto]
SVR	kernel: [poly, rbf], degree: [1, 2, 3, 4, 5], gamma: [scale, auto]

Interpretable models

Model	Hyperparameters
EBMR	inner bags: [50, 100], outer bags: [50, 100], learning rate: [0.001, 0.01, 0.1], n jobs: [-2], binning: [uniform, quantile, quantile humanized], interactions: [1, 2], max rounds: [500], min samples leaf: [2], max leaves: [3, 5, 7]
SLIMR	alpha: [1e-06, 5e-06, 1e-05, 5e-05, 0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
BRCG	lambda0: [0.0001, 0.001, 0.01], lambda1: [0.0001, 0.001, 0.01], CNF: [False], iterMax: [100], K: [5, 10], D: [5, 10], B: [5, 10], eps: [1e-6, 1e-4, 1e-2]
CORELS	c: [1e-12, 1e-09, 1e-06, 1e-03], n iter: [10000], map type: [prefix], policy: [bfs], verbosity: [[rulelist]], ablation: [0], max card: [2], min support: [1e-12, 1e-09, 1e-06, 1e-03]
OSDT	lamb: [1e-09, 1e-06, 1e-03], prior metric : [curiosity], MAXDEPTH: [3, 4], MAX NLEAVES: [3, 4, 5], timelimit: [1800], niter: [100, 200, 300]

Appendix E. Python Programming Interface

We create a simple interface that can be used to generate plots similar to those generated in this paper. The function headers and descriptions of their parameters are listed below. The code is available at https://github.com/zhesimon/ComparativeMetaModels.

MC_heatmap_of_prediction(Y_list, export_file = False, filename = None):

- Y_list: a list of tuples that consists of model names and model predictions. The first tuple contains the true labels, while the subsequent tuples contain model predictions.
- export_file: a Boolean parameter indicating whether to export the figure to a file or simply display to screen
- filename: output file name

MC_scatterplot_prediction(Y_list, color_indices = None, colors = None, export_file = False, filename = None):

- Y_list: a list of tuples that consists of model names, model predictions, and test accuracies
- color_indices: a list of indices for the colors
- colors: a list of colors
- export_file: a Boolean parameter indicating whether to export the figure to a file or simply display to screen
- filename: output file name

MC_scatterplot_confusion(Y_list, color_indices = None, colors = None, export_file = False, filename = None):

- Y_list: a list of tuples that consists of model names, model predictions, and test accuracies. The first tuple contains the true labels, while the subsequent tuples contain model predictions.
- color_indices: a list of indices for the colors
- colors: a list of colors
- export_file: a Boolean parameter indicating whether to export the figure to a file or simply display to screen
- filename: output file name

MC_hierarchical_tree(Y_list, export_file = False, filename = None):

- Y_list: a list of tuples that consists of model names and model predictions
- export_file: a Boolean parameter indicating whether to export the figure to a file or simply display to screen
- filename: output file name

MC_visual_density_plot(X_DR, Y_list, export_file = False, filename = None):

- X_DR: a dimensionality reduction transformation of data matrix X in 2D represented by a pandas dataframe of shape (n samples, 2)
- Y_list: a list of tuples that consists of model names and model predictions
- export_file: a Boolean parameter indicating whether to export the figure to a file or simply display to screen
- filename: output file name

MC_cluster_analysis(X, Y_list, models, n_cl = 10 , export_file = False, filename = None):

- X: a data matrix represented by a pandas dataframe of shape (n samples, p features)
- Y_list: a list of tuples that consists of model names and model predictions. The first tuple contains the true labels, while the subsequent tuples contain model predictions.
- models: a list that consists of model names in Y_list
- n_cl: number of clusters (default is 10)
- export_file: a Boolean parameter indicating whether to export the figure to a file or simply display to screen
- filename: output file name

MC_simple_dr_comparison(X_DR, Y_list, export_file = False, filename = None):

- X_DR: a dimensionality reduction transformation of data matrix X in 2D represented by a pandas dataframe of shape (n samples, 2)
- Y_list: a list of tuples that consists of model names and model predictions. The first tuple contains the true labels, while the subsequent tuples contain model predictions.
- export_file: a Boolean parameter indicating whether to export the figure to a file or simply display to screen
- filename: output file name

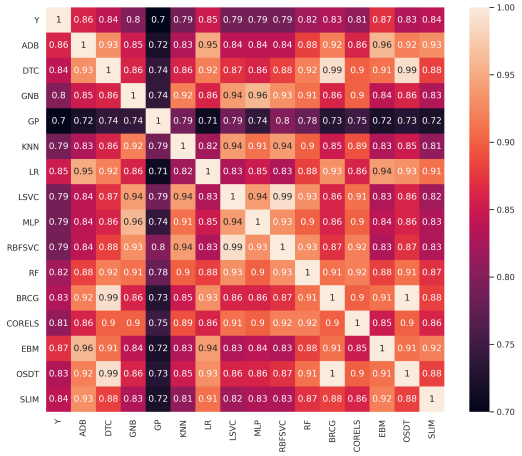
MC_visual_confusion_matrix(X_DR, Y_list, export_file = False, filename = None):

- X_DR: a dimensionality reduction transformation of data matrix X in 2D represented by a pandas dataframe of shape (n samples, 2)
- Y_list: a list of tuples that consist of model name and model predictions
- export_file: a Boolean parameter indicating whether to export the figure to a file or simply display to screen
- filename: output file name

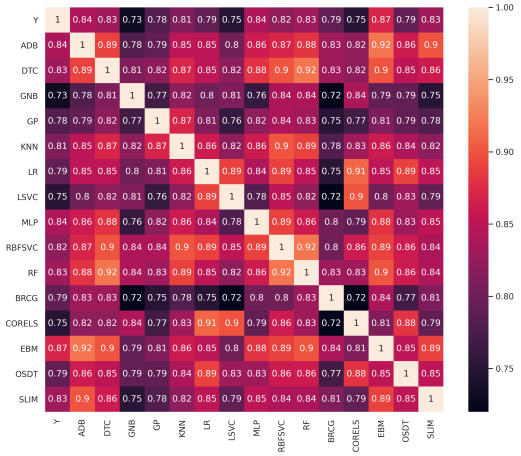
MC_biplot(X_pca, pca_model, Y_list, features, export_file = False, filename = None):

- X_pca: a PCA dimensionality reduction transformation of data matrix X in 2D represented by a pandas dataframe of shape (n samples, 2)
- pca_model: model of PCA
- Y_list: a list of tuples that consists of model names and model predictions
- features: features of data matrix X
- export_file: a Boolean parameter indicating whether to export the figure to a file or simply display to screen
- filename: output file name

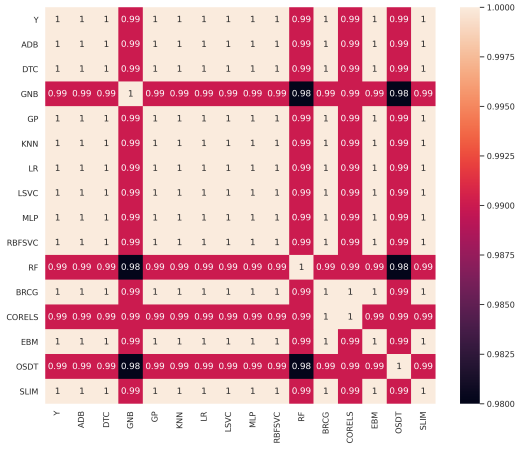
Appendix F. Heatmaps of Model Comparisons



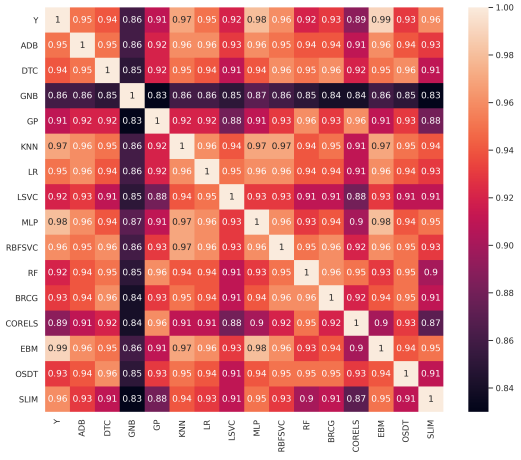
(a) Adult dataset.



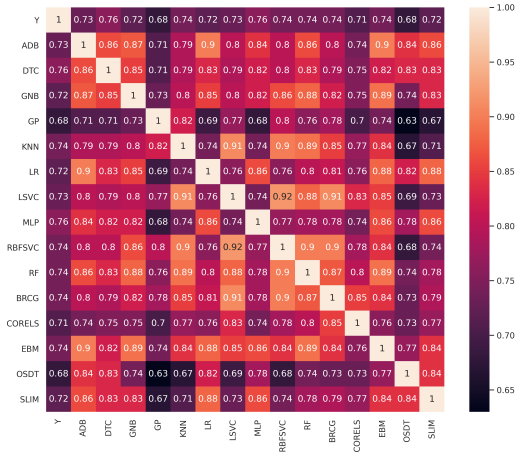
(b) Magic dataset.



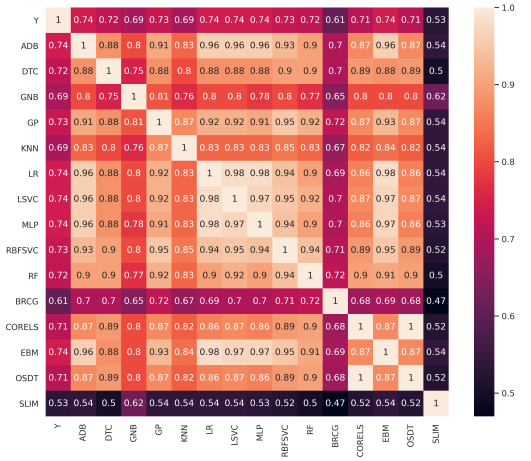
(c) Mushroom dataset.



(d) Musk2 dataset.



(e) German Credit dataset.



(f) FICO (binary) dataset.

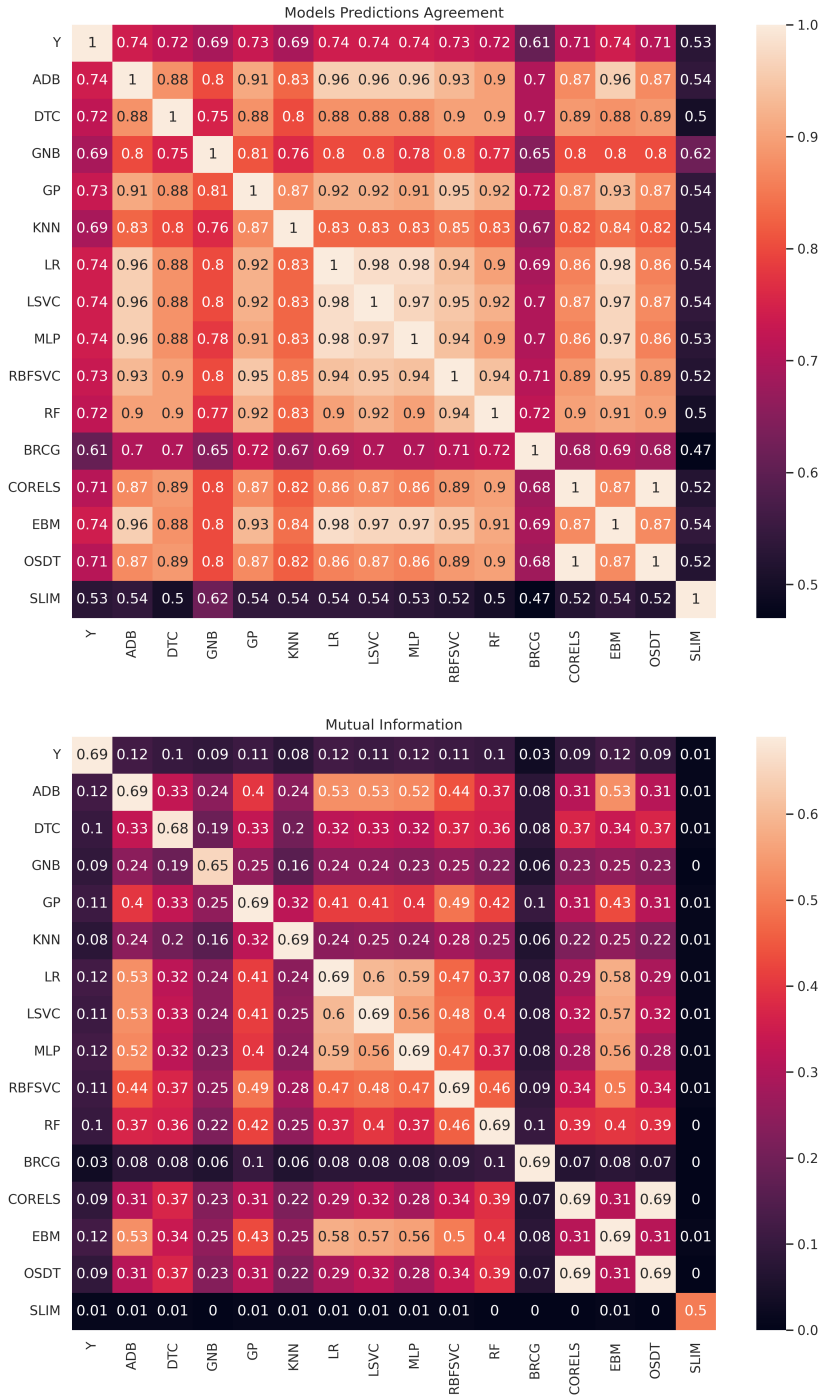


Figure 24: Heatmaps indicating the models predictions' pairwise agreement (top) and mutual information (bottom) for the FICO (binary) dataset.

Appendix G. Additional Examples

G.1 Additional Examples of ICM

Example 1: OSDT

OSDT (Optimal Sparse Decision Tree, Hu et al., 2019) is an algorithm that generates a decision tree classifier for binary datasets. Figure 25 shows an example for an OSDT model trained to identify the 10-mismatch between the models RBFSVC vs. DTC on the Bank Marketing dataset.

	Predicted rest	Predicted 10-mismatch
Actual rest	8135	2
Actual 10-mismatch	17	84

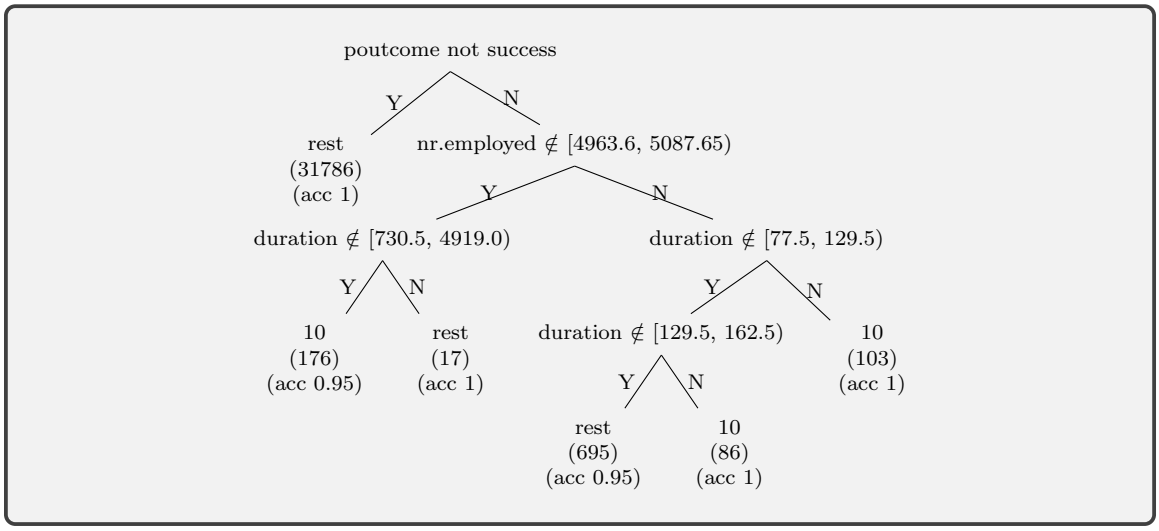


Figure 25: OSDT confusion matrix and decision tree for RBFSVC vs. DTC "10-mismatch" on the Bank Marketing dataset. "Poutcome" indicates outcome of the previous marketing campaign, "nr.employed" indicates the number of employees, and "duration" indicates the last contact duration, in seconds (S. Moro & Rita, 2014).

The leaf nodes of the trees show the predictions, the number of samples and the training accuracy of each leaf. The union of the leafs whose prediction is 10 indicate the conditions for the 10-mismatch between the two models.

Similarly, we can construct an OSDT to explain the 01-mismatch between the RBFSVC and DTC models as shown in Figure 26.

Example 2: EBM

EBM (Explainable Boosting Machine, Nori et al., 2019) is a generalized additive model (GAM) that utilizes boosting. When trained on data, it returns a non-linear function ("Score") for each feature (or pairs of features if interaction terms are enabled), which are used for classification based on where their total summation exceeds a certain threshold.

	Predicted rest	Predicted 01-mismatch
Actual rest	7600	134
Actual 01-mismatch	179	325

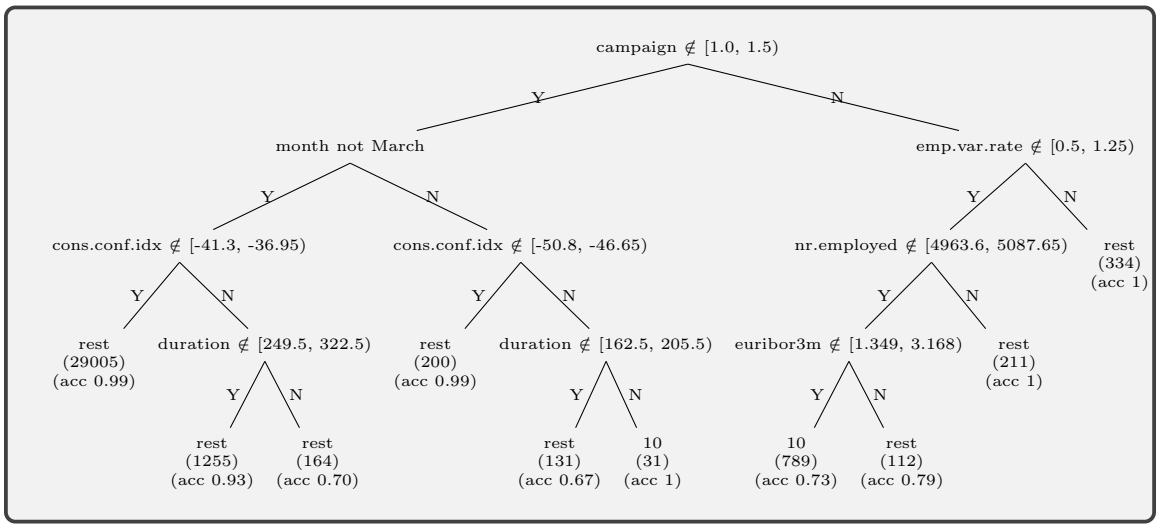


Figure 26: OSDT decision tree for RBFSVC vs. DTC "01-mismatch" on the Bank Marketing dataset. "Campaign" indicates the number of contacts performed during this campaign and for this client, "cons.conf.idx" indicates the consumer confidence index, and "emp.var.rate" indicates the employment variation rate.

The confusion matrix below shows the performance of EBM in identifying the 01-mismatch between the models RBFSVC and DTC on the Bank Marketing dataset. We find that this is very accurate.

Figure 27 shows several plots that improve our understanding of the EBM model. The top plot shows the mean absolute score, which is the average contribution of a feature to prediction (i.e., the average absolute values returned by the non-linear functions corresponding to each feature). The subsequent plots show the non-linear function of individual and pairs of features. For example, the duration feature is important to understanding the difference between the two models. Moreover, when the value of duration is between roughly 900 to 1600, the 01-mismatch is more likely to occur. We similarly observe other features and pairs of features that are important for 01-mismatch prediction.

	Predicted rest	Predicted 01-mismatch
Actual rest	7727	7
Actual 01-mismatch	10	494

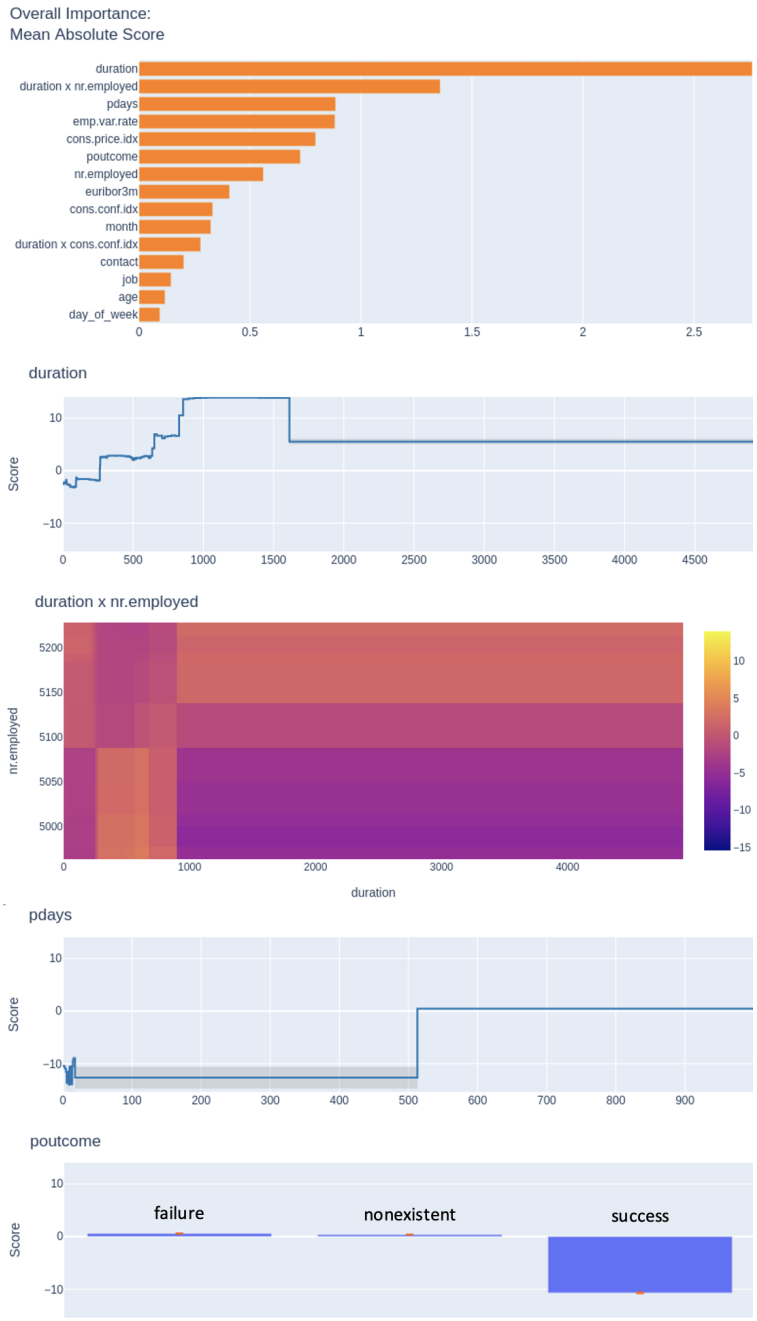


Figure 27: EBM confusion matrix and score functions for the features with the highest importance for predicting 01-mismatch for the pair of models RBFSVC vs. DTC on the Bank Marketing dataset. "Pdays" indicates the number of days that passed after the client was last contacted since the previous campaign.

G.2 Additional Table: PCA Table for Figure 9

	DR1	DR2	Test Accuracy
Y	−9.77	8.15	1.0
ADB	−13.44	1.56	0.86
DTC	−8.51	−2.72	0.84
GNB	10.54	−8.76	0.8
GP	14.82	29.28	0.7
KNN	13.52	−1.14	0.79
LR	−13.01	−0.38	0.85
LSVC	14.69	−4.54	0.79
MLP	11.29	−8.83	0.79
RBFSVC	14.59	−3.26	0.79
RF	4.3	−2.96	0.82
BRCG	−9.44	−3.45	0.83
CORELS	4.65	−5.09	0.81
EBM	−13.41	2.42	0.87
OSDT	−9.44	−3.45	0.83
SLIM	−11.4	3.17	0.84

Table 1: The numeric values of the PCA transformation of Figure 9. The first two columns indicate the principal components, and the third column shows each models' test accuracy.

G.3 Additional Examples of VML

Example 1: Simple Comparison of Model Predictions Using DR

One of the most straightforward ways to compare models is to use scatter plots to visualize the dimension reduction of the data matrix, color observations according to their predictions, and present the plots of each model next to each other. For example, Figure 28 compares model predictions on the Adult dataset. In this case, PCA was used to reduce the data matrix to two dimensions. The yellow markers represent the positive labels ("y_pos"), while the blue markers represent observations with negative ("y_neg") labels. The top-left subfigure shows the true labels and serves as the baseline, while the other subfigures show the predictions of the different models. This provides information about how frequently models make different predictions (e.g., GP tends to make negative predictions and RBFSVC tends to make positive predictions) and how these predictions change across the DR space, for each model.

Example 2: Comparing Hyperparameters

As with comparing models of different hypothesis class, we could visualize multiple models in the same hypothesis class with different hyperparameters, which could potentially assist in hyper-parameter tuning. Figure 29 illustrates this using various combinations of hyperparameters for three models (KNN, MLP, and RF) on the German Credit dataset,

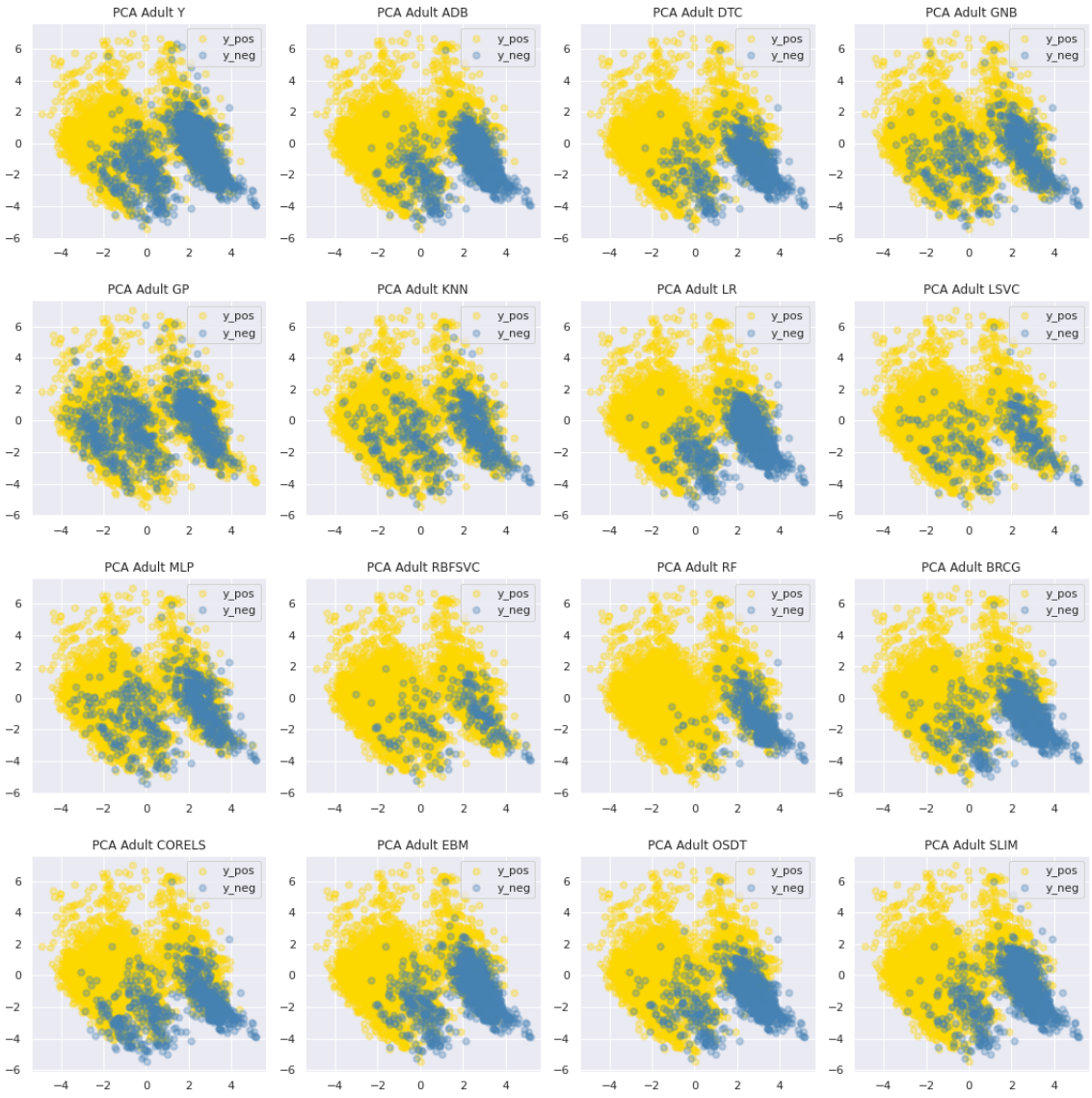


Figure 28: Illustration of a simple DR comparison of VMLs obtained using PCA on the Adult dataset.

where each model is represented by a different color. Models from the same family tend to cluster together (this can be observed by the color of the model in the figure). In the case of overlaps between models, it could help identify substitutes from models of different classes that may be more appropriate for the specific application.

Example 3: Accelerating Computation Using Confusion Matrices

A potential computation challenge in creating VML could arise in datasets with a large number of observations (this, in turn, affects the matrix that is reduced to create the plot).

Figure 29: A scatter plot that uses dimensionality-reduced model predictions through multiple parameter combinations of KNN, MLP, and RF (distinguished by color and shape) obtained using PCA on the German Credit dataset. The KNN models coincide.

One alternative is to create similar plots using the confusion matrix of each model. That is, we first compute the confusion matrix of each model, and organize these values in a matrix with a row per model and a column per value in the confusion matrix. We then apply any desirable DR method on this matrix to create the plot. Figure 30 illustrates a scatter plot created this way using the Magic dataset. Similar to Figure 9, the 10 circle markers in orange represent standard models; the five diamond markers in blue represent interpretable models; and the square green marker represents the baseline of the true labels. Despite being created using summary statistics and not complete predictions, the plot meaningfully presents models according to their performance. This implies that close distance results in similar overall performance (as captured by the confusion matrix) rather than similarity in all predictions.

G.4 Additional Examples of VCM

Example 1: Scatter Plots of VCMs

Scatter plots of VCMs visualize how joint predictions of the compared models are distributed in a reduced-feature space. Figures 31a and 31b illustrate a VCM scatter plot that compares two models (GNB and OSDT) on the Magic dataset. The plot was generated by applying a DR method (PCA) to the data matrix $\mathbf{X}$ and by coloring observations according to the joint predictions of GNB and OSDT. In each subplot, the X-axis represents DR1 (the first dimension of the two reduced dimensions), while the Y-axis represents DR2 (the second dimension). Figure 31a shows instead of having one figure, we now have four figures separated according to the joint predictions. As with the confusion matrix, the subplots

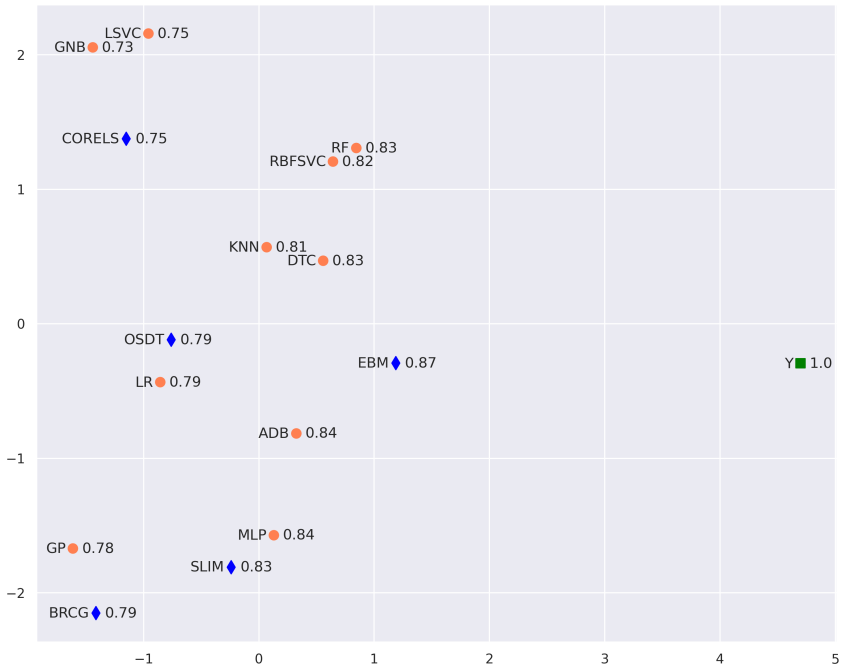
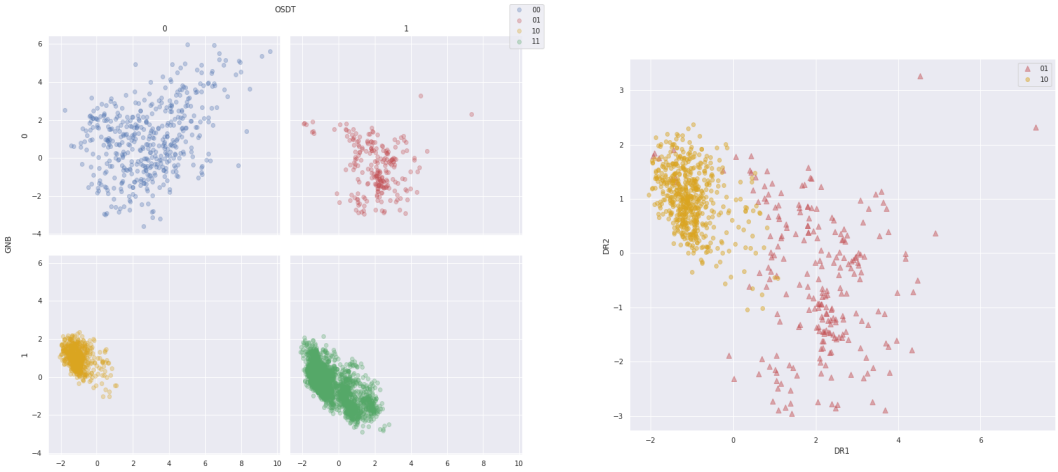


Figure 30: A scatter plot VML based on confusion matrices obtained using PCA on the Magic dataset.

on the main diagonal present observations in which both models predict the same values, and on the secondary diagonal are the observations that the models predict differently. For example, the bottom left part of Figure 31a shows observations in which GNB predicts 1 and OSDT predicts 0. When we choose between these two models, we exchange red and yellow observations.

Figure 31b consolidates differently-predicted observations into a single plot, which illustrates the tradeoff between the two models—choosing one model over another results in trading off the one type of prediction error over the other. Observations with the 10 label (in yellow) and observations labeled 01 (in red) are mostly located in separate areas of the reduced-dimensional space. There is a rough separation between the joint predictions 01 and 10 near the value 0 on the first dimension. This means that the 10 discrepancy is more likely to occur on clusters and observations that are located in the negative region of the first dimension, and the 01 discrepancy is more likely to occur in clusters that are located in the positive part of the first dimension. If there is an intuitive meaning to each dimension that relates them to features, we could use it to explain which observations are affected by choosing between the two models. This is a direction that we explore in the next example.

Note that by comparing density and scatter VCMs, we observe that the scatter plot better presents detailed information about particular observations and the predictions of the models. On the other hand, the density plot approximates well the overall distribution of the data and the regions in which predictions are either similar or different.



(a) Visual confusion matrix

(b) Visual confusion matrix on 01 and 10

Figure 31: Scatter plots of VCM obtained using PCA for the models GNB vs. OSDT on the Magic dataset.

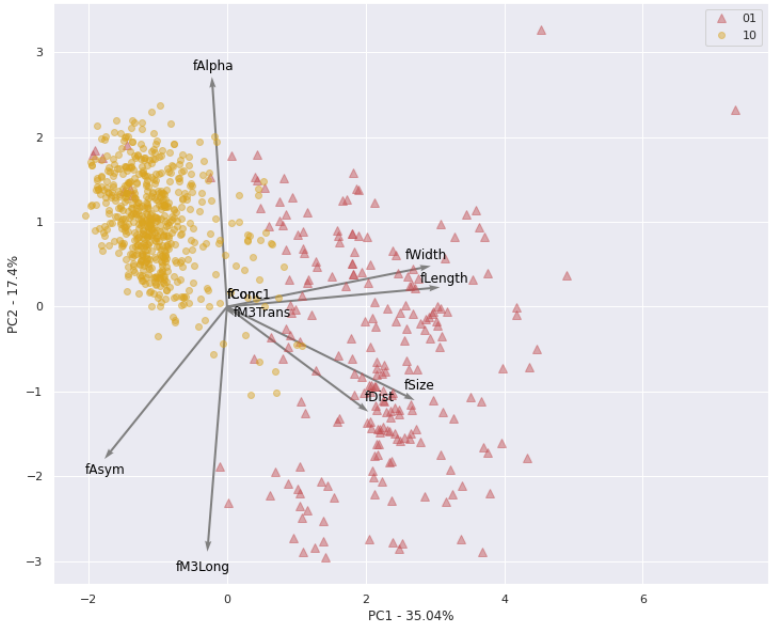


Figure 32: Biplot of VCM (principle components) obtained using PCA for the models GNB and OSDT using the Magic dataset.

Example 2: Biplots of VCMs

Biplots of VCMs aim to translate the meaning of the reduced dimensions into features. They visualize features on top of scatter plots of observations and reveal the role of feature importance in model selection. Figure 32 shows a biplot on top of the scatter VCM plot that focuses on the 01 and 10 joint predictions (the same dataset and model). We see that the first loading vector (PC1), shown on the X-axis, explains 35% of the variation; thus placing more weight on features whose corresponding vectors point horizontally. The second loading vector (PC2) is, by definition, orthogonal to PC1. It is shown on the Y-axis and explains 17% of the variation and places more weight on the features whose corresponding vectors point vertically. We can then deduce that large and small values of certain features tend to lead to certain types of joint predictions. For example, large values of the features fWidth, fLength, fSize, and fDists are associated with the joint prediction 01 where the model GNB predicts 0 and OSDT predicts 1. This could potentially help model designers decide whether a certain tendency is desirable.

G.5 Additional Example of VCX

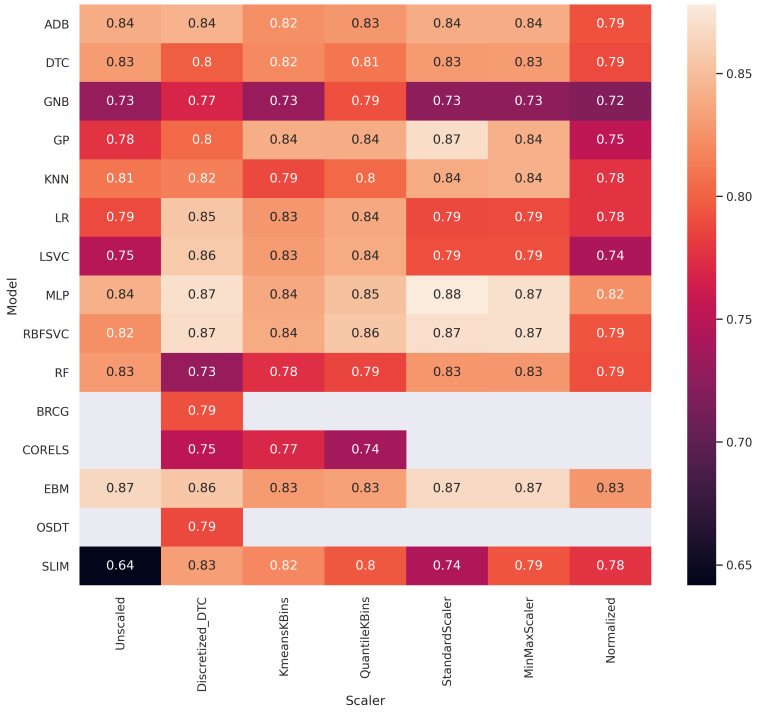


Figure 33: VCX for comparison of model performance across various preprocessing methods on the Magic dataset.

Example 1: A Comparison Preprocessing Methods

Figure 33 illustrates the use of VCX to compare models and preprocessing methods and presents a heatmap of the test accuracy on the Magic dataset across different models (X-

axis) trained after various preprocessing methods (Y-axis) were applied. Linear models such as LR and LSVC are not flexible enough and work better with discretized data, while generalized additive models (i.e., linear models based on nonlinear transformation of each feature) such as EBM perform well with continuous datasets. Furthermore, non-parametric models (including tree-based methods) such as MLP, DTC, and RF work better with continuous datasets. The Normalization preprocessing method performs poorly, and SLIM does not perform well on unscaled continuous datasets due to its sparse nature. Finally, there is no dominant preprocessing method. Such observations can be easily made using this approach—they inform the model developer about the behavior of the models and guide the model development process.

Appendix H. Additional VML Dendrograms

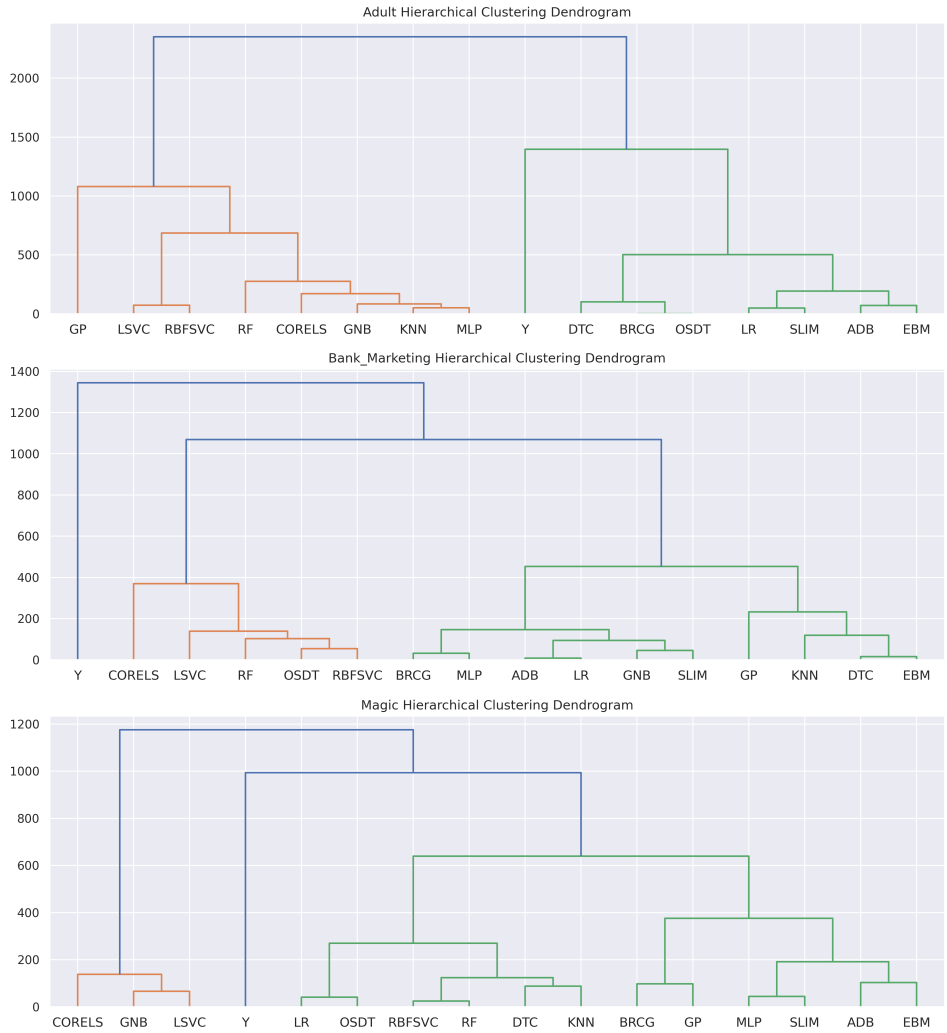


Figure 34: Dendrograms of VMLs on the Adult, Bank Marketing, and Magic datasets; hierarchical clustering applied to the confusion matrix.

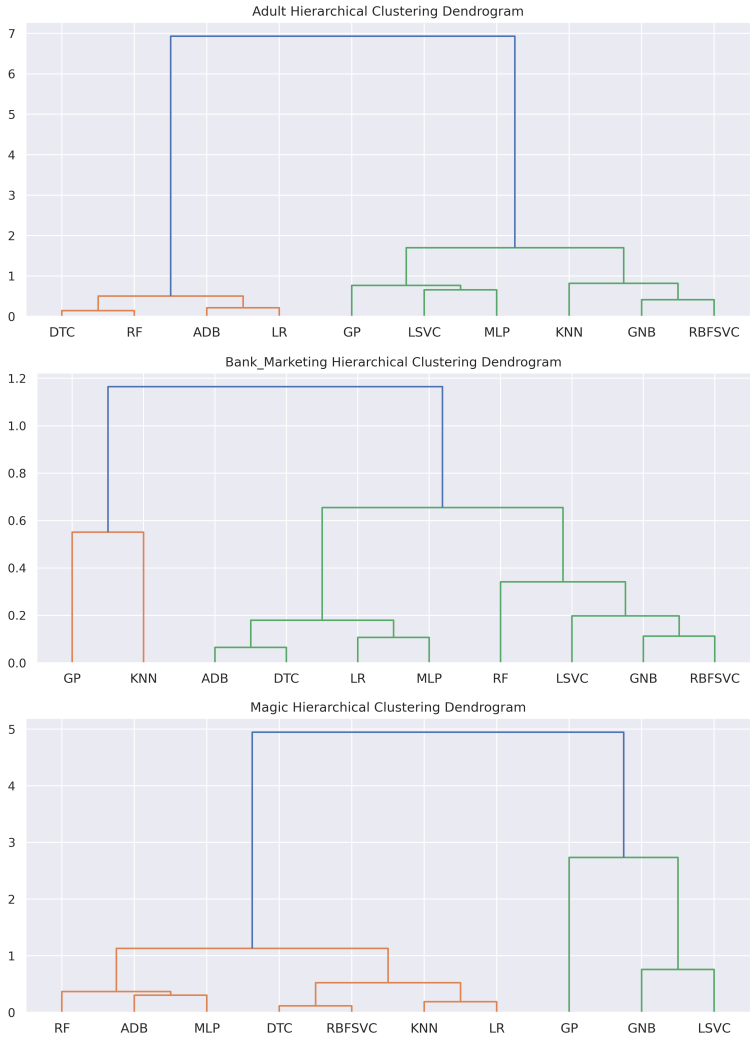


Figure 35: Illustration of dendrograms of VMLs on the Adult, Bank Marketing, and Magic dataset: hierarchical clustering applied to ROC curves.

Appendix I. Barplot of Average Model Decile Score Predictions

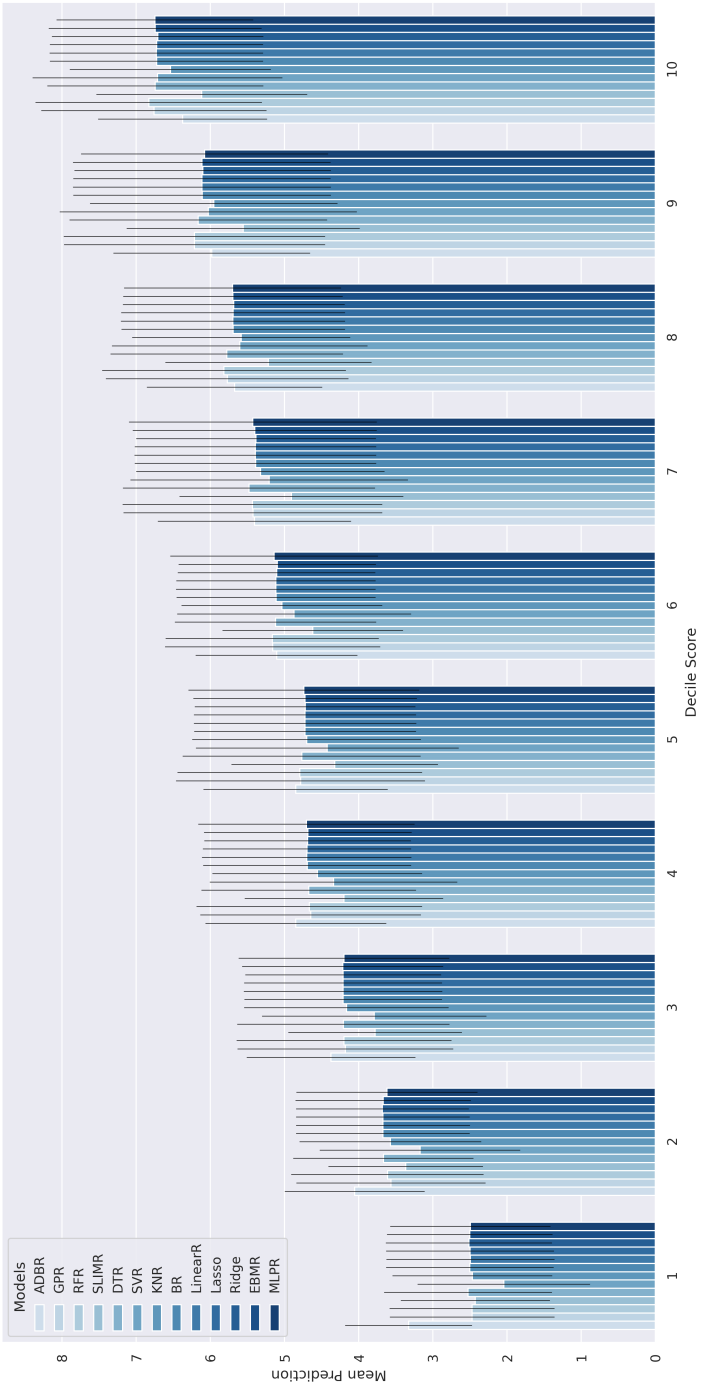


Figure 36: Barplot of average model predictions (Y-axis) for every actual COMPAS decile score (X-axis). The models are ranked from the lowest overall performance (left) to highest overall performance (right). The error bars indicate the standard deviation.

References

Amid, E., & Warmuth, M. K. (2019). Trimap: Large-scale dimensionality reduction using triplets. *arXiv preprint arXiv:1910.00204*.

Angelino, E., Larus-Stone, N., Alabi, D., Seltzer, M., & Rudin, C. (2017). Learning certifiably optimal rule lists for categorical data. *arXiv preprint arXiv:1704.01701*.

Asuncion, A., & Newman, D. (2007). *UCI Machine Learning Repository.* Irvine, CA, USA.

Ban, G.-Y., El Karoui, N., & Lim, A. E. (2018). Machine learning and portfolio optimization. *Management Science*, *64*(3), 1136–1154.

Bennetot, A., Donadello, I., Qadi, A. E., Dragoni, M., Frossard, T., Wagner, B., ... others (2021). A practical tutorial on explainable ai techniques. *arXiv preprint arXiv:2111.14260*.

Bertomeu, J. (2020). Machine learning improves accounting: discussion, implementation and research opportunities. *Review of Accounting Studies*, *25*(3), 1135–1155.

Bertsimas, D., Allison, K., & Pulleyblank, W. R. (2016). *The analytics edge.* Dynamic Ideas LLC Charlestown, MA.

Bertsimas, D., Brynjolfsson, E., Reichman, S., & Silberholz, J. (2015). Or forum—tenure analytics: Models for predicting research impact. *Operations Research*, *63*(6), 1246–1261.

Bertsimas, D., O'Hair, A., Relyea, S., & Silberholz, J. (2016). An analytics approach to designing combination chemotherapy regimens for cancer. *Management Science*, *62*(5), 1511–1531.

Bertsimas, D., Pauphilet, J., Stevens, J., & Tandon, M. (2021). Predicting inpatient flow at a major hospital using interpretable analytics. *Manufacturing & Service Operations Management*.

Bock, R. K. (1998). *Magic gamma telescope data set.* Retrieved 2021-04-25, from https://archive.ics.uci.edu/ml/datasets/MAGIC+Gamma+Telescope

Bose, I., & Mahapatra, R. K. (2001). Business data mining—a machine learning perspective. *Information & Management*, *39*(3), 211–225.

Brei, V. A., et al. (2020). Machine learning in marketing: Overview, learning strategies, applications, and future developments. *Foundations and Trends® in Marketing*, *14*(3), 173–236.

Breiman, L. (2001). Statistical modeling: the two cultures (with comments and a rejoinder by the author). *Statistical Science*.

Busemeyer, J. R., & Wang, Y.-M. (2000). Model comparisons and model selections based on generalization criterion methodology. *Journal of Mathematical Psychology*, *44*(1), 171–189.

Cadamuro, G., Gilad-Bachrach, R., & Zhu, X. (2016). Debugging machine learning models. In *ICML Workshop on Reliable Machine Learning in the Wild* (Vol. 103).

Center, M. O. R. (2021). *Delphi epidemiological case predictions.* Retrieved from https://www.covidanalytics.io/projections

Chakarov, A., Nori, A., Rajamani, S., Sen, S., & Vijaykeerthy, D. (2016). Debugging machine learning tasks. *arXiv preprint arXiv:1603.07292*.

Chapman, A., David or Jain. (1994). *Musk (version 2) data set.* Retrieved 2021-04-25, from https://archive.ics.uci.edu/ml/datasets/Musk+(Version+2)

Chen, C., Li, O., Tao, C., Barnett, A. J., Su, J., & Rudin, C. (2018). This looks like that: deep learning for interpretable image recognition. *arXiv preprint arXiv:1806.10574*.

Chen, C., Lin, K., Rudin, C., Shaposhnik, Y., Wang, S., & Wang, T. (2022). A holistic approach to interpretability in financial lending: Models, visualizations, and summary-explanations. *Decision Support Systems*, *152*, 113647.

Choi, T.-M., Wallace, S. W., & Wang, Y. (2018). Big data analytics in operations management. *Production and Operations Management*, *27*(10), 1868–1883.

Claeskens, G., & Hjort, N. L. (2008). *Model selection and model averaging*. Cambridge University Press.

Coker, B., Rudin, C., & King, G. (2021). A theory of statistical inference for ensuring the robustness of scientific results. *Management Science*.

Dash, S., Günlük, O., & Wei, D. (2018). Boolean decision rules via column generation. *arXiv preprint arXiv:1805.09901*.

Dixon, M. F., Halperin, I., & Bilokon, P. (2020). *Machine learning in finance*. Springer.

Dong, J., & Rudin, C. (2020). Variable importance clouds: A way to explore variable importance for the set of good models. *arXiv:1901.03209v2*.

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

Doshi-Velez, F., & Kim, B. (2018). Considerations for evaluation and generalization in interpretable machine learning. In *Explainable and interpretable models in computer vision and machine learning* (pp. 3–17). Springer.

Du, M., Liu, N., & Hu, X. (2019). Techniques for interpretable machine learning. *Communications of the ACM*, *63*(1), 68–77.

Dua, D., & Graff, C. (2017). *UCI machine learning repository*. Retrieved from http://archive.ics.uci.edu/ml

Elmachtoub, A., Liang, J. C. N., & McNellis, R. (2020). Decision trees for decision-making under the predict-then-optimize framework. In *International Conference on Machine Learning* (pp. 2858–2867).

Elmachtoub, A. N., & Grigas, P. (2021). Smart "predict, then optimize". *Management Science*.

Emerson, S., Kennedy, R., O'Shea, L., & O'Brien, J. (2019). Trends and applications of machine learning in quantitative finance. In *8th International Conference on Economics and Finance Research (ICEFR 2019)*.

Ferreira, K. J., Lee, B. H. A., & Simchi-Levi, D. (2016). Analytics for an online retailer: Demand forecasting and price optimization. *Manufacturing & Service Operations Management*, *18*(1), 69–88.

FICO. (2018). *Explainable machine learning challenge*. Retrieved from https://community.fico.com/s/explainable-machine-learning-challenge (Accessed: 2018-11-02)

Fisher, A., Rudin, C., & Dominici, F. (2019). All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *arXiv:1801.01489v5*.

Friedman, J., Hastie, T., Tibshirani, R., et al. (2001). *The elements of statistical learning* (Vol. 1) (No. 10). Springer series in statistics New York.

Fu, R., Huang, Y., & Singh, P. V. (2021). Crowds, lending, machine, and bias. *Information Systems Research*, *32*(1), 72–92.

Galli, S. (2021). Feature-engine: A python package for feature engineering for machine learning. *Journal of Open Source Software*, *6*(65), 3642.

Guo, X., Grushka-Cockayne, Y., & De Reyck, B. (2021). Forecasting airport transfer passenger flow using real-time data and machine learning. *Manufacturing & Service Operations Management*.

Hofmann, H. (1994). *Statlog (german credit data) data set.* Retrieved 2020-06-01, from https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)

Holzinger, A., Saranti, A., Molnar, C., Biecek, P., & Samek, W. (2022). Explainable ai methods-a brief overview. In *xxAI-Beyond Explainable AI: International workshop, held in conjunction with ICML 2020, July 18, 2020, Vienna, Austria, revised and extended papers* (pp. 13–38).

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, *24*(6), 417.

Hu, X., Rudin, C., & Seltzer, M. (2019). Optimal sparse decision trees. *Advances in Neural Information Processing Systems (NeurIPS)*.

Ibrahim, R. (2018). Sharing delay information in service systems: a literature survey. *Queueing Systems*, *89*(1-2), 49–79.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning* (Vol. 112). Springer.

Kadane, J., & Lazarand, N. (2004). Methods and criteria for model selection. *Journal of the American Statistical Association*.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kohavi, R., & Becker, B. (1994). *Adult data set.* Retrieved 2021-04-25, from https://archive.ics.uci.edu/ml/datasets/adult

Kuo, Y.-H., Chan, N. B., Leung, J. M., Meng, H., So, A. M.-C., Tsoi, K. K., & Graham, C. A. (2020). An integrated approach of machine learning and systems thinking for waiting time prediction in an emergency department. *International Journal of Medical Informatics*, *139*, 104143.

Larson, J., Mattu, S., Kirchner, L., & Angwin, J. (2016). *How we analyzed the compas recidivism algorithm.* Retrieved 2022-01-04, from https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm

Lee, E. K., Nakaya, H. I., Yuan, F., Querec, T. D., Burel, G., Pietz, F. H., ... Pulendran, B. (2016). Machine learning for predicting vaccine immunogenicity. *Interfaces*, *46*(5), 368–390.

Lin, A. X., Ho, A. F. W., Cheong, K. H., Li, Z., Cai, W., Chee, M. L., ... Ong, M. E. H. (2020). Leveraging machine learning techniques and engineering of multi-nature features for national daily regional ambulance demand prediction. *International Journal of Environmental Research and Public Health*, *17*(11), 4179.

Lourenço, R., Freire, J., & Shasha, D. (2019). Debugging machine learning pipelines. In *Proceedings of the 3rd International Workshop on Data Management for End-to-End Machine Learning* (pp. 1–10).

Ma, L., & Sun, B. (2020). Machine learning and ai in marketing–connecting computing power to human insights. *International Journal of Research in Marketing*, *37*(3), 481–504.

MacKay, D. J. (2003). *Information theory, inference and learning algorithms*. Cambridge University Press.

McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Melançon, G. G., Grangier, P., Prescott-Gagnon, E., Sabourin, E., & Rousseau, L.-M. (2021). A machine learning-based system for predicting service-level failures in supply chains. *INFORMS Journal on Applied Analytics*, *51*(3), 200–212.

Mišić, V. V., & Perakis, G. (2020). Data analytics in operations management: A review. *Manufacturing & Service Operations Management*, *22*(1), 158–169.

Molnar, C. (2020). *Interpretable machine learning*. Lulu.com.

Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., & Yu, B. (2019). Interpretable machine learning: definitions, methods, and applications. *arXiv preprint arXiv:1901.04592*.

Narkar, S., Zhang, Y., Liao, Q. V., Wang, D., & Weisz, J. D. (2021). Model lineupper: Supporting interactive model comparison at multiple levels for automl. In *26th International Conference on Intelligent User Interfaces* (pp. 170–174).

Niculescu-Mizil, A., Perlich, C., Swirszcz, G., Sindhwani, V., Liu, Y., Melville, P., ... others (2009). Winning the KDD cup orange challenge with ensemble selection. In *KDD-Cup 2009 Competition* (pp. 23–34).

Nori, H., Jenkins, S., Koch, P., & Caruana, R. (2019). Interpretml: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223*.

Northpointe. (2019). *Practitioner's Guide to COMPAS Core.* Retrieved 2021-12-01, from http://www.equivant.com/wp-content/uploads/Practitioners-Guide-to-COMPAS-Core-040419.pdf

Parzen, E. (1962). On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, *33*(3), 1065–1076.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Punj, G., & Stewart, D. W. (1983). Cluster analysis in marketing research: Review and suggestions for application. *Journal of Marketing Research*, *20*(2), 134–148.

Qi, M., Mak, H.-Y., & Shen, Z.-J. M. (2020). Data-driven research in retail operations—a review. *Naval Research Logistics (NRL)*, *67*(8), 595–616.

Rao, C., Wu, Y., Konishi, S., & Mukerjee, R. (2001). On model selection. *Lecture Notes-Monograph Series*, 1–64.

Ren, D., Amershi, S., Lee, B., Suh, J., & Williams, J. D. (2016). Squares: Supporting interactive performance analysis for multiclass classifiers. *IEEE Transactions on Visualization and Computer Graphics*, *23*(1), 61–70.

Rudin, C., Chen, C., Chen, Z., Huang, H., Semenova, L., & Zhong, C. (2021). Interpretable machine learning: Fundamental principles and 10 grand challenges. *arXiv preprint arXiv:2103.11251*.

Rudin, C., & Shaposhnik, Y. (2023). Globally-consistent rule-based summary-explanations for machine learning models: Application to credit-risk evaluation. *Journal of Machine Learning Research*, *24*(16), 1–44.

Rundo, F., Trenta, F., di Stallo, A. L., & Battiato, S. (2019). Machine learning for

quantitative finance applications: A survey. *Applied Sciences*, *9*(24), 5574.

Rust, R. T., Simester, D., Brodie, R. J., & Nilikant, V. (1995). Model selection criteria: An investigation of relative accuracy, posterior probabilities, and combinations of criteria. *Management Science*, *41*(2), 322–333.

Schlimmer, J. (1981). *Mushroom data set.* Retrieved 2021-04-25, from https://archive.ics.uci.edu/ml/datasets/mushroom

Semenova, L., Rudin, C., & Parr, R. (2020). A study in rashomon curves and volumes: A new perspective on generalization and model simplicity in machine learning. *arXiv:1908.01755v2*.

Shah, P., Kendall, F., Khozin, S., Goosen, R., Hu, J., Laramie, J., ... Schork, N. (2019). Artificial intelligence and machine learning in clinical development: a translational perspective. *NPJ Digital Medicine*, *2*(1), 1–5.

Shen, X., & Ye, J. (2002). Adaptive model selection. *Journal of the American Statistical Association*, *97*(457), 210–221.

Shin, D., He, S., Lee, G. M., Whinston, A. B., Cetintas, S., & Lee, K.-C. (2020). Enhancing social media analysis with visual data analytics: A deep learning approach. *MIS Quarterly*, *44*(4), 1459–1492.

Singh, C., Nasseri, K., Tan, Y. S., Tang, T., & Yu, B. (2021). imodels: a python package for fitting interpretable models. *Journal of Open Source Software*, *6*(61), 3192.

S. Moro, P. C., & Rita, P. (2014). *Bank marketing data set.* Retrieved 2021-04-25, from https://archive.ics.uci.edu/ml/datasets/Bank+Marketing

Sun, D., Feng, Z., Chen, Y., Wang, Y., Zeng, J., Yuan, M., ... Qu, H. (2020). Dfseer: A visual analytics approach to facilitate model selection for demand forecasting. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (pp. 1–13).

Sun, J., Zhang, D. J., Hu, H., & Van Mieghem, J. A. (2021). Predicting human discretion to adjust algorithmic prescription: A large-scale field experiment in warehouse operations. *Management Science*.

Tenenbaum, J. B., De Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*(5500), 2319–2323.

Ustun, B., & Rudin, C. (2016). Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, *102*(3), 349–391.

Wang, K., Shat, S., Chen, H., Perrault, A., Doshi-Velez, F., & Tambe, M. (2021). Learning mdps from features: Predict-then-optimize for sequential decision problems by reinforcement learning. *arXiv preprint arXiv:2106.03279*.

Wang, Y., Huang, H., Rudin, C., & Shaposhnik, Y. (2021). Understanding how dimension reduction tools work: an empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *The Journal of Machine Learning Research*, *22*(1), 9129–9201.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, *5*(2), 241–259.

Xia Liu, A., Li, Y., & Xu, S. X. (2021). Assessing the unacquainted: Inferred reviewer personality and review helpfulness. *MIS Quarterly*, *45*(3).

Zhang, J., Wang, Y., Molino, P., Li, L., & Ebert, D. S. (2018). Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models. *IEEE Transactions on Visualization and Computer Graphics*, *25*(1), 364–373.

Zhang, X., Zhu, X., & Wright, S. (2018). Training set debugging using trusted items. In *Proceedings of the AAAI conference on Artificial Intelligence* (Vol. 32).

Zhang, Y., Nguyen, L. T., & Zhang, J. (2013). Wait time prediction: How to avoid waiting in lines? In *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication* (pp. 481–490).