Understanding and Exploring Meta-Features based on Data Properties in Meta-Learning for Regression Analysis

by

Yuqi Meng

B.Sc., University of California, Los Angeles, 2020

Project Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
Department of Statistics and Actuarial Science
Faculty of Science

SIMON FRASER UNIVERSITY
Summer 2023

Declaration of Committee

Name: **Yuqi Meng**

Degree: **Master of Science**

Thesis title: **Understanding and Exploring Meta-Features based on Data Properties in Meta-Learning for Regression Analysis**

Committee:

Chair: Richard Lockhart
Professor, Statistics and Actuarial Science

Thomas Loughin
Supervisor
Professor, Statistics and Actuarial Science

Owen Ward
Committee Member
Assistant Professor, Statistics and Actuarial Science

Haolun Shi
Examiner
Assistant Professor, Statistics and Actuarial Science

Abstract

In recent years, meta-learning has gained significant attention in recommending machine learning algorithms. These recommendations rely on meta-features that are used to quantify the characteristics of input datasets. However, existing meta-features are predominantly designed for classification tasks, leaving a gap in their potential use for regression analysis. This paper aims to address this gap by identifying seven data properties that might be crucial in differentiating regression algorithms and by proposing a set of meta-features designed to capture these properties. To evaluate the efficacy of these meta-features, we conduct a simulation study that investigates their ability to reflect the desired data properties. The simulation study systematically manipulates key factors, including data linearity, true error variance, the proportion of relevant explanatory variables, and the signal-to-noise ratio, among others. This enables us to examine how the meta-features respond to changes in the targeted data properties and whether they exhibit sensitivity or specificity toward those specific properties. By analyzing the correlation between the identified data properties and their corresponding meta-features, along with considering the computational time involved, we demonstrate that many of these measures exhibit strong discriminative power without imposing excessive computational complexity.

Keywords: meta-features; regression analysis; algorithm selection; meta-learning; simulation

Acknowledgements

First and foremost, I would like to express my heartfelt gratitude to my supervisor, Dr. Thomas Loughin. Without his continuous support, this project would not have been completed with such smoothness. The guidance provided by Tom throughout my master's program went beyond just professional advice. Looking back on our regular meetings, I find myself genuinely missing the knowledge and encouragement he provided. Tom created a truly supportive and motivating environment that fostered my academic and personal growth. What added a uniquely delightful layer to our talks was Tom's wonderful sense of humor. His anecdotes never failed to brighten my day, leaving a lasting smile on my face and turning each meeting into an enlightening and enjoyable experience. Above all, he showed genuine care for my development as an individual. His influence on my academic and personal journey has been invaluable. I will always remember the support, wisdom and warmth he shared.

I am also deeply thankful to Dr. Richard Lockhart for chairing my defence and to Dr. Haolun Shi and Dr. Owen Ward for serving on my committee and providing insightful suggestions that enhanced my work. Special thanks go to all the staff and faculty in the Department of Statistics and Actuarial Science, especially Dr. Thomas Loughin, Dr. Richard Lockhart, Dr. Rachel Altman, Dr. Boxin Tang, and Dr. Jinko Graham, for offering such valuable courses. Their wisdom and expertise have played an enormous role in shaping my academic journey and making it so fulfilling. Additionally, I am grateful for the opportunity to work with Professor Ian Bercovitz as a statistical consultant. This experience has provided me with valuable insights that I will carry with me into my future career.

I also want to give a big thank you to my fellow graduate students. Your companionship and support made such a huge difference for me in this new environment. I especially want to thank Linwan and Yuxin for always having my back and being such awesome friends. Additionally, I would like to express my gratitude to my friends who, despite not being physically present, still consistently showed me so much love and encouragement from afar.

Lastly, my deepest appreciation goes to my family, especially my parents, for their love and support over the years. I would not be where I am today without them, and I am so grateful for everything they have done for me.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Over the past few decades, machine learning has become a revolutionary tool and fundamentally transformed how we do data analysis, prediction, and decision-making. It has continuously been explored and applied in various fields, such as medical research, financial risk management, autonomous vehicles, and energy management. However, with the growing complexity of data and the need for sophisticated analysis, it has been found that machine learning is not a magical tool and has restrictions.

The effectiveness of machine learning algorithms heavily relies on their underlying data assumptions, which can lead to bias when these assumptions do not hold in practice [1] [2] [3]. For example, linear regression is often regarded as one of the most intuitive and widely used supervised machine-learning algorithms, assuming a linear relationship between the explanatory variable and the response variable, but this may not hold true in real-world datasets. When non-linear relationships or interactions exist, more complex models need to be applied to reduce bias caused by the overly simplistic model. However, the increase in model complexity can lead to a higher variance and a greater risk of overfitting, which indicates the model exhibits a stronger agreement with the sample data compared to the true underlying relationship in the population data. This introduces a trade-off between bias and variance that must be carefully handled [1]. To help with optimizing this trade-off, most learning algorithms typically have some tunable parameters, such as the tree size in the Random forest, the number of hidden layers in Neural networks, and regularized terms in LASSO and Ridge regression. However, this tuning process generally means evaluating the algorithms repeatedly on different subsets of data and using different values for the tuning parameters, which can be time-consuming and, therefore, prohibitive in some real-world situations.

Moreover, we cannot assume that there is one universally optimal algorithm that can solve all problems in any domain, as stated in the "no free lunch" theorem [4]. Usually, when choosing a model for the target dataset, researchers may need to evaluate multiple potential algorithms, each with a set of optimal hyperparameter values, to determine the most suitable one based on the training data [5]. Unfortunately, this process is often rather

impractical for the same reasons that tuning can be prohibitive. Since selecting the proper algorithm for a given dataset is crucial to obtaining a satisfactory model, it is important to have efficient approaches to determine which machine learning algorithm to choose.

With the increasing interest in automatic learning, meta-learning has become a popular and active research area. One application of meta-learning is algorithm selection, where the goal is to automatically determine the most suitable algorithm or combination of algorithms for a specific dataset without extensive experimental evaluation. The core idea of this technique is to understand how the properties of the input data relate to the performance of individual machine-learning algorithms. There are two key components in meta-learning: meta-features and meta-learner [6]. Specifically, meta-features are data characteristics measured from the data that are believed to be related to the performance of machine-learning algorithms [7], while the meta-learner is a machine or model that is trained on meta-features to predict a recommended learning algorithm.

Selecting a set of effective and generalizable meta-features presents a non-trivial challenge for two main reasons. First, there can be a large number of potential meta-features to choose from; for example, Rivolli et al. [8] provides an extensive list of over 90 existing meta-features. OpenML [9], which is a public online platform for machine learning research and discussion, provides a set of more than 70 data characteristics that can be used for meta-learning. However, in order to mitigate unnecessary computational costs and redundancy, it is crucial to carefully select a subset of meta-features that are relevant but non-redundant. Second, many of the meta-features mentioned in prior research papers are considered because they are easily extracted from the data rather than being carefully designed to measure characteristics that are known to be related to methods' differential performance.

Most existing meta-features in meta-learning have been primarily developed for classification tasks. For example, the survey article by Rivolli et al. [8] categorizes the existing meta-features based on their applicability to specific tasks. Most measures derived directly from the dataset use only information about the explanatory variables, such as the number of variables, the number of binary variables and the ratio of categorical versus numeric variables, and hence can be used in both supervised and unsupervised tasks. Certain measures that relate to the response variable require a categorical response variable or rely on criteria like entropy that are limited to classification problems. Additionally, there are measures that cannot be directly extracted from the data and require a general response variable. These measures can be applied to all supervised tasks but are less suitable or useful for regression tasks. For example, measures that extract information from 1-nearest neighbor models fail to adequately consider the intricate relationships between explanatory variables and response variables.

This paper aims to bridge this gap by identifying various data properties that are explicitly critical in regression analysis in relation to the predictive performance of different

learning methods, such as multicollinearity, heteroscedasticity, and nonlinearity. We refer to the 7 data properties outlined in [10] and then identify a suitable set of meta-features that are specified to capture these data properties. Then, we conduct a simulation study aimed at demonstrating the efficacy of meta-features in accurately reflecting these data properties. The objective is to assess whether the meta-features within specific groups exhibit specificity and sensitivity in capturing the desired property of the data. By investigating the correlation between data properties and meta-features, it is feasible to develop a well-selected set of meta-features tailored for regression problems.

The outline of this paper is as follows. In Chapter 2, we briefly review the meta-learning structure and summarize the commonly used meta-features in previous literature. Chapter 3 reviews the 7 data properties and provides details of the proposed meta-features. Chapter 4 describes the design of the simulation study to evaluate the performance of meta-features in Chapter 3 with various settings, and the results are provided in Chapter 5. We discuss the limitations of our approaches, challenges that need to be addressed, as well as some possible suggestions for future work in Chapter 6.

Chapter 2

Background

2.1 Meta-learning

2.1.1 Meta-learning Definition

In the past decade, the term "meta-learning", also known as "learning to learn", has been used by many researchers in the field of machine learning in various ways. Lemke et al. [11] propose a definition of meta-learning in their survey by rephrasing the common aspects found in earlier definitions in prominent review papers and books. According to Lemke et al., a meta-learning system should contain a learning subsystem that adapts through experience. This experience can be obtained either from a) exploiting meta-knowledge, which is derived from previous learning episodes on the same data, or b) from different domains and problems.

The application of meta-learning can be observed in various contexts, such as ensemble methods and hyperparameter optimization, whereas the existing literature exhibits some discrepancies regarding the precise qualifications of a meta-learning problem [11] [12] [13]. One particularly intriguing application of meta-learning lies in the domain of algorithm selection. Specifically, this approach involves training a meta-learning system to evaluate the performances of diverse algorithms and generate rankings of candidate algorithms based on their historical performances among different datasets. Subsequently, the system can produce recommendations on which algorithm is likely to perform well on a new, unseen dataset. In the following subsection, we introduce a framework that builds upon the groundwork laid out in [13] and [14], providing more detailed steps in the meta-learning process for algorithm selection.

2.1.2 Meta-learning Architecture

As defined by Vilalta et al. in [13], the meta-learning system can be structured into two stages: knowledge acquisition mode and advisory mode. During the knowledge acquisition mode, the meta-learning system initiates by generating a set of meta-features, also known as meta-knowledge or meta-data in other literature, with slight variations in definition depend-

ing on the task domain. Essentially, these meta-features attempt to relate the characteristics of the data to the performance of various learning algorithms. These meta-features are then used as input in the subsequent stage of meta-learning to determine the most suitable algorithm along with an optimal set of hyperparameters without evaluating all candidate models [15]. To illustrate, suppose certain meta-features indicate a given dataset is high-dimensional with substantial noise, intricate interactions, and complex, nonlinear relationships between explanatory and response variables. In this case, a meta-learning model should infer that a flexible method like the random forest or gradient boosting with a reasonably large number of deep trees would be more suitable than a linear regression model. Conversely, if the meta-features indicate that the dataset has a limited sample size or exhibits a strong linear relationship, linear regression may be preferred over random forest.

In the advisory mode, a meta-learner, typically a machine algorithm itself, is generated using aggregated information obtained from the previous phase. The choice of machine learning algorithms employed as meta-learner depends on several factors, including predictive accuracy, computational time, and the task's objective [12]. The first two factors, predictive accuracy and computational time, are intuitive considerations, and the third factor, the task's objective, refers to the various forms of results a meta-learner can produce. These results can include recommending the best algorithm for a given dataset, generating a ranking of different algorithms, or predicting a subset of the top-performing algorithms [14]. However, similar to the conventional process of choosing machine learning algorithms, there are no strict rules or direct guidelines for determining the most suitable meta-learner [16][17]. Since the main focus of this paper is to explore potential meta-features for regression, we will not delve into the specific details of the meta-learner. Instead, the remaining portion of this chapter provides an overview of the meta-features.

2.2 Meta-features

Meta-features are descriptive statistics extracted from the given dataset, aiming to carry a wide range of relevant information such as data types, relationships between variables, statistical distribution, the presence of random noise or irrelevant data, and more [2]. Some meta-features may be derived from outcomes produced by trained models or even directly defined based on their predictive performances, but it should be emphasized that the suitability of an algorithm for developing meta-features is not solely determined by its model performance. For instance, Gradient Boosting, a known greedy algorithm that can occasionally outperform random forest, demonstrates significant improvements in test error when using small learning rates [18]. However, this improvement comes at the cost of increased computational time during training, which contradicts the original intent of using meta-features. Therefore, as a general guideline, algorithms that require extensive tuning or involve high computational costs are not preferred for creating meta-features.

Within the field of meta-learning, meta-features are commonly classified into five categories: simple, statistical, information-theoretic, model-based, and landmarking [6][8][19]. Among these categories, the first three — simple, statistical, and information-theoretic — are the most common and traditional meta-features, as they can be directly computed from the dataset and be free of hyperparameters, whereas the remaining two categories involve applying machine learning algorithms. As a result, an alternative categorization suggests grouping meta-features into three distinct categories: direct characterization measures, model-based meta-features, and landmarkers [8][20]. In the following subsections, we will provide explanations for each category of meta-features and also introduce an additional category, complexity-related meta-features.

2.2.1 Direct Characterization Measures

Simple measures contain basic information obtained from the dataset, such as sample size and the number of numeric variables. Statistical measures focus on numeric variables and provide insights into the data distribution, including properties like outliers, skewness, and kurtosis. Information-theoretic measures, on the other hand, are mainly used for classification problems and describe the characteristics of categorical response variable [6][8]. Since measures falling within these three groups are relatively straightforward and can be easily extracted or computed directly from the dataset, we focus on the remaining categories of meta-features.

2.2.2 Model-based Measures

Model-based meta-features are measures obtained from a fitted model or algorithm; however, they should not incorporate or rely on any performance metrics. A simple example includes coefficients and their statistical significance in linear models, which are estimated during the optimization process inherent to machine learning algorithms using training data. By exploring the model-based measures, we gain valuable insights into how the model generates predictions and understand the relationships between variables and predicted outcomes. It is important to distinguish model parameters from hyperparameters, as the latter are predefined rather than learned during training [14].

The choice of algorithm for constructing model-based meta-features depends on the task. Given that the majority of existing meta-features are primarily developed for classification problems, the decision tree is the most popular option [8][21]. A trained classification tree model provides interpretable information regarding its size and tree structure, including attributes like the splitting conditions and class predictions associated with each node. These outputs are developed into meta-features to quantify data characteristics in classification problems. In their work [22], Bensusan et al. propose 10 meta-features extracted from an unpruned decision tree. For example, one of these meta-features is the ratio of the number of nodes to the number of variables, where lower values indicate the presence of irrelevant

attributes. Another meta-feature is the maximum depth of the tree, which measures the longest path from the root to a leaf node. A deeper tree potentially indicates a more complex dataset, as the model draws intricate decision boundaries by recursively splitting to capture relationships between variables.

Considering the broader context of utilizing a trained tree-based algorithm, an important factor to consider is the tree's size or complexity, which can be controlled through predefined hyperparameters. A larger tree size allows for a more complex model with more splits, while a smaller tree size results in a simpler model with fewer decision rules. This introduces a trade-off: growing a larger tree increases the computational time and the potential for misleading results due to overfitting, while a shallow tree may limit the usefulness of the measures, as the model prematurely stops splitting for complex datasets, resulting in insufficient information and potentially overlooking important variables. Similar considerations apply when developing model-based meta-features using other algorithms. Specifically, these meta-features are designed for supervised learning problems, including regression, and are expected to be deterministic conditional on the sample but sensitive to the choice of models and their associated hyperparameters [8]. Therefore, determining suitable hyperparameter values prior to training is essential to develop these meta-features. Prior knowledge or techniques such as cross-validation or grid search may be necessary for this process, but this then increases the cost of computing meta-features. Hence, it is worth exploring other simple algorithms like linear regression, which are free of hyperparameters, or some simplified versions of flexible algorithms that provide outputs with discriminatory power across different learning methods. This consideration also applies to the next category of meta-features, landmarkers.

2.2.3 Landmarkers

Landmarking is an approach used to characterize datasets by evaluating the performance of a predefined set of simple supervised learning algorithms. The essential idea of using landmarkers is to guide the choice of more complex learners based on the results obtained from these simpler and faster algorithms, such as linear regression and a shallow decision tree, or simplified versions of algorithms. The latter may be either abbreviated to run quickly or fully trained on a small fraction of the data, which is particularly useful when dealing with large sample sizes, whereas training complex models on small subsets is infeasible. Another valuable technique is the use of relative landmarkers, which involves comparing the performance of different algorithms or the same algorithm under different configurations. Relative landmarks are computed comparisons of performance between two landmarks, such as ratios or differences of error, or functions thereof [14].

There are two critical concerns raised and examined by Pfahringer [19] regarding the use of landmarking measures: the feasibility of using such measures and the appropriate choice of landmarkers. The feasibility issue arises from the reliability of using landmarking

measures as indicators for algorithm selection. While fitting simple models is convenient, there is no guarantee that the performance results from these simple learners adequately reflect the characteristics of the dataset. Additionally, the choice of landmarkers presents another significant challenge. With a wide range of available algorithms and a vast domain of learning problems, the selected model must be somehow related to the candidate algorithms to provide useful landmarkers.

In previous studies, such as [19] and [23], some widely used landmarkers include the decision node learner, randomly chosen node learner, and worst node learner. These three landmarkers are generated by fitting different versions of decision trees, which is reasonable for classification tasks. The decision node learner, for instance, indicates linear separability by choosing the most informative variable for a single split, and the other two serve similar purposes. However, in this case, these three measures are highly correlated with each other and provide limited additional information when used in combination. Besides these three, the work presented by [19] also explores the landmarkers applying other algorithms, such as the 1-nearest neighbor classifier, linear discriminant analysis, and naive Bayes classifier.

In the context of our study, which focuses on regression problems, potential candidates for simple learners include linear regression, LASSO regression, generalized additive models (GAMs), and random forest. As discussed in the previous subsection on model-based meta-features, the issue of potential needs for tuning parameters also arises for landmarkers. Fortunately, based on the findings of [24], it is observed that the default values for tuning parameters in random forest as set in the R package `randomForest` typically yield satisfactory results across various cases. Consequently, we decide to utilize the default random forest in most cases when developing our own meta-features to reduce complexity. Further details on the proposed meta-features are provided in Chapter 3.

2.2.4 Complexity-related Meta-features

As mentioned in Chapter 1, the study of meta-features for classification problems is fairly well-explored. Ho and Basu [25] propose three primary factors contributing to the complexity of a classification problem: (i) class ambiguity; (ii) data sparsity and dimensionality; and (iii) the complexity of the boundary separating the classes. Building upon this work, Lorena et al. [2] transfer and extend the measures in [25] to the regression context, categorizing them into four groups: (i) feature correlation measures; (ii) linearity measures; (iii) smoothness measures; and (iv) geometry, topology, and density measures.

Feature correlation measures primarily capture the correlation between explanatory and response variables, as well as between pairs of explanatory variables. Higher correlation suggests that simpler models may be more appropriate for the data.

Linearity measures are determined by computing the mean absolute value of residuals and the average of squared residuals obtained from a multiple linear regression model. When

these measures yield small values, it indicates that the data exhibits a more linear pattern. In such cases, simpler models are likely to be appropriate.

Smoothness measures adapt the measures proposed by Ho and Basu [25]. For instance, the *Output distribution* measure applies the Minimum Spanning Tree (MST) technique to the dataset. MST connects the most similar input data points, with the edges weighted by the Euclidean distance. This measure provides an estimate of the mean distance between the output values for neighboring points. Similarly, the *Input distribution* measure orders the data points based on their output values and computes the Euclidean distance of the ordered input data for one dimension. Another measure, the *Error of a nearest neighbor regressor*, calculates the mean squared errors of 1-nearest neighbor to investigate the associations and variations in the input and/or output data.

The final category of complexity measures focuses on the distribution and density in the input/output space. One of the measures used is the *Average number of examples per dimension*, which provides insight into the data's density along a single dimension, referred to as "data richness" in Chapter 3. The other two measures in this category require a new test set, which is generated by randomly interpolating pairs of input data points with similar outputs. Subsequently, the difference between the training and test errors is calculated for two regression models: the linear regression model (*Non-linearity of a linear regressor*) and the 1-nearest neighbor (*Non-linearity of nearest neighbor regressor*).

The primary objective of Lorena et al.'s study is to formalize a set of meta-features specifically designed to assess the complexity of regression problems. Rather than categorizing meta-features based on their function and characteristics, their emphasis is on understanding the intrinsic properties of the data itself. Their research also includes the evaluation of these meta-features using both synthetic and real datasets, making a valuable contribution to the field of meta-learning regression. Nevertheless, upon careful analysis, we identified certain areas where further improvements could be made in the development of these measures.

Firstly, the existing meta-feature categories adapted from the classification domain in [2] may not provide specific relevance or advantages to regression problems. It is acknowledged that measuring data properties like nonlinearity poses inherent difficulties due to the lack of straightforward quantification methods. Additionally, certain data properties, like multicollinearity, may have multiple measurement options, thereby adding difficulty to the process of selecting suitable measures. Still, further considerations should be given to including more explicit categories that align with the data properties influencing the selection of regression learning methods.

Secondly, the current set of measures may not be sufficient for accurately identifying the complexity level of datasets. While linear regression and 1-nearest neighbour methods can differentiate between simple and complex datasets to some extent, our aim is to determine

complexity levels more precisely. Hence, more flexible methods should be considered and applied to develop meta-features.

Lastly, it is predictable and acceptable that measures within the same category exhibit correlations with each other. However, we expect that meta-features from different categories should capture specific data properties relevant to regression without exhibiting a significant correlation to measures from other distinct groups.

In the upcoming chapter, we intend to introduce measures for 7 distinct properties to address the aforementioned areas for improvement.

Chapter 3

Review of Data Properties for Regression Problems

In this chapter, our main objective is to explore and understand 7 data properties that have the potential to impact the performance of learning algorithms for regression problems. We aim to create a set of meta-features that capture and quantify the characteristics associated with each of these properties. These meta-features are either derived from existing measures or newly developed measures specifically designed to estimate each respective property.

Notation and Review of Some Regression Methods

We use the following notation throughout the paper. We define Y as a continuous response variable, and $\mathbb{X}$ as a set of p explanatory variables, which includes $X_1, X_2, \ldots, X_p$.

The general regression model can be stated as follows:

$$Y = f(X, \beta) + \epsilon,$$

where $f(X, \beta)$ is the function describing the true relationship between the expected value of the response and the explanatory variables, also called the response surface; β represents a vector of the unknown parameters, and ϵ represents the random error between the actual response variable Y and its expected value. Unless otherwise stated, $\epsilon_1, \ldots \epsilon_n$ are assumed to be independent and normally distributed with an expected value of zero ($E(\epsilon_i) = 0$) and a constant variance ($Var(\epsilon_i) = \sigma_\epsilon^2$).

Multiple linear regression is widely used in practice and can be considered a special case of a general regression model. A multiple linear regression model assumes that the relationship between the response variable Y and the explanatory variables $X_1, X_2, \ldots, X_p$ can be expressed as a linear combination,

$$Y_i = \beta_0 + \beta_1 X_{i1} + \ldots + \beta_p X_{ip} + \epsilon_i. \tag{3.1}$$

In matrix form, the observed values of the response variable are represented as the vector $\boldsymbol{Y}$ consisting of n elements $(Y_i, i = 1, \ldots, n)$. $\boldsymbol{X}$ is the design matrix consisting of $p + 1$ n-dimensional column-vectors $[\mathbf{1}, \boldsymbol{X_j}]$ $(j = 1, \ldots, p)$. The regression coefficients are represented as the $(p + 1)$-dimensional vector $\boldsymbol{\beta}$, and the unobserved errors are represented as the n-dimensional vector $\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma_\epsilon^2 \boldsymbol{I})$. The equation (3.1) can then be written as

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}.$$

Under these model assumptions, the regression coefficients $\boldsymbol{\beta}$ can be estimated using the ordinary least squares (OLS) method, which minimizes the residual sum of squares (RSS) and provides an unbiased estimator:

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X'X})^{-1}\boldsymbol{X'Y}. \tag{3.2}$$

The variance of the OLS estimator is

$$Var(\hat{\boldsymbol{\beta}}) = \sigma^2(\boldsymbol{X'X})^{-1}.$$

The Least Absolute Shrinkage and Selection Operator (LASSO) is a popular and powerful penalized regression technique used in linear regression models. By introducing an L1-norm penalty to RSS, LASSO seeks to determine the optimal coefficient estimates, which minimize the combined objective function as given below:

$$(\boldsymbol{Y} - \boldsymbol{X}\hat{\boldsymbol{\beta}})'(\boldsymbol{Y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}) + \lambda \sum_{j=1}^{p} |\hat{\beta}_j|,$$

where λ is a tuning parameter that controls the amount of shrinkage and is selected through cross-validation [18]. If λ is not zero, the coefficient estimates are shrunk in comparison to the OLS estimates determined in (3.2). This process can lead to some coefficients for variables that do not substantially contribute to the model being reduced to zero. As a result, LASSO serves as a means of both model regularization and variable selection [26].

Regression splines offer a more flexible method to model nonlinear relationships between variables, mainly for $p = 1$. They are based on piecewise polynomial regression, dividing the range of X into distinct regions with cutpoints, known as knots, denoted as ξ_k, where $k = 1, \ldots, K$ [18]. Within each region, a polynomial function is fitted separately with added constraints that ensure the polynomials join smoothly at the knots. By introducing more knots and maintaining a fixed polynomial degree, regression splines increase flexibility, allowing the model to capture intricate non-linear patterns in the data without becoming excessively complex. This property makes regression splines and their extended methods valuable approaches for handling nonlinearity effectively in various practical applications.

However, determining the optimal number of knots (K) and their placement requires tuning. Typically, the knots are placed uniformly, which means we have equal numbers of data points between knots, and cross-validation techniques are commonly used to select the appropriate value of K that results in the smallest RSS.

By contrast, smoothing splines consider the infinite space of functions with continuous second derivatives, controlling overfitting by penalizing the amount of wiggliness in the model. It turns out that smoothing splines can be estimated by fitting cubic regression splines with a knot at each observed value of X_i and estimating the parameters by minimizing penalized residual sum of squares [18].

Generalized Additive Models (GAMs) are a direct extension of splines to the multiple regression problem. In a GAM, each linear term in (3.1) is replaced by a smoothing spline in the respective variable X_j,

$$g(\mu(X)) = \alpha + \sum_{j=1}^{p} h_j(X_j),$$

where $g(\cdot)$ represents a link function that connects the expected value of the response variable with linear predictors [18]. In linear regression, the link is the identity function. By applying the smoothing spline technique, GAMs provide a flexible framework that relaxes the additivity assumption inherent in linear models.

Multivariate Adaptive Regression Splines (MARS) offers another approach to capturing nonlinear relationships by using linear splines with adaptive knot selection. MARS considers each data point X_{ij} of X_j as a potential knot and creates pairs of piecewise linear basis functions called reflected pairs [18], which can be expressed as:

$$(X_j - t)_+ = \begin{cases} X_j - t, \text{ if } X_j > t, \\ 0, \text{ otherwise,} \end{cases} \quad \text{and} \quad (t - X_j)_+ = \begin{cases} t - X_j, \text{ if } X_j < t \\ 0, \text{ otherwise,} \end{cases}$$

where the knot value $t \in \{X_{1j}, X_{2j}, \ldots, X_{nj}\}$

MARS iteratively selects knots and their related basis functions that lead to the most significant reduction in the residual sum of squares. The process continues until a predefined maximum model size is reached, and subsequently, backward elimination is used to remove terms with the smallest reductions in the generalized cross-validation (GCV) error. This helps in determining the most relevant predictor variables and their contributions to the model's predictive performance. The optimization of the model's performance involves tuning two key parameters: the degree of interactions (cross-products of basis functions) and the number of retained terms, which can be achieved using grid search and cross-validation.

Random forest (RF) is a popular supervised learning algorithm that aggregates information from multiple regression trees. A regression tree recursively partitions the data into

subsets based on the values of the explanatory variables. The algorithm selects the variable and corresponding splitting point that best reduces the RSS of the previous split. This partitioning process continues until a stopping criterion is met, such as a predefined maximum tree depth or a minimum number of samples in each leaf node. This process cuts p-dimensional space into hyper-rectangular regions. Within each region, the sample mean of responses is used to estimate the response surface. The Random Forest algorithm generates B bootstrap samples from the original training data, randomly drawing them with replacement, usually of size n. It then constructs B regression trees using bootstrapped samples, considering only a predefined number of randomly sampled explanatory variables for splitting. The trees create different partitionings of p-dimensional space, resulting in different estimated means. The final predicted value at any point in space is the average of all tree means there. As for the data that are not included in the bootstrapped sample, they form the out-of-bag (OOB) sample. For each observation in the training sample, an OOB prediction can be created. This OOB prediction consists of the average tree mean from all trees to which the observation does not contribute. Since this predicted value is independent of the training observation, the OOB mean squared prediction error, $MSPE_b$, is a measure of generalization error for the RF.

3.1 Nonlinearity

Nonlinearity in regression describes a situation where there is a departure from a linear relationship between the explanatory variables and the response variable. The relationship may take various forms, such as quadratic ($\alpha + \beta X + \beta X^2$), power ($\alpha X^\beta$), exponential ($\alpha e^{-\beta X}$), or other forms. Theoretically, when we have prior knowledge of the specific nonlinear form, we can use that form in a model for the response. However, in many real-world situations, the true form of the nonlinear relationship remains unknown or undetectable through human inspection, especially when the dimension of data is high. If linear models are blindly fitted to data that exhibits a nonlinear relationship, it results in significant bias and inadequate predictions, regardless of how many predictors are added. In that case, it becomes essential to employ more flexible regression techniques capable of capturing nonlinearity.

One easy and commonly used technique to model nonlinearity is polynomial regression, where the linear model is extended to include polynomial terms, adding curvature to the model. For example, a polynomial regression model with a degree of d can be represented as:

$$Y_i = \beta_0 + \sum_{j=1}^{p} (\beta_{1j} X_{ij} + \beta_{2j} X_{ij}^2 + \beta_{3j} X_{ij}^3 \cdots + \beta_{dj} X_{ij}^d) + \epsilon_i.$$

However, the fully specified polynomial regression still specifies a shape that may not match the actual response surface or may overfit in some or all dimensions. To mitigate the

latter concern, higher-degree polynomial models above degree 3 are usually avoided to limit the potential for overfitting, which restricts the applicability of polynomial regression [26].

In our study, we aim to develop meta-features that can effectively differentiate the performance of more general models without assuming a specific structured form for the unknown regression function. To quantify the degree of nonlinearity in our data, we compare the mean square error (MSE) obtained from fitting models that assume linearity to those that allow for nonlinear relationships. Specifically, we use MLR as our baseline model for linearity. As for nonlinearity, we seek algorithms that do not require manual specifications for various nonlinear transformations on individual variables. Also, the selected algorithm must respond to nonlinearity and its unique characteristics while avoiding any reaction to additional complexities associated with other properties, such as interactions among variables. Consequently, we opt to explore extensions of linear models to detect nonlinearity without influence from non-additivity. A GAM requires determining the appropriate degree of smoothness by generalized cross-validation or by restricted maximum likelihood (REML), and we consider both methods when developing our measures in the following subsections. MARS offers computational efficiency advantages over GAMs [27], but in cases involving complex nonlinear relationships that require smoother curves, MARS might be less effective.

3.1.1 NL.1 - GAMs Measure

Regarding the selection of smoothing parameters, the Generalized Cross-Validation (GCV) method is widely employed for prediction, as it utilizes prediction error-based criteria to make such selections. This approach is particularly advantageous when dealing with a large number of predictors due to its computational efficiency. Meanwhile, Wood [28] introduces the method of Restricted Maximum Likelihood (REML), treating the smooth components as random effects, and smoothing parameters are chosen based on the likelihood of the model residuals. In his analysis, REML outperforms GCV in terms of mean square error. Moreover, it effectively addresses the issue of potential severe undersmoothing failures commonly observed with GCV. Importantly, these performance enhancements are achieved without increasing computational cost relative to GCV.

We construct two metrics, denoted as `NL.1-gam.REML` and `NL.1-gam.GCV`, indicating GAM with different methods of choosing smoothing parameters. These metrics are defined as the ratio of MSE obtained from the MLR model to that from the GAM model. A higher ratio indicates a more pronounced deviation from linearity and suggests a stronger presence of nonlinearity in the data.

3.1.2 NL.2 - MARS Measure

To explore nonlinearity in another manner, we adopt the MARS model with a constraint of one degree of interaction. By restricting the upper limit of interaction to one, MARS

becomes an additive model that is suitable to measure nonlinearity while employing a different spline approach compared to GAMs.

Similar to `NL1-gam`, the metric `NL2-mars` is defined as the ratio of the MSE obtained from the MLR model to the MSE obtained from the one-degree MARS. All of these measures are relative-landmarking-based meta-features.

3.2 Interactivity

Interactivity refers to a situation where the effect of one explanatory variable on the response is influenced by the values of other independent variables. In this case, additive models that contain only terms involving individual variables are insufficient for accurately predicting the response. In linear regression, interactions are typically modeled as cross-products of two or more variables, but their actual structure can be more flexibly modeled in other machine learning algorithms. Model interactions using cross-products require either knowing which cross-product terms are needed or including all of them, resulting in a potential explosion of the number of parameters in the model, and we still assume linear relationships between the response and each explanatory variable. We can modify GAM models in a similar way by replacing linear cross-products with smoothing spline on the cross-products. Again, this requires knowing which terms to include or fitting a potentially enormous model. Consequently, we prefer algorithms like MARS, which possess both interactive and additive versions and automatically capture nonlinearity and interactivity effectively according to where it appears to be needed.

3.2.1 INT.1 - F-test Measure

In the context of linear regression, one common approach to evaluate the significance of interaction terms is to use an Analysis of Variance (ANOVA) to conduct an F-test. This test involves comparing the RSS between a multiple regression model that only contains the main effects with a larger model that includes both main effects and two-way interaction terms. The first meta-feature, denoted as `INT.1-F`, is the F statistic from the ANOVA F-test. A higher value of the F-statistic suggests that incorporating interaction terms in the model provides valuable additional information, enabling a better explanation of the relationship between predictors and the response variable. Consequently, interactivity exists.

3.2.2 INT.2 - MARS Measure

In scenarios where nonlinearity and interactivity may both exist in the response surface, an alternative measure based on MARS may be more responsive and useful. For the detection of interactivity, MARS allows the flexibility of controlling an upper limit on the order of interaction. By setting this upper limit, we can control the consideration of cross-products within piecewise linear functions. Specifically, an upper limit of 2 permits the inclusion of

second-order interactions, while an upper limit of 1 corresponds to an additive model with no interactions [18].

Hence, `INT.2-mars` [10] is calculated by dividing the MSE obtained from MARS with no interaction by the MSE obtained from MARS with second-order interactions.

3.3 Heteroskedasticity

Heteroskedasticity, which refers to the existence of non-constant error variance in regression analysis, violates one of the key assumptions of linear regression, leading to inefficiency in the OLS estimates. In general, in situations where algorithms do not account for heteroscedasticity, certain data points with higher error variance might be assigned excessive weight. The model tends to accommodate noisy and uncertain observations with greater variability, leading to inaccurate predictions.

Despite the importance of this issue, there are relatively few existing papers that study the impact of heteroskedasticity on prediction accuracy across different algorithms. In Gelfand's study [29], she evaluates the effect of heteroskedasticity on the prediction performance of various predictive methods. The study finds that tree-based methods perform relatively less well than other methods on heteroscedastic data than on data with constant variance.

3.3.1 HET.1 - Standard Deviation (SD) Ratio Measure

The 'Standard Deviation (SD) Ratio' measure was introduced by Gelfand [29] to quantify heteroscedasticity. This measure aims to address the challenge of explaining inequality in error variance when the model is unknown and/or the dimension is large. To calculate the SD Ratio, we first fit the data using a default Random forest model. RF is chosen for its ability to effectively adapt to various surface patterns in data with arbitrary dimensions. Then, we obtain the prediction residuals from the RF model using the out-of-bag sample. By treating the residuals e_i as a proxy for the error term ϵ_i, the absolute values of the residuals, $|e_i|$, are considered to represent the standard deviation. The `HET.1-sdRatio` is calculated by dividing the average absolute residuals for the largest 10 percent of predicted responses ($\hat{Y}$) by the average for the smallest 10 percent of $\hat{Y}$. In cases of homoscedastic data, this ratio tends to be close to 1 to indicate constant variance in $\hat{Y}$. However, if the variance increases with the mean, which is a very common manifestation of heteroscedasticity, the ratio is expected to be greater than 1. Conversely, if the variance decreases with the mean, the ratio will be less than 1.

3.3.2 HET.2 - Breusch Pagan Test Measure

Various tests for heteroscedasticity have been developed for linear regression models. A classic one is the Breusch–Pagan test, which examines whether the variance of the errors is associated with any of the explanatory variables.

To conduct this test, the linear regression model is first fitted under the assumption that the error variance is independent of the explanatory variables. Squared residuals are then calculated to estimate the error variance. Next, a new regression model is fitted using the squared residuals as the response values. The purpose of this step is to test whether there is a significant linear relationship between the squared residuals and the independent variables, which is essentially a chi-square test. The distribution of test statistic nR^2 approaches the χ^2_{p-1} distribution asymptotically. `HET.2-bp` is defined as the p-value associated with this test statistic. If this p-value is found to be lower than a pre-selected significance level (e.g., $\alpha = 0.05$), it indicates the existence of heteroscedasticity.

However, the limitation of the Breusch–Pagan test is obvious as it only detects linear associations between the squared residuals and explanatory variables. An alternative commonly used test called the White test expands upon the Breusch-Pagan test by considering quadratic terms and cross-product interactions among the predictors [30]. Nevertheless, we decide to use the Breusch-Pagan test due to potential issues with degrees of freedom. When there is a large number of explanatory variables, the White test intends to include an excessive number of terms in the regression equation involving linear, quadratic, and interaction terms.

3.4 Signal Strength

Signal strength, adapted from the concept of the signal-to-noise ratio (SNR), quantifies how much of the variability in a set of data comes from changes in the response surface as opposed to the variability of the errors in the data. Specifically, the SNR is the ratio of the variance of the true mean to the variance of the noise. Higher SNR indicates that the changes in the response surface are more easily distinguished from random noise. In such cases, flexible regression models can be applied to reduce bias without being overly concerned about variance.

3.4.1 SS - SNR RF Measure

Theoretically, a model-free version of SNR can be represented as

$$\text{SNR} = \frac{Var(\boldsymbol{f}(\boldsymbol{X};\boldsymbol{\beta}))}{Var(\boldsymbol{\epsilon})}. \tag{3.3}$$

However, to compute signal strength in practice, a model needs to be fitted to assess the variability explained by the model and the variability that the model fails to capture.

Jiang [10] suggests using a default RF to measure signal strength, as RF models are known for their flexibility in capturing complex relationships in arbitrary dimensions and their ability to adapt to unknown interactions. By fitting a default RF model, the ratio of mean square regression to mean squared error can be calculated as a measure of signal strength. Our measure `SS(snr)-rf` is the ratio of the signal obtained from the variance of the OOB predicted values to the mean squared prediction error, which is computed based on the OOB sample.

3.5 Data Richness

The dimensionality of data plays a crucial role in algorithm selection. As indicated in [31], as the dimensionality increases, an exponential increase in the amount of data is required to maintain model accuracy. This phenomenon is commonly referred to as the curse of dimensionality. In other words, as the dimensionality increases, the prediction error grows more rapidly. This is because higher-dimensional spaces allow for potential more complex relationships that are challenging to capture using simple models. Additionally, with the growth in dimensionality, a fixed number of observations becomes more dispersed in space, leading to less informative data about the underlying response surface and its potential for complexity.

Data richness explores the relationship between the sample size and the number of explanatory variables. In the context of regression analysis, simple algorithms like MLR are commonly used because they are easy to implement, not prone to overfitting, and computationally efficient. By taking on bias and lowering variance, they are particularly useful in situations where there are limited training cases or sparse data. However, when the sample size is large relative to the number of variables, more complex algorithms may perform better. In general cases, data richness guides algorithm selection by signaling when controlling variance maybe more important than controlling bias.

3.5.1 DR Measure

Unlike most measures discussed in this paper, `DR` is a direct characterization measure that does not require fitting a model. `DR` is calculated by $n^{(1/p)}$, which adopts measure of sampling density [10] [18].

3.6 Sparsity

A regression problem is said to be sparse in its variables if the true response surface depends on only a fraction of the available explanatory variables. Recognizing the level of sparsity is particularly valuable when working with high-dimensional datasets. The presence of sparsity implies the reduction of variables, resulting in more data per variable and potentially im-

proved fit with more complex models. Algorithms with the internal capacity to completely ignore unimportant variables may gain an advantage in sparsity settings, whereas methods utilizing all variables, like neural networks and Multiple Linear Regression (MLR), might not be suitable.

To quantify sparsity, it is crucial to establish a connection with variable selection methods with low computational costs. One such method is LASSO Regression, which incorporates regularization by selecting an appropriate parameter λ to balance model complexity and prediction accuracy [18]. Despite the need for parameter tuning, the LASSO algorithm remains relatively fast compared to some other techniques. Some additional variable selection methods embedded in algorithms are capable of handling higher-order interactions of the variables, such as Random forest and MARS. These methods offer variable importance scores, which are scores assigned to explanatory variables based on their usefulness in predicting response. Throughout the training process, these methods iteratively calculate the contribution of predictors and rank the candidate predictors accordingly at the end.

3.6.1 SP.1 - LASSO Measure

As in Jiang's study [10], we use LASSO to measure sparsity, with a chosen λ to minimize the CV error. `SP.1-lasso` is defined as the proportion of coefficients that are constrained to zero in LASSO.

3.6.2 SP.2 - MARS VI Score Measure

MARS extends beyond linear models and has the ability to identify the relevant variables during the training process automatically. In particular, MARS uses backward elimination to progressively select and remove terms that model changes in the surface with one variable. During this process, an explanatory variable can be entirely removed from the model if none of its basis functions significantly contribute to predictive performance. After obtaining a MARS model, three metrics associated with each predictor's importance - *nsubsets*, *GCV*, and *RSS* are provided in `earth` [32]. *GCV* and *RSS* metrics are quantified by the relative amount of decrease in errors, depending on the scale of the response variable, and are therefore not useful for comparing different data sets. Hence, we focus on the *nsubsets* to gauge variable importance. Each subset represents the best set of terms corresponding to a particular model size, ranging from 1 to the size of the selected model. The *nsubsets* measure for each variable is the number of these subsets in which the variable is present. The metric `SP.2-mars.vi` is a model-based measure, calculated by the proportion of *nsubsets* that exceed 0. This approach implies that we consider variables to be important if they are included in at least one of the subsets that are smaller than or equal in size to the final model.

3.6.3 SP.3 - RF Average VI Score Measure

Random forests is a popular technique for assessing variable importance in statistical modelling. Two different metrics are commonly recommended.

The first metric, denoted as *%IncMSE*, measures the impact of excluding each predictor variable on the model's accuracy [33]. In each tree, after we compute the mean squared prediction error with OOB data, denoted as $MSPE_b$, predictor variable X_j is randomly permuted, and the model's performance is evaluated again on the same trees. The variable importance measure for variable X_j is calculated by taking an average of the difference in the MSPE of the original OOB data versus permuted OOB data over all B trees, which can be expressed as follows:

$$VI_p(X_j) = \frac{1}{B}\sum_{b\in B} MSPE_b(X', Y') - MSPE_b(X'_{\text{permute j}}, Y'), \tag{3.4}$$

where X' and Y' represent the corresponding OOB sample.

If a variable is important, the corresponding permuted OOB error will increase substantially [34]. The *%IncMSE* is calculated as the percentage change in the permuted model's performance compared to the original model's performance.

The second metric, *IncNodePurity*, measures the total decrease in RSS in the bootstrap training data accounted for by splits on each variable, averaged across all trees. Variables that contribute more substantially to reducing RSS exhibit higher values. For regression problems, the *%IncMSE* metric is generally preferred as it is based on prediction error, providing a more reliable measure of variable importance compared to the *IncNodePurity* metric. `SP.3-rf.avg.vi` is defined as the proportion of variables that has an importance score calculated by *%IncMSE* above the average.

3.6.4 SP.4 - RF Scaled VI Score Measure

`SP.4-rf.scaled.vi` is an adaptation of `SP.3-rf.avg.vi` that modifies the threshold for important variables and takes into account the variability among trees, which is achieved by adjusting (3.4). To calculate `SP.4-rf.scaled.vi`, the importance scores are standardized by dividing the average loss of accuracy due to the permutation by its corresponding standard error among all trees, analogous to the z score. It is important to note that the distribution of the scaled importance scores is not necessarily a standard normal distribution, but the standardization accounts for the fluctuations in mean squared prediction errors across trees. Hence, the VI score for X_j is adjusted as $\frac{VI_p(X_j)}{\hat{\sigma}}$, and `SP.4-rf.scaled.vi` is defined as the proportion of variables for which the corresponding z score, z_j, exceeds the standard normal critical value $z_{0.05/2}$. The decision is somewhat arbitrary, so using the normal critical value as a threshold makes an approximate parallel to significance.

3.6.5 SP.5 - Boruta.conf. Measure

The algorithm `'Boruta'` provides a feature selection method, aiming to obtain statistically significant relevant variables by introducing extra randomness to the variable importance techniques used in random forests [35]. The process begins by extending the dataset with a minimum of 5 additional shadow attributes, which are copies of existing variables, even if the original set has fewer than 5 variables. These added attributes are then shuffled to remove any correlations with the response variable. Next, a random forest is applied to the extended dataset to compute the importance score for each variable based on *%IncMSE*. The maximum importance score among the shadow attributes serve as a threshold, and variables with an importance score exceeding it are considered potentially important variables. A two-sided t-test is employed to compare the importance score of each variable with the maximum shadow score across all the trees in the forest. Variables with significantly higher scores are labelled as important. To manage the potential complexity of this method, especially with large datasets, a predefined limit is set to restrict the number of runs of fitting an RF with the extended dataset and comparing the importance score [35]. In such cases, the calculation can be stopped prematurely if the importance of variables is unclear, leaving some variables marked as "tentative", but the method still provides valuable information about the significantly relevant variables, with reduced randomness involved. Therefore, `SP.5-Boruta.conf.` is defined as the proportion of confirmed important variables, aiming to identify the features that consistently demonstrate importance across the runs.

3.6.6 SP.6 - Boruta.tent. Measure

As we restricted the number of Boruta run, numerous variables may be labeled as "tentative" due to incomplete confirmation, especially in complex datasets. To address this, the `TentativeRoughFix` method introduces a straightforward test to determine which variables should be confirmed based on whether their median importance exceeds the median importance of the maximum shadow attribute among all runs. Variables that do not meet this criterion are classified as "Rejected." `SP.6-Boruta.tent.` represents the proportion of confirmed important variables after incorporating the tentative fixes.

3.7 Multicollinearity

In regression analysis, multicollinearity is a condition where two or more explanatory variables are correlated with one other. While it is a well-known concern in certain applications of linear models, its significance in the context of machine learning remains less explored. Nevertheless, understanding its impact on linear regression can provide valuable insights that can be extended to more general methods.

The effects of multicollinearity depend on the extent of its presence and the objectives of the study [36]. Strong multicollinearity can cause an increase in the variance of the regression

coefficient estimates, making them more sensitive to even minor changes in the data. For example, in the context of linear models, when X_1 and X_2 are highly correlated, the variation in X_1 is low while X_2 is held constant, which implies a high variance in the regression parameter estimate $\hat{\beta}_1$. One special case is that X_1 and X_2 are perfectly correlated, making it impossible to ascertain the effect of a change in X_1 with X_2 held constant, as there is no variation present. Additionally, it is possible for one explanatory variable X_j to be expressed as a linear combination of other variables. In linear regression, this results in $\boldsymbol{X}$ not being full rank and $\boldsymbol{X'X}$ being non-invertible, thereby making it impossible to compute the OLS estimator. In a broader sense, when sparsity is also a consideration, highly correlated variables may need to be removed to enhance model accuracy by eliminating redundant variables.

If the correlated variables are all important in the response surface, multicollinearity is generally not a major concern for prediction in linear models. If one parameter β_1 is overestimated, another parameter, such as β_2, is typically underestimated to maintain consistent predicted values. In other words, there can exist an infinite number of possible coefficient combinations that yield similar predictions [37][38].

Correlated predictor variables can be a major issue for nonparametric methods in machine learning. For example, Veaux and Ungar [39] conducted a comparative study between the feedforward artificial neural network and multivariate adaptive regression splines (MARS). They found that MARS encounters difficulties in selecting the knots and their basis function when predictor variables are correlated. The forward selection may arbitrarily choose one over the other, which may affect the predictors included in the final model. On the other hand, neural networks, with their redundant architecture, do not suffer from this issue and demonstrate better predictive capabilities in the presence of multicollinearity. Therefore, multicollinearity does appear to have the potential to impact different algorithms in different ways. Measuring it could be useful for recommending algorithms.

3.7.1 MC.1 - Correlation Matrix Measure

In the context where the explanatory variables are standardized, $\boldsymbol{X'X}$ is the matrix of correlations between explanatory variables. An intuitive measure of multicollinearity is the average of the absolute value of off-diagonal elements $r_{jk}, j, k = 1, \ldots, p, j \neq k$, in $\boldsymbol{X'X}$. This measure is defined as `MC.1-avg.cor` $= \frac{1}{p(p-1)} \sum_{j=1}^{p} \sum_{k=1}^{p} |r_{jk}|$, where $j \neq k$. When there is a high correlation between each X_j and X_k, this measure is close to one. Therefore, `MC.1-avg.cor` can be used as a direct characterization measure to indicate the level of multicollinearity, with larger values suggesting a stronger presence of multicollinearity among the explanatory variables.

3.7.2 MC.2 - Condition Number Measure

The condition number of a matrix is a measure of how close to singular the matrix is. A large condition number indicates that the computed inverse of the matrix is sensitive to small variations in the original data. In a general linear model, the condition number of $\boldsymbol{X}$ is often computed on the correlation matrix derived from the standardized $\boldsymbol{X'X}$, and hence is a direct measure of the potential presence of multicollinearity.

Given that $\boldsymbol{X'X}$ is invertible, as required for the calculation in equation (3.2), the matrix possesses p eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_p$. The sum of these eigenvalues is equivalent to the sum of diagonal values of the matrix, which is equal to p since $\boldsymbol{X'X}$ is scaled as the correlation matrix. If one of the p eigenvalues is substantially larger than the others, it suggests that the remaining eigenvalues are relatively small and approaching zero [40], indicating the explanatory variables exhibit high intercorrelation.

Therefore, the condition number measure is defined as `MC.2-condition` $= \sqrt{\frac{\lambda_{max}}{\lambda_{min}}}$, capturing the spread of the eigenvalues in $\boldsymbol{X'X}$ [41]. In general, a condition number less than 10 suggests no significant issue with multicollinearity, while values between 10 and 30 indicate moderate to strong multicollinearity. Condition numbers exceeding 30 indicate severe or nearly perfect multicollinearity [40].

3.7.3 MC.3 - Variance Inflation Factor Measure

The variance inflation factor (VIF) for the jth explanatory variable is determined by taking the inverse of $1 - R_j^2$, where R_j^2 is the coefficient of determination obtained by regressing X_j on the remaining $p - 1$ predictors. VIF represents the proportion of variance in the X_j that can be attributed to the rest of the explanatory variables [42]. In situations where X_j is completely uncorrelated with the other predictors, resulting in $R_j^2 = 0$, the VIF reaches its minimum value of 1. Several rules of thumb for interpreting VIF are presented in the existing literature. In general, if the VIF value exceeds 10, it signals the presence of excessive multicollinearity [42]. This implies that the variance of the regression coefficient associated with the jth explanatory variable becomes 10 times as large when compared to the scenario where the jth explanatory variable is linearly independent of the others. There is no known upper limit for the VIF value.

Summary Table of Meta-features

Table 3.1 provides an overview of meta-features along with the primary functions and R packages utilized for their computation.

Meta-features	Category	Package::Function
NL.1-gam.REML	rel. landmarks	stats::lm; mgcv::gam
N1-gam-GCV	rel. landmarks	stats::lm; mgcv::gam
NL.2-mars	rel. landmarks	earth::earth
INT.1-F	rel. landmarks	car::anova
INT.2-mars	rel. landmarks	earth::earth
HET.1-sdRatio	model-based	randomForest::randomForest
HET.2-bp	model-based	lmtest::bptest
DR	direct	
SS(snr)-rf	model-based	randomForest::randomForest
SP.1-lasso	model-based	glmnet:cv.glmnet
SP.2-mars.vi	model-based	earth::evimp
SP.3-rf.avg.vi	model-based	randomForest::importance
SP.4-rf.scaled.vi	model-based	randomForest::importance
SP.5-Boruta.conf.	model-based	Boruta:Boruta
SP.6-Boruta.tent.	model-based	Boruta:TentativeRoughFix
MC.1-avg.cor	direct	stats::cor
MC.2-condition	direct	base::kappa
MC.3-vif	direct	car::vif

Table 3.1: A summary table of meta-features, along with the corresponding functions and packages used in R.

Chapter 4

Simulation Study

This chapter presents simulation studies aimed at investigating the efficacy of the meta-features proposed in Chapter 3. The main goal is to assess the performance of these meta-features under various experimental settings by manipulating different levels of variations of a baseline dataset. Each experimental setting focuses on a specific data property, and we incorporate key factors that reflect changes in that particular property. Throughout the evaluation process, once the dataset manipulation is performed, other model parameters are adjusted to maintain a constant signal-to-noise ratio. Ten replicate data sets, each containing 800 observations, are simulated from each model, and all proposed meta-features are evaluated on each data set. In each setting, the factor in question is incorporated into the model at various levels, denoted by the flexible parameter m, starting from the baseline model and gradually increasing its intensity. Subsequently, Spearman correlation is computed between the evaluated meta-feature and the level of the property within each setting. If a meta-feature is sensitive and specific to a particular property, it should have a correlation of 1 with the levels of that property and close to 0 for all other properties. To evaluate the performance of each meta-feature, we compute two metrics. The first metric assesses the sensitivity of a meta-feature by calculating the average correlation across all settings within the corresponding data property, using a predefined threshold value of 0.9. This threshold value is chosen arbitrarily to indicate strong correlations. The second metric computes specificity as 1 minus the average correlations for the other 6 properties, also using the same threshold value. While specificity is considered, our study prioritizes sensitivity, since it measures the meta-feature's ability to capture changes in the property it was designed to detect. Such a meta-feature is considered qualified to be included in the set of meta-features used to support the meta-learner in recommending suitable methods.

4.1 Model for Baseline Synthetic Data

Since the baseline is compared with all other settings characterized by specific factors, we simulate a basic multiple linear regression model as our baseline. The model can be referenced from Equation (3.1):

$$Y_i = \beta_0 + (\beta_1 X_{i1} + ... + \beta_p X_{ip}) + \epsilon_i. \tag{4.1}$$

The explanatory variables $X_1, X_2, \ldots, X_p$ are generated from a multivariate normal distribution, $X \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, with p-dimensional mean vector $\boldsymbol{\mu'} = [0, 0, 0, \ldots, 0]$ and a variance-covariance matrix $\boldsymbol{\Sigma} = \sigma_X^2 \boldsymbol{I}$ of size $p \times p$, where σ_X^2 is fixed at 1. The value of β_0 is set to 10 to ensure the values of $f(X_i)$ are positive, as required by some simulation settings, and all other β_j's are set to be 1. Unless otherwise stated, $p = 5$. The error term ϵ_i follows a normal distribution with a mean of 0 and a standard deviation of 1 for the baseline model.

In our study, we pay attention to the value of SNR in all synthetic datasets since most of the proposed meta-features are specifically designed to capture some aspect of the relationship between explanatory and response variables. We strive for a robust SNR, avoiding excessively low values so that signal would not be overwhelmed by random noise. Similarly, excessively high SNR values are undesirable as they indicate unrealistically inflated signals, hindering generalizability. To determine the desired SNR, we use a relationship between SNR and the coefficient of determination (R^2) provided in [43]. The coefficient of determination measures the proportion of the total variance in the response variable that can be explained by the explanatory variables in a model. This explained portion could be thought of as the "signal," and meanwhile, the unexplained variance that remains in the response variable can be regarded as the "noise", which can be expressed as:

$$\text{SNR} = \frac{SSR}{RSS} = \frac{SSR/SST}{RSS/SST} = \frac{R^2}{1 - R^2},$$

where SSR represents the sum of squared regression, which measures the sum of the squares of the deviations of the predicted values from the mean of a response variable; SST represents the total sum of squares, which measures the sum of squared deviations between the observed data and the mean of the response variable. Considering that an R^2 value of 0.8 represents a reasonably strong signal but not totally unrealistic, we decide that an SNR of 4 is an appropriate value for our synthetic baseline dataset.

At the same time, maintaining a consistent SNR across different models is crucial for effectively assessing the impact of manipulation on these meta-features. Since $\beta_1, \ldots, \beta_5 = 1$, the SNR value can be conveniently adjusted by introducing a scaling factor denoted as "c". The resulting model used for adjusting any newly generated dataset is expressed as follows:

$$Y = \beta_0 + c\,(X_1 + X_2 + \ldots + X_p) + \epsilon, \tag{4.2}$$

where c is approximately 0.89 in the baseline model.

4.2 Different Settings for Each Data Property

4.2.1 Nonlinearity

In nonlinearity setting 1, we introduce a modified polynomial regression function to create nonlinearity. With each level, we introduce additional terms that involve higher powers of the corresponding explanatory variables, reaching up to degree 5. To start, in the first level, the functional relationship between Y and X defined in 4.2 is modified as:

$$f(X) = \beta_0 + c\left(\beta_1 X_1 + \ldots + \beta_p X_p + \beta_{p+1} X_1^2 + \ldots + \beta_{2p} X_p^2\right) + \epsilon.$$

As we progress to the m-th level, $f(X)$ is updated to include higher-order terms:

$$f(X) = \beta_0 + c\left(\beta_1 X_1 + \ldots + \beta_p X_p + \beta_{p+1} X_1^2 + \ldots + \beta_{2p} X_p^2 + \ldots + \beta_{p(m-1)+1} X_1^m + \ldots + \beta_{pt} X_p^m\right) + \epsilon.$$

In nonlinearity setting 2, we include a modification to the scalar term of all second-degree polynomial terms. The function $f(X)$ is defined as follows:

$$f(X) = \beta_0 + c\left[\beta_1 X_1 + \ldots + \beta_p X_p + m\left(\beta_{p+1} X_1^2 + \ldots + \beta_{2p} X_p^2\right)\right].$$

We adjust the scalar variable m applied to the second-degree terms in the polynomial, where m varies as (0.01, 0.05, 0.1, 0.3, 0.5, 0.8, 1, 3, 5, 8, 10). This modification allows us to explore the impact of scaling on the quadratic components of the model.

4.2.2 Interactivity

In interactivity setting 1, we introduce all two-way interaction terms. The function $f(X)$ is defined as

$$f(X) = \beta_0 + c\left\{\beta_1 X_1 + ... + \beta_p X_p + m\left[(X_1 \times X_2) + (X_1 \times X_3) + ... + (X_{p-1} \times X_p)\right]\right\}.$$

We consider various values of m: (0.01, 0.1, 0.3, 0.5, 1, 2, 3, 5, 8, 13, 21). By manipulating this scalar, we can examine the increasing effects of the two-way interaction.

In interactivity setting 2, our objective is to investigate the performance interactivity measures in the presence of three-way interactions. This is important because the proposed meta-features for assessing interactivity either calculate the ratio of MSE between the first and second-degree Multivariate Adaptive Regression Splines (MARS) models or utilize an F-like statistic to measure the reduction in MSE when incorporating second-order interactions. The function $f(X)$ in setting 2 is represented as

$$f(X) = \beta_0 + c\left\{\beta_1 X_1 + ... + \beta_p X_p + m\left[(X_1 \times X_2 \times X_3) + ...(X_{p-2} \times X_{p-1} \times X_p)\right]\right\},$$

where there are again 10 added terms and m varies across the same range of values as interactivity setting 1.

4.2.3 Heteroscedasticity

In heteroscedasticity setting 1, we introduce a modification to the true error variance, making it proportional to a power of the mean value, denoted as $f(X)^m$, where $f(X)$ is defined to be positive. For each level, we vary the value of m, which represents a different degree of power applied to $f(X)$, and m can take on the following values: 0.1, 0.3, 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5.

4.2.4 Signal Strength

To manipulate the signal strength within the dataset, we can adapt the regression coefficients β_j using the Signal-to-Noise Ratio (SNR) as defined in (3.3). For each level, we adjust these coefficients by multiplying them with a scalar factor m, which ranges from 0.9 to 0.1 in a decrement of 0.1. This modification allows us to control the SNR in the dataset effectively, and the corresponding SNR values are 4.05, 3.20, 2.45, 1.80, 1.25, 0.80, 0.45, 0.20, and 0.05, respectively, for each value of m. The adjusted expression for the function $f(X)$ is as follows:

$$f(X) = \beta_0 + m \cdot (\beta_1 X_1 + \ldots + \beta_p X_p)\,.$$

4.2.5 Data Richness

To modify the level of data richness, we change the sample size taken from the following values 30, 50, 100, 500, 800, 1100, 1400, where 800 is the baseline.

4.2.6 Sparsity

Our baseline dataset consists of 5 relevant variables. In sparsity setting 1, we introduce m additional non-relevant variables to the dataset. These non-relevant variables are independently drawn from a uniform distribution $U_{[3,5]}$.

The modified form of the function $f(X)$ is given by:

$$f(X) = \beta_0 + c\,(\beta_1 X_1 + \ldots + \beta_p X_p + \beta_{p+1} X_{p+1} + \ldots + \beta_{p+m} X_{p+m})\,,$$

where $\beta_{p+1} = \ldots = \beta_{p+m} = 0$, indicating that the non-relevant variables have no effect on the response variable. For each level, the number of irrelevant variables m varies as (1, 3, 5, 10, 15, 20).

In sparsity setting 2, we use the Friedman function to generate the synthetic dataset, and the focus is on testing cases where the functional relationship between the response variable and relevant explanatory variables is not linear. The modified form of the function

$f(X)$ in this setting is given by

$$f(X) = \beta_0 + c\left(10\,sin(\pi X_1 X_2) + 20(X_3 - 0.5)^2 + 10X_4 + 5X_5 + \beta_{p+1}X_{p+1} + \ldots + \beta_{p+m}X_{p+m}\right).$$

Similar to setting one, for each level, m irrelevant variables are added to the dataset.

4.2.7 Multicollinearity

For multicollinearity setting 1, we focus on testing the ability of measures to detect the existence of multicollinearity. A rough rule of thumb for ascertaining the presence of multicollinearity suggests that if the absolute value of the correlation coefficient between two explanatory variables is above 0.7, it indicates a sign of strong multicollinearity, whereas when the absolute value of the correlation coefficient is between 0.3 and 0.7, the two variables are mildly correlated [41]. Hence, in each level, we keep the correlation coefficients r_{ij} among all variables constant, with values ranging from 0.3 to 0.9 in increments of 0.2.

In multicollinearity setting 2, we fix the value of correlation coefficients to be either 0.3 or 0.8 and manipulate the number of highly correlated variables. Within the matrix $\boldsymbol{X'X}$, there are a total of $\frac{p(p-1)}{2} = 10$ pairs of off-diagonal elements. In this case, we assign a value of 0.8 to m of these pairs, while the remaining pairs are set to 0.3. The value of m can range from 1 to 10, allowing for different degrees of correlation among the variables.

In multicollinearity setting 3, in order to further examine the measures' ability to detect extreme cases of highly intercorrelated variables, we manipulate the correlations among the explanatory variables to vary only at 0.7 and 0.9. Similar to setting 2, we assign a value of 0.9 to m pairs of off-diagonal elements, while the rest are set to 0.7. It is expected that as more explanatory variables approach perfect correlation, the degree of multicollinearity will increase, thereby presenting a different challenge for the measures to accurately detect and quantify the extent of multicollinearity.

Chapter 5

Experimental Results

5.1 Results of Sensitivity and Specificity

In this section, we present the results of our experiments, focusing on the sensitivity and specificity analyses of the meta-features. Our study consists of 7 data properties, and for some of them, we have multiple settings to observe changes. In Appendix Tables A.1 - A.12, we present the simulation outcomes for each setting. We calculated the Spearman correlation coefficients between the evaluated meta-features and the parameter index of manipulated changes. Figure 5.1 summarizes the correlation coefficients by data properties. In cases where a data property has multiple settings, we averaged their correlation coefficients to obtain a single metric. The sensitivity of a meta-feature is determined by its correlation coefficient for its corresponding data property, while the specificity is calculated as 1 minus the average correlation coefficient for the other six properties.

Regarding sensitivity, we found that all meta-features except `MC.2-condition`, `MC.3-vif`, `HET.2-bp`, `INT.2-mars`, `SP.1-lasso`, and `SP.3-rf.avg.vi` achieved the threshold value of 0.9. In the first setting of multicollinearity, the two multicollinearity measures demonstrated nearly perfect sensitivity when we varied the levels of multicollinearity and ensured that all pairs of variables had the same correlations. However, in the second and third settings, where we focused on the measures' ability to distinguish the levels of multicollinearity while keeping the correlation fixed and changing the number of correlated variable pairs, these two meta-features exhibited poor discriminative power. The `HET.2-bp`, depending on the Brausch-Pagan test, was recognized as an effective measure when explanatory variables were treated as factors influencing the variance of the error term, as discussed in [30]. However, in our study, when we examined situations where the variance of the error term changed with the mean, `HET.2-bp` had a relatively lower Spearman correlation. `INT.2-mars` may have experienced challenges due to the presence of three-way interactions in setting 2. The setting involving three-way interactions can introduce complex relationships between variables, which might have affected the ability of `INT.2-mars` with degree 2 to capture and represent these intricate interactions accurately. Additionally, `SP.1-lasso` and `SP.3-rf.avg.vi`

failed to achieve high correlations, which was mainly due to the second setting of sparsity when the functional relationship between X and Y was not linear.

Regarding specificity, only one meta-feature, `MC.1-avg.cor`, had a specificity of at least 0.9 as shown in Figure 5.1. We observed that all meta-features exhibited a strong correlation with changes in the Data Richness case, which represents variations in sample size in our setting. This result was anticipated, as most of the meta-features are model-based or derived from landmarking methods, meaning they depend on fitted models. Consequently, their outputs changed with the sample size. However, this high correlation with Data Richness negatively affected the specificity of almost all meta-features.

	Nonlinearity	Interactivity	Heteroscedasticity	Signal.Strength	Data.Richness	Sparsity	Multicollinearity	sensitivity	specificity
NL.1-gam.REML	0.99	0.25	0.47	0.23	0.90	0.35	0.11	0.99	0.62
NL.1-gam.GCV	0.99	0.57	0.52	0.46	1.00	0.83	0.10	0.99	0.42
NL.2-mars	0.97	0.60	0.64	0.03	0.89	0.61	0.33	0.97	0.48
INT.1-F	0.74	0.93	0.31	0.09	0.43	0.19	0.13	0.93	0.68
INT.2-mars	0.19	0.72	0.15	0.10	0.20	0.84	0.03	0.72	0.75
HET.1-sdRatio	0.85	0.44	0.98	0.26	0.71	0.45	0.04	0.98	0.54
HET.2-bp	0.30	0.45	0.32	0.04	0.37	0.17	0.05	0.32	0.77
SS(snr)-rf	0.76	0.97	0.48	0.99	1.00	0.93	0.96	0.99	0.15
DR	0.00	0.00	0.00	0.00	1.00	1.00	0.00	1.00	0.83
SP.1-lasso	0.81	0.87	0.00	0.52	0.00	0.86	0.10	0.86	0.62
SP.2-mars.vi	0.00	0.28	0.00	0.26	0.66	0.93	0.06	0.93	0.79
SP.3-rf.avg.vi	0.16	0.02	0.46	0.19	0.23	0.75	0.14	0.75	0.80
SP.4-rf.scaled.vi	0.00	0.05	0.00	0.52	0.66	0.98	0.00	0.98	0.80
SP.5-Boruta.conf.	0.00	0.44	0.00	0.56	0.84	0.96	0.00	0.96	0.69
SP.6-Boruta.tent.	0.00	0.38	0.00	0.47	0.66	1.00	0.00	1.00	0.75
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.14	0.17	1.00	1.00	0.95
MC.2-condition	0.00	0.00	0.00	0.00	0.94	0.96	0.69	0.69	0.68
MC.3-vif	0.05	0.05	0.00	0.05	1.00	0.97	0.76	0.76	0.65

Figure 5.1: Results of sensitivity and specificity based on Spearman correlation.

5.2 Relative Change of Meta-features in Different Settings

In this section, we investigate the relative performances of meta-features across various experimental settings. While correlation is a valuable technique that effectively assesses the consistency of meta-feature changes with the manipulation factors in matching data properties, it can be sensitive to minor value fluctuations of the meta-feature. Furthermore,

the extent of changes in meta-feature values across different settings remains unknown. To complement the correlation analysis, we adopt an alternative approach that standardizes each meta-feature's performances across all settings, which is analogous to taking z scores. Specifically, we consider the performance values of one meta-feature in all settings as a dataset and calculate the z score for each performance value with respect to the mean and standard deviation of performance values within that set.

This approach is useful because meta-features lack rules of thumb and standardized units, making it challenging to interpret the explicit meaning of their values or compare their relative changes with other measures. For instance, there are no strict rules for determining the threshold at which the ratio between the MSE of MARS and the MSE of MLR should be considered an indication of severe nonlinearity because these measures depend on the units of measurement for the response.

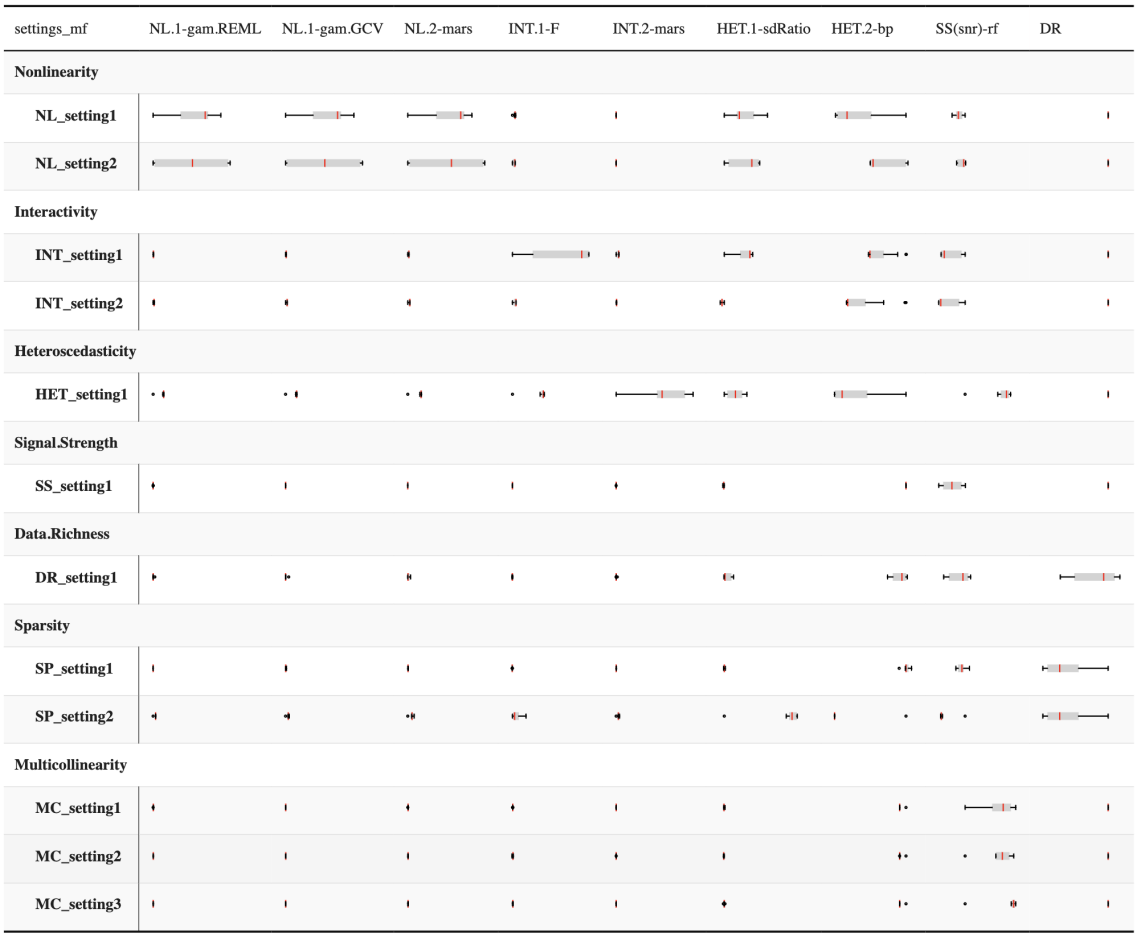

Figure 5.2: Relative change of meta-features in different settings (Part 1).

In Figure 5.2 and Figure 5.3, each boxplot of the z scores provides insights into the properties or manipulated settings that influence changes in meta-feature values. From

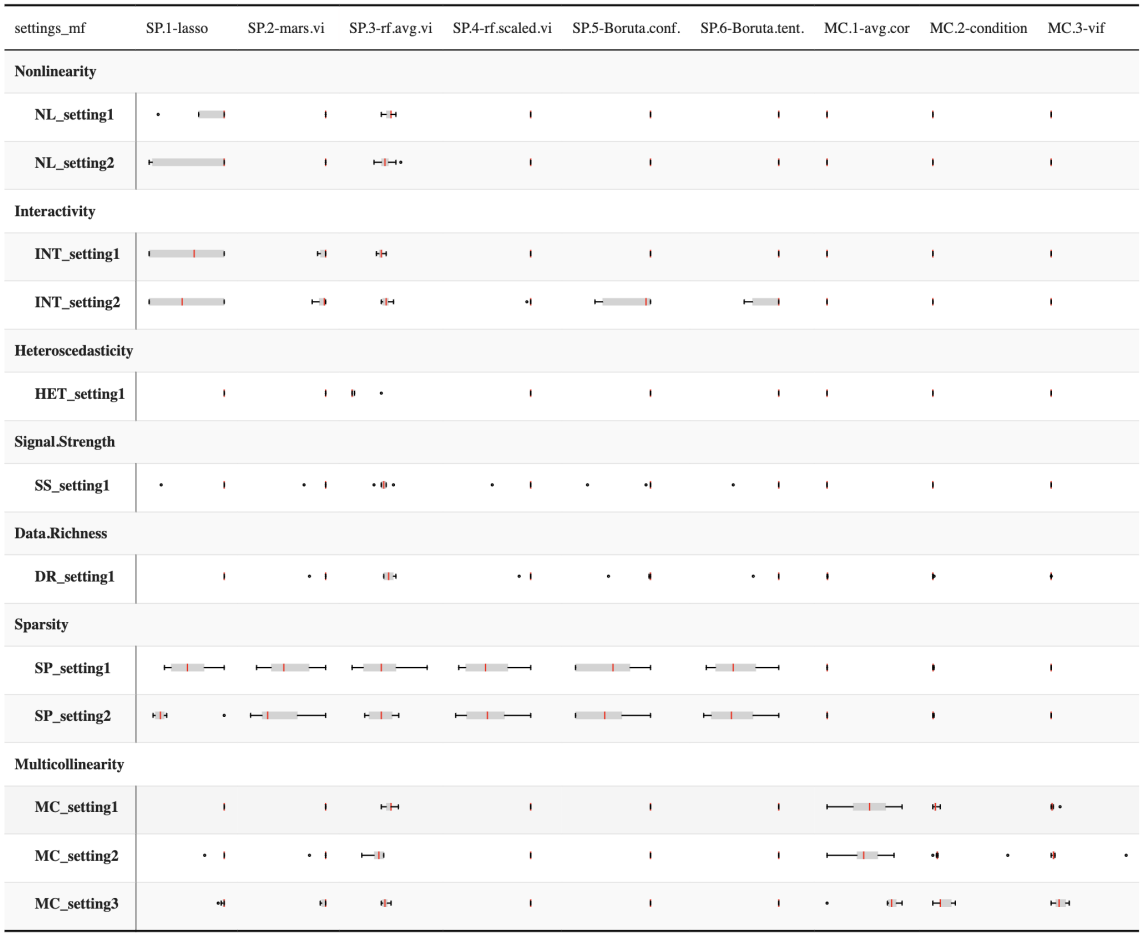

Figure 5.3: Relative change of meta-features in different settings (Part 2).

the figures, it is evident that most meta-features exhibit the longest range of values in settings corresponding to their respective data properties. However, we observed that meta-features associated with heteroskedasticity exhibited inaccurate responses in cases involving nonlinearity and interaction. Surprisingly, `INT.2-mars` changed more dramatically in heteroskedasticity settings than in interactivity settings. Moreover, the random forest-related measures, namely `HET.1-sdRatio`, `SS(snr)-rf`, and `SP.3-rf.avg.vi`, demonstrated less specificity compared to other measures. The random forests are renowned for their ability to capture complex relationships in arbitrary dimensions and adapt to unknown interactions. As a result, they may change unanticipatedly in some scenarios. Another notable finding is that in the data-richness setting, most measures did not experience relatively large changes, indicating their resistance to changes in sample size.

5.3 Computational Time

Efficiency and computational speed are crucial factors when selecting meta-features for practical problems. Therefore, we track the computation time for each meta-feature to assess its computational efficiency.

In Figure 5.4, we present a boxplot illustrating the median computational time for each meta-feature based on 10 replicates across all settings. However, we observed that the two GAM-based measures appear to consume relatively more time and have several large outlying times, corresponding to the nonlinearity settings for which they were designed. To improve overall computational efficiency, it may be prudent to consider removing one of these measures, as long as doing so does not significantly compromise the overall quality of the meta-learner.

To understand the computation time of the remaining measures without the influence of outliers, we scaled the y-axis by restricting the upper limit to 7 in Figure 5.5. This rescaling allowed us to observe that the two measures from `Boruta` consume more time than others. Also, certain measures related to random forests, such as `HET.1-sdRatio` and `SS(snr)-rf`, tend to require relatively more time compared to other measures. However, it is worth noting that certain measures utilize the same algorithm and can be efficiently combined, enabling the model to be fitted only once while extracting necessary outputs. For example, both `HET.1-sdRatio` and `SS(snr)-rf` are model-based features derived from a fitted default random forest. Similarly, the two measures extracted from the `Boruta` function, while applying different rules to define the measures, still require running the function once. This strategic approach significantly reduces the computation time and enhances the overall computational efficiency of the meta-feature analysis.

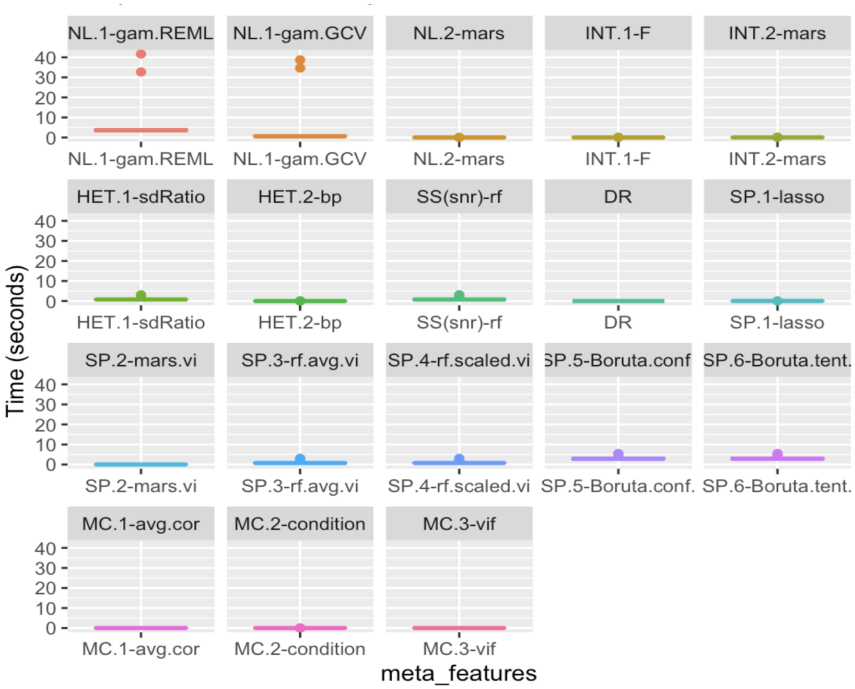

Figure 5.4: Median computational time.

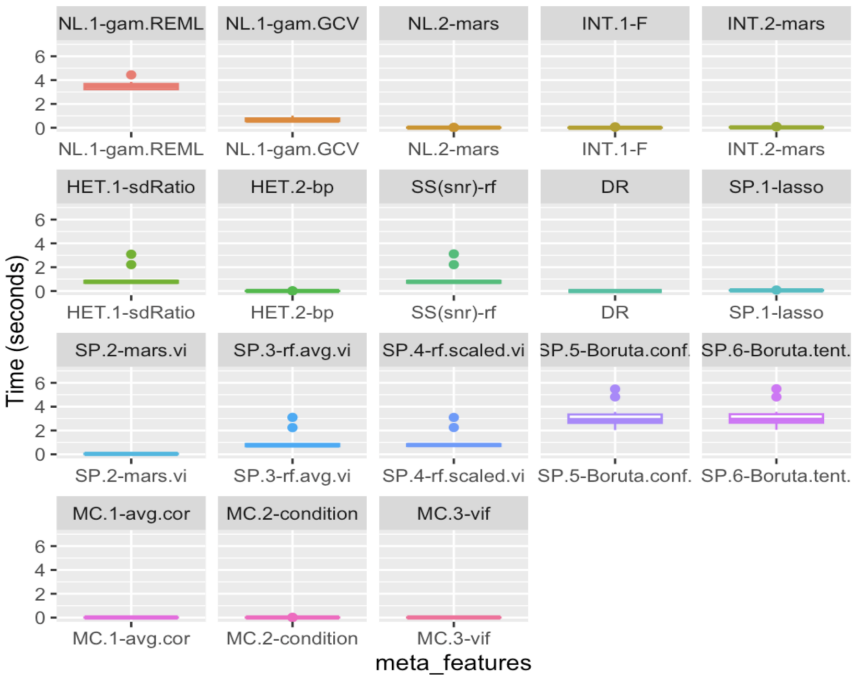

Figure 5.5: Median computational time with scaled time.

Chapter 6

Conclusion and Future Work

In this thesis, we aimed to explore a set of meta-features based on 7 data properties with the potential to influence the prediction performance of supervised learning regression methods. We discussed these data properties and developed measures to effectively recognize and quantify them. Subsequently, we conducted a simulation study to evaluate the performance of these measures in response to manipulated changes within the specified data properties.

The experimental results from our simulation study convincingly demonstrated that most of the measures we used or developed were highly responsive to the corresponding manipulated changes and sensitive to their respective data properties but not specific to them. This lack of specificity can be partially attributed to intercorrelations between some meta-features. For instance, sparsity and data richness may exhibit interdependence, as changing the number of predictors to simulate sparsity automatically impacts data richness. Also, meta-features associated with heteroskedasticity and interactivity are challenging to separate, as both employ random forests and MARS, which can accommodate other changing features such as nonlinearity. The lack of specificity in some meta-features can also be attributed to certain algorithms being adversely affected by the excessive presence of specific data characteristics, such as heteroskedasticity, high nonlinearity, or interactions between variables. `INT.2-mars` unexpectedly performed poorly in the heteroskedasticity setting, possibly because by using OLS as a loss function, MARS with degree 2 may mistake the high error variability in high means as a signal in the data. These findings highlight the need to carefully consider the development and relationships of meta-features when analyzing their performance and relevance in specific data property settings. Despite this lack of specificity, these meta-features remain valuable for a meta-learner as long as they are sensitive to at least one property.

However, there are certain limitations in our study. Firstly, the primary focus on regression analysis means that the explored 7 data properties may have specific relevance to regression learning methods, particularly within the context of linear models. However, their potential effects on more diverse and general data structures have not been extensively explored. There also could be other data properties not covered in our investigation that

significantly impact supervised learning methods, which should be further investigated. As the field of supervised learning continues to evolve, exploring the effects of various data properties on predictive performances can provide valuable insights into the selection and evaluation of appropriate algorithms for different types of datasets and foster advancements in machine learning.

In our simulated baseline model, we deliberately used relatively small sample sizes and a limited number of variables to ensure the feasibility of computations. However, it would be beneficial to explore larger sample sizes and higher variable dimensions to better understand how meta-features behave when dealing with more extensive and complex datasets commonly encountered in practical meta-learning applications. Additionally, while the use of a multivariate normal distribution for generating X allowed us to easily control data properties, particularly for manipulating factors like multicollinearity, considering different data distributions or combinations of distributions when generating datasets could better reflect real-world data scenarios that exhibit diverse distributional characteristics and therefore provide valuable insights into the robustness and applicability of meta-features. Further, our current focus is primarily on numeric variables, and it may be worthwhile to incorporate categorical variables into the analysis.

Currently, our simulation study heavily focuses on variations based on linear models, and we only applied some non-linear transformations and interaction cross-products to X while maintaining a linear model structure. We did not extensively explore the impact of different nonlinear or interactive models on meta-feature performance. For instance, functions like $f(X;\beta) = \frac{\beta_1 X_1}{X_1+\beta_2}$ cannot be expressed as a linear combination of the input variables. Investigating more complex nonlinear functions and their effects on meta-feature performance could provide valuable insights into the suitability of different meta-features for nonlinear modelling tasks. Also, our simulation settings were designed to apply common and easily controlled changes for each data property. Real data may have different structures, but we cannot simulate them all. By choosing simpler settings, we can manipulate them in a controlled experimental environment and observe their effects clearly.

For future work, we recommend expanding the set of meta-features beyond model-based or landmarking-related features to include traditional meta-features like simple, statistical, and information-theoretic measures introduced in Chapter 2 and in the study by Lorena et al. [2]. These additional meta-features, which can be directly computed from the dataset without requiring hyperparameters, may enrich our understanding when working with real-world datasets. Also, we recommend considering the settings that contain interactive changes in different data properties, like nonlinearity and heteroscakasity and see how the meta-features behave. Looking at the broader context of meta-learning, the next step would involve selecting a meta-learner and examining the meta-features' ability to recommend suitable algorithms for specific datasets, contributing to a more comprehensive and practical meta-learning framework.

Bibliography

[1] Simon Parsons. Introduction to machine learning, second editon by ethem alpaydin, mit press, 584 pp., isbn 978-0-262-01243-0. *Knowledge Eng. Review*, 25(3):353, 2010.

[2] Ana Lorena, Aron Maciel, Péricles Miranda, Ivan Costa, and Ricardo Prudêncio. Data complexity meta-features for regression problems. *Machine Learning*, 107, 01 2018.

[3] Adriano Rivolli, Luís Paulo F. Garcia, Carlos Soares, Joaquin Vanschoren, and André Carlos Ponce de Leon Ferreira de Carvalho. Characterizing classification datasets: a study of meta-features for meta-learning. *arXiv: Learning*, 2018.

[4] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

[5] Subramanian Chandramouli, Saikat Dutt, and Amit Kumar Dāśa. *Machine Learning*. Pearson Education India, 2018.

[6] Ciro Castiello, Giovanna Castellano, and Anna Fanelli. Meta-data: Characterization of input features for meta-learning. pages 457–468, 07 2005.

[7] Włodzisław Duch, Tomasz Maszczyk, and Marek Grochowski. *Optimal Support Features for Meta-Learning*, pages 317–358. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[8] Adriano Rivolli, Luís P.F. Garcia, Carlos Soares, Joaquin Vanschoren, and André C.P.L.F. de Carvalho. Meta-features for meta-learning. *Knowledge-Based Systems*, 240:108101, 2022.

[9] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. Openml: Networked science in machine learning. *SIGKDD Explor. Newsl.*, 15(2):49–60, jun 2014.

[10] Haiyang Jiang. *Understanding and estimating predictive performance of statistical learning methods based on data properties*. PhD thesis, 2020.

[11] Christiane Lemke, Marcin Budka, and Bogdan Gabrys. Metalearning: a survey of trends and technologies. *Artificial Intelligence Review*, DOI: 10.1007/s10462-013-9406-y, 06 2013.

[12] Alexander Tornede, Lukas Gehring, Tanja Tornede, Marcel Wever, and Eyke Hüllermeier. Algorithm selection on a meta level. *Machine Learning*, 112(4):1253–1286, 2022.

[13] Ricardo Vilalta, Christophe Giraud-Carrier, and Pavel Brazdil. *Meta-Learning - Concepts and Techniques*, pages 717–731. Springer US, Boston, MA, 2010.

[14] Pavel Brazdil, Jan N. van Rijn, Carlos Soares, and Joaquin Vanschoren. *Metalearning: Applications to Automated Machine Learning and Data Mining*. Cognitive Technologies. Springer International Publishing, 2022.

[15] Xianghua Chu, Jiayun Wang, Shuxiang Li, Yujuan Chai, and Yuqiu Guo. Empirical study on meta-feature characterization for multi-objective optimization problems. *Neural Computing and Applications*, 34, 05 2022.

[16] Joaquin Vanschoren. *Meta-Learning*, pages 35–61. 05 2019.

[17] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18(2):77–95, 2002.

[18] Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin. The elements of statistical learning: Data mining, inference, and prediction. *Math. Intell.*, 27, 11 2004.

[19] Bernhard Pfahringer. Meta-learning by landmarking various learning algorithms. 05 2001.

[20] Pavel Brazdil, Christophe Giraud-Carrier, Carlos Soares, and Ricardo Vilalta. *Metalearning: Applications to Data Mining*. Springer, 1 edition, 2009.

[21] Yonghong Peng, Peter A. Flach, Carlos Soares, and Pavel Brazdil. Improved dataset characterisation for meta-learning. In Steffen Lange, Ken Satoh, and Carl H. Smith, editors, *Discovery Science*, pages 141–152, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[22] Hilan Bensusan, Christophe Giraud-Carrier, and Claire Kennedy. A higher-order approach to meta-learning. In *Proceedings of the ECML'2000 workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, pages 109 – 117. ECML'2000, 2000. Other page information: 109-117 Conference Proceedings/Title of Journal: Proceedings of the ECML'2000 workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination Other identifier: 1000471.

[23] Sarah Daniel Abdelmessih, Faisal Shafait, Matthias Reif, and Markus Goldstein. Landmarking for meta-learning using rapidminer. 2010.

[24] María José Gacto, Jose Manuel Soto-Hidalgo, Jesús Alcalá-Fdez, and Rafael Alcalá. Experimental study on 164 algorithms available in software tools for solving standard non-linear regression problems. *IEEE Access*, 7:108916–108939, 2019.

[25] Tin Kam Ho and M. Basu. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):289–300, 2002.

[26] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013.

[27] John R. Leathwick, Jane Elith, and Trevor Hastie. Comparative performance of generalized additive models and multivariate adaptive regression splines for statistical modelling of species distributions. *Ecological Modelling*, 199(2):188–196, 2006. Predicting Species Distributions.

[28] Simon N. Wood. Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society (B)*, 73(1):3–36, 2011.

[29] Sharla Jaclyn Gelfand. *Understanding the impact of heteroscedasticity on the predictive ability of modern regression methods*. PhD thesis, 2015.

[30] Oscar Olvera Astivia and Bruno Zumbo. Heteroskedasticity in multiple regression analysis: What it is, how to detect it and how to solve it with applications in r and spss. *Practical Assessment, Research and Evaluation*, 24:1–16, 01 2019.

[31] Bertrand Clarke, Ernest Fokoue, and Hao Helen Zhang. *Principles and Theory for Data Mining and Machine Learning*. Springer Publishing Company, Incorporated, 1st edition, 2009.

[32] Stephen Milborrow. Derived from mda:mars by Trevor Hastie and Rob Tibshirani. Uses Alan Miller's Fortran utilities with Thomas Lumley's leaps wrapper. *earth: Multivariate Adaptive Regression Splines*, 2023. R package version 5.3.2.

[33] Carolin Strobl, Anne-Laure Boulesteix, Achim Zeileis, and Torsten Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics*, 8(1), January 2007.

[34] Brandon M. Greenwell and Bradley C. Boehmke. Variable importance plots—an introduction to the vip package. *The R Journal*, 12(1):343–366, 2020.

[35] Miron B. Kursa and Witold R. Rudnicki. Feature selection with the Boruta package. *Journal of Statistical Software*, 36(11):1–13, 2010.

[36] Andrew F. Siegel. *Practical Business Statistics (Seventh Edition)*, pages 355–418. Academic Press, 2016.

[37] John D Morris and Mary G Lieberman. Multicollinearity's effect on regression prediction accuracy with real data structures. *General Linear Model Journal*, 44:29–34, 2018.

[38] Jamal I. Daoud. Multicollinearity and regression analysis. *Journal of Physics: Conference Series*, 949(1):012009, 12 2017.

[39] Richard D. De Veaux and Lyle H. Ungar. Multicollinearity: A tale of two nonparametric regressions. 1994.

[40] Jong Hae Kim. Multicollinearity and misleading statistical results. *Korean Journal of Anesthesiology*, 72:558 – 569, 2019.

[41] Daniel T Larose. *Data Mining Methods and Models*, chapter 3, pages 93–154. John Wiley Sons, Ltd, 2005.

[42] Robert M. O'brien. A caution regarding rules of thumb for variance inflation factors. *Quality & Quantity*, 41(5):673–690, 2007.

[43] Joram Soch and Carsten Allefeld. Macs – a new spm toolbox for model assessment, comparison and selection. *Journal of Neuroscience Methods*, 306:19–31, 2018.

Appendix A

Simulation Results for All Settings

Meta-features\Levels	baseline	m=2	m=3	m=4	m=5
NL.1-gam.REML	1.00	3.81	2.48	4.65	3.96
NL.1-gam.GCV	1.01	3.79	2.48	4.66	3.98
NL.2-mars	1.01	3.43	2.23	3.75	3.28
INT.1-F	0.84	1.43	1.27	1.58	1.44
INT.2-mars	1.01	1.00	1.00	1.00	1.01
HET.1-sdRatio	1.00	2.02	1.52	2.48	1.43
HET.2-bp	0.57	0.29	0.01	0.10	0.02
DR	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	1.30	1.17	0.99	0.88	0.70
SP.1-lasso	1.00	1.00	1.00	0.12	0.66
SP.2-mars.vi	1.00	1.00	1.00	1.00	1.00
SP.3-rf.avg.vi	0.44	0.48	0.52	0.52	0.56
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.17	1.17	1.17	1.17
MC.3-vif	1.01	1.01	1.01	1.01	1.01

Result table of meta-features in Nonlinearity Setting 1.

Nonlinearity Setting 2 Table PART 1.

Meta-features\Levels	baseline	m=.01	m=.05	m=.1	m=.3	m=.5
NL.1-gam.REML	1.00	1.01	1.03	1.12	1.80	2.65
NL.1-gam.GCV	1.01	1.01	1.04	1.12	1.80	2.64
NL.2-mars	1.01	1.02	1.04	1.12	1.76	2.50
INT.1-F	0.84	0.84	0.85	0.94	1.25	1.36
INT.2-mars	1.01	1.02	1.01	1.01	1.00	1.00
HET.1-sdRatio	1.00	1.03	1.10	1.19	1.69	1.89
HET.2-bp	0.57	0.57	0.58	0.57	0.37	0.32
DR	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	1.30	1.31	1.32	1.32	1.33	1.28
SP.1-lasso	1.00	1.00	1.00	1.00	1.00	1.00
SP.2-mars.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.3-rf.avg.vi	0.44	0.46	0.38	0.48	0.48	0.52
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.17	1.17	1.17	1.17	1.17
MC.3-vif	1.01	1.01	1.01	1.01	1.01	1.01

(a) Nonlinearity Setting 2 Table PART 2.

Meta-features\Levels	m=0.8	m=1	m=3	m=5	m=8	m=10
NL.1-gam.REML	3.60	4.00	4.99	5.10	5.14	5.15
NL.1-gam.GCV	3.58	3.98	4.96	5.07	5.11	5.12
NL.2-mars	3.26	3.56	4.21	4.27	4.30	4.30
INT.1-F	1.42	1.44	1.46	1.47	1.47	1.47
INT.2-mars	1.00	1.00	1.00	1.00	1.00	1.00
HET.1-sdRatio	2.00	2.07	2.17	2.21	2.19	2.19
HET.2-bp	0.30	0.29	0.28	0.28	0.28	0.28
DR	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	1.20	1.15	0.96	0.94	0.91	0.91
SP.1-lasso	1.00	1.00	0.08	0.00	0.00	0.00
SP.2-mars.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.3-rf.avg.vi	0.48	0.60	0.46	0.56	0.42	0.44
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.17	1.17	1.17	1.17	1.17
MC.3-vif	1.01	1.01	1.01	1.01	1.01	1.01

Result table of meta-features in Nonlinearity Setting 2.

Interactivity Setting 1 Table PART 1.

Meta-features\Levels	baseline	m=.01	m=.1	m=.3	m=.5	m=1
NL.1-gam.REML	1.00	1.00	1.01	1.02	1.03	1.03
NL.1-gam.GCV	1.01	1.01	1.02	1.03	1.04	1.05
NL.2-mars	1.01	1.01	1.02	1.04	1.06	1.07
INT.1-F	0.84	0.88	2.93	7.90	11.44	15.57
INT.2-mars	1.01	1.01	1.08	1.34	1.56	1.76
HET.1-sdRatio	1.00	1.03	1.34	1.96	1.91	1.74
HET.2-bp	0.57	0.57	0.50	0.28	0.27	0.28
DR	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	1.30	1.31	1.27	1.01	0.74	0.41
SP.1-lasso	1.00	1.00	1.00	1.00	1.00	0.98
SP.2-mars.vi	1.00	1.00	1.00	1.00	1.00	0.98
SP.3-rf.avg.vi	0.44	0.42	0.48	0.44	0.40	0.44
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.17	1.17	1.17	1.17	1.17
MC.3-vif	1.01	1.01	1.01	1.01	1.01	1.01

Interactivity Setting 1 Table PART 2.

Meta-features\Levels	m=2	m=3	m=5	m=8	m=13	m=21
NL.1-gam.REML	1.03	1.02	1.02	1.02	1.02	1.02
NL.1-gam.GCV	1.05	1.05	1.05	1.05	1.05	1.05
NL.2-mars	1.06	1.07	1.06	1.07	1.06	1.06
INT.1-F	17.46	17.86	18.06	18.12	18.13	18.14
INT.2-mars	1.77	1.81	1.84	1.80	1.81	1.77
HET.1-sdRatio	1.77	1.87	1.89	1.89	1.98	1.94
HET.2-bp	0.28	0.28	0.28	0.28	0.28	0.28
DR	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	0.26	0.22	0.20	0.19	0.19	0.19
SP.1-lasso	0.22	0.00	0.00	0.00	0.00	0.00
SP.2-mars.vi	1.00	1.00	0.98	0.94	0.94	0.94
SP.3-rf.avg.vi	0.48	0.44	0.46	0.40	0.42	0.42
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.17	1.17	1.17	1.17	1.17
MC.3-vif	1.01	1.01	1.01	1.01	1.01	1.01

Result table of meta-features in Interactivity Setting 1.

Interactivity Setting 2 Table PART 1.

Meta-features\Levels	baseline	m=.01	m=.1	m=.3	m=.5	m=1
NL.1-gam.REML	1.00	1.00	1.01	1.04	1.06	1.07
NL.1-gam.GCV	1.01	1.02	1.03	1.07	1.09	1.10
NL.2-mars	1.01	1.02	1.02	1.06	1.09	1.11
INT.1-F	0.84	0.84	0.98	1.36	1.52	1.62
INT.2-mars	1.01	1.01	1.03	1.09	1.12	1.14
HET.1-sdRatio	1.00	0.98	0.98	0.93	0.89	0.90
HET.2-bp	0.57	0.56	0.39	0.09	0.09	0.10
DR	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	1.30	1.31	1.22	0.84	0.53	0.23
SP.1-lasso	1.00	1.00	1.00	1.00	1.00	0.88
SP.2-mars.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.3-rf.avg.vi	0.44	0.48	0.44	0.54	0.46	0.54
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.17	1.17	1.17	1.17	1.17
MC.3-vif	1.01	1.01	1.01	1.01	1.01	1.01

Interactivity Setting 2 Table PART 2.

Meta-features\Levels	m=2	m=3	m=5	m=8	m=13	m=21
NL.1-gam.REML	1.07	1.06	1.06	1.05	1.06	1.06
NL.1-gam.GCV	1.10	1.10	1.10	1.10	1.10	1.10
NL.2-mars	1.10	1.10	1.10	1.10	1.10	1.11
INT.1-F	1.65	1.66	1.66	1.66	1.66	1.66
INT.2-mars	1.14	1.13	1.13	1.14	1.12	1.12
HET.1-sdRatio	0.90	0.86	0.88	0.93	0.94	0.92
HET.2-bp	0.10	0.10	0.10	0.10	0.10	0.10
DR	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	0.11	0.09	0.08	0.08	0.08	0.08
SP.1-lasso	0.00	0.00	0.00	0.00	0.00	0.00
SP.2-mars.vi	0.98	0.96	0.96	0.94	0.92	0.90
SP.3-rf.avg.vi	0.50	0.48	0.44	0.50	0.46	0.50
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	0.96
SP.5-Boruta.conf.	0.88	0.64	0.36	0.36	0.30	0.26
SP.6-Boruta.tent.	1.00	0.94	0.72	0.70	0.62	0.62
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.17	1.17	1.17	1.17	1.17
MC.3-vif	1.01	1.01	1.01	1.01	1.01	1.01

Result table of meta-features in Interactivity Setting 2.

Heteroscedasticity Setting Table PART 1.

Meta-features\Levels	baseline	m=0.1	m=0.3	m=0.5	m=1	m=1.5
NL.1-gam.REML	1.00	1.52	1.53	1.55	1.56	1.57
NL.1-gam.GCV	1.01	1.55	1.56	1.57	1.59	1.60
NL.2-mars	1.01	1.53	1.54	1.54	1.56	1.58
INT.1-F	0.84	7.86	7.93	7.99	8.04	8.01
INT.2-mars	1.01	12.27	13.18	14.73	21.57	22.99
HET.1-sdRatio	1.00	1.05	1.08	1.10	1.21	1.29
HET.2-bp	0.57	0.00	0.00	0.00	0.06	0.26
DR	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	1.30	3.42	3.36	3.36	3.37	3.30
SP.1-lasso	1.00	1.00	1.00	1.00	1.00	1.00
SP.2-mars.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.3-rf.avg.vi	0.44	0.22	0.20	0.20	0.20	0.20
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.17	1.17	1.17	1.17	1.17
MC.3-vif	1.01	1.01	1.01	1.01	1.01	1.01

Heteroscedasticity Setting 1 Table PART 2.

Meta-features\Levels	m=2	m=2.5	m=3	m=3.5	m=4	m=4.5	m=5
NL.1-gam.REML	1.58	1.59	1.59	1.58	1.58	1.57	1.56
NL.1-gam.GCV	1.62	1.62	1.62	1.62	1.62	1.61	1.60
NL.2-mars	1.59	1.59	1.60	1.60	1.60	1.59	1.59
INT.1-F	7.93	7.82	7.70	7.56	7.41	7.25	7.09
INT.2-mars	23.58	23.99	19.03	16.00	13.70	14.34	11.64
HET.1-sdRatio	1.38	1.47	1.54	1.62	1.67	1.75	1.78
HET.2-bp	0.33	0.27	0.17	0.07	0.02	0.00	0.00
DR	3.81	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	3.23	3.23	3.15	3.07	2.96	2.95	2.82
SP.1-lasso	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SP.2-mars.vi	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SP.3-rf.avg.vi	0.22	0.22	0.20	0.20	0.20	0.20	0.20
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.17	1.17	1.17	1.17	1.17	1.17
MC.3-vif	1.01	1.01	1.01	1.01	1.01	1.01	1.01

Result table of meta-features in Heteroscedasticity Setting.

Meta-features\Levels	baseline	.9	.8	.7	.6	.5	.4	.3	.2	.1
NL.1-gam.REML	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.01	1.00	1.00
NL.1-gam.GCV	1.01	1.01	1.01	1.02	1.01	1.01	1.01	1.01	1.01	1.01
NL.2-mars	1.01	1.01	1.01	1.02	1.01	1.01	1.02	1.02	1.01	1.01
INT.1-F	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.84
INT.2-mars	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.00
HET.1-sdRatio	1.00	1.00	0.99	0.99	0.99	0.96	0.97	0.96	0.95	0.98
HET.2-bp	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57
DR	3.81	3.81	3.81	3.81	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	1.30	1.32	1.14	0.96	0.78	0.60	0.43	0.28	0.16	0.08
SP.1-lasso	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.16
SP.2-mars.vi	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.84
SP.3-rf.avg.vi	0.44	0.48	0.44	0.44	0.38	0.46	0.48	0.54	0.48	0.46
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.60
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.94	0.16
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.50
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.17	1.17	1.17	1.17	1.17	1.17	1.17	1.17	1.17
MC.3-vif	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01

Result table of meta-features in Signal Strength Setting.

Meta-features\Levels	m=30	m=50	m=100	m=500	m=800	m=1100	m=1400
NL.1-gam.REML	NA	1.23	1.10	1.01	1.00	1.01	1.00
NL.1-gam.GCV	NA	2.76	1.18	1.03	1.01	1.01	1.01
NL.2-mars	1.08	1.12	1.13	1.03	1.01	1.02	1.01
INT.1-F	0.71	0.77	0.91	0.84	0.84	0.79	0.71
INT.2-mars	1.34	1.05	0.98	1.02	1.01	1.01	1.01
HET.1-sdRatio	1.25	1.07	1.32	1.03	0.99	1.00	1.01
HET.2-bp	0.42	0.51	0.55	0.52	0.57	0.58	0.46
DR	1.97	2.19	2.51	3.47	3.81	4.06	4.26
SS(snr)-rf	0.31	0.42	0.55	1.11	1.29	1.45	1.56
SP.1-lasso	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SP.2-mars.vi	0.88	1.00	1.00	1.00	1.00	1.00	1.00
SP.3-rf.avg.vi	0.46	0.50	0.56	0.54	0.48	0.52	0.46
SP.4-rf.scaled.vi	0.88	0.94	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	0.44	0.70	0.98	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	0.72	0.96	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.01	0.01	0.01	0.00	0.00	0.00
MC.2-condition	2.02	1.79	1.48	1.20	1.17	1.17	1.12
MC.3-vif	1.14	1.10	1.04	1.01	1.01	1.01	1.00

Result table of meta-features in Data Richness Setting.

Meta-features\Levels	baseline	m=6	m=8	m=10	m=15	m=20	m=25
NL.1-gam.REML	1.00	1.01	1.01	1.01	1.00	1.00	1.01
NL.1-gam.GCV	1.01	1.02	1.02	1.02	1.04	1.05	1.06
NL.2-mars	1.01	1.02	1.02	1.02	1.04	1.03	1.02
INT.1-F	0.84	0.79	0.75	0.77	0.79	0.78	0.78
INT.2-mars	1.01	1.01	1.02	1.02	1.04	1.06	1.08
HET.1-sdRatio	1.00	0.98	0.98	1.01	1.00	1.01	1.04
HET.2-bp	0.57	0.61	0.59	0.58	0.57	0.56	0.51
DR	3.81	3.05	2.31	1.95	1.56	1.40	1.31
SS(snr)-rf	1.30	1.51	1.16	1.17	1.06	0.90	0.87
SP.1-lasso	1.00	0.83	0.64	0.51	0.33	0.25	0.20
SP.2-mars.vi	1.00	0.93	0.82	0.69	0.65	0.54	0.49
SP.3-rf.avg.vi	0.44	0.82	0.62	0.50	0.33	0.25	0.20
SP.4-rf.scaled.vi	1.00	0.87	0.66	0.53	0.33	0.31	0.25
SP.5-Boruta.conf.	1.00	0.83	0.62	0.50	0.00	0.00	0.00
SP.6-Boruta.tent.	1.00	0.85	0.65	0.50	0.35	0.26	0.20
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.20	1.26	1.33	1.53	1.78	1.99
MC.3-vif	1.01	1.01	1.01	1.01	1.02	1.02	1.03

Result table of meta-features in Sparsity Setting 1.

Meta-features\Levels	baseline	m=6	m=8	m=10	m=15	m=20	m=25
NL.1-gam.REML	1.00	1.13	1.14	1.14	1.14	1.14	1.14
NL.1-gam.GCV	1.01	1.14	1.15	1.16	1.17	1.19	1.21
NL.2-mars	1.01	1.19	1.19	1.18	1.22	1.24	1.29
INT.1-F	0.84	3.90	2.60	1.97	1.32	1.05	0.95
INT.2-mars	1.01	1.66	1.64	1.65	1.93	2.01	2.03
HET.1-sdRatio	1.00	3.12	3.31	3.32	3.48	3.50	3.50
HET.2-bp	0.57	0.00	0.00	0.00	0.00	0.00	0.00
DR	3.81	3.05	2.31	1.95	1.56	1.40	1.31
SS(snr)-rf	1.30	0.25	0.20	0.20	0.19	0.17	0.17
SP.1-lasso	1.00	0.23	0.17	0.15	0.09	0.07	0.05
SP.2-mars.vi	1.00	0.88	0.70	0.57	0.54	0.52	0.44
SP.3-rf.avg.vi	0.44	0.58	0.56	0.50	0.35	0.32	0.30
SP.4-rf.scaled.vi	1.00	0.83	0.62	0.55	0.37	0.30	0.22
SP.5-Boruta.conf.	1.00	0.72	0.52	0.39	0.00	0.00	0.00
SP.6-Boruta.tent.	1.00	0.80	0.64	0.48	0.29	0.22	0.18
MC.1-avg.cor	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MC.2-condition	1.17	1.20	1.26	1.33	1.53	1.78	1.99
MC.3-vif	1.01	1.01	1.01	1.01	1.02	1.02	1.03

Result table of meta-features in Sparsity Setting 2.

Meta-features\Levels	baseline	r=0.3	r=0.5	r=0.7	r=0.9
NL.1-gam.REML	1.00	1.02	1.01	1.01	1.01
NL.1-gam.GCV	1.01	1.03	1.02	1.02	1.02
NL.2-mars	1.01	1.03	1.03	1.03	1.03
INT.1-F	0.84	0.91	0.93	0.90	0.91
INT.2-mars	1.01	1.02	1.01	1.01	1.01
HET.1-sdRatio	1.00	0.98	0.99	0.98	1.03
HET.2-bp	0.57	0.52	0.52	0.52	0.52
DR	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	1.30	2.56	3.07	3.44	3.66
SP.1-lasso	1.00	1.00	1.00	1.00	1.00
SP.2-mars.vi	1.00	1.00	1.00	1.00	1.00
SP.3-rf.avg.vi	0.44	0.52	0.58	0.52	0.48
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.31	0.51	0.71	0.90
MC.2-condition	1.17	2.15	2.85	3.77	6.16
MC.3-vif	1.01	1.26	1.71	2.81	8.37

Result table of meta-features in Multicollinearity Setting 1.

Meta-features\Levels	baseline	n.8=1	n.8=2	n.8=3	n.8=6	n.8=10
NL.1-gam.REML	1.00	1.01	1.01	1.02	1.02	1.01
NL.1-gam.GCV	1.01	1.02	1.02	1.03	1.03	1.02
NL.2-mars	1.01	1.02	1.03	1.03	1.03	1.03
INT.1-F	0.84	0.99	0.72	1.00	0.93	1.02
INT.2-mars	1.01	1.02	1.00	1.00	1.01	1.01
HET.1-sdRatio	1.00	1.00	0.98	0.97	0.98	0.98
HET.2-bp	0.57	0.52	0.52	0.52	0.52	0.52
DR	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	1.30	2.73	2.98	3.09	3.37	3.56
SP.1-lasso	1.00	1.00	0.74	1.00	1.00	1.00
SP.2-mars.vi	1.00	1.00	0.88	1.00	1.00	1.00
SP.3-rf.avg.vi	0.44	0.46	0.38	0.40	0.28	0.46
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.36	0.42	0.46	0.61	0.81
MC.2-condition	1.17	3.57	51.35	3.82	4.56	4.59
MC.3-vif	1.01	1.91	62.90	2.63	3.40	4.20

Result table of meta-features in Multicollinearity Setting 2.

Meta-features\Levels	baseline	n.9=1	n.9=2	n.9=3	n.9=4	n.9=6	n.9=7	n.9=10
NL.1-gam.REML	1.00	1.01	1.01	1.02	1.01	1.01	1.01	1.01
NL.1-gam.GCV	1.01	1.02	1.02	1.02	1.02	1.02	1.02	1.02
NL.2-mars	1.01	1.02	1.02	1.04	1.03	1.03	1.03	1.03
INT.1-F	0.84	0.96	0.91	0.99	0.96	0.93	0.92	0.91
INT.2-mars	1.01	1.02	1.02	1.01	1.01	1.01	1.00	1.01
HET.1-sdRatio	1.00	0.97	1.00	1.00	1.01	1.00	0.99	1.03
HET.2-bp	0.57	0.52	0.52	0.52	0.52	0.52	0.52	0.52
DR	3.81	3.81	3.81	3.81	3.81	3.81	3.81	3.81
SS(snr)-rf	1.30	3.45	3.50	3.55	3.57	3.62	3.63	3.66
SP.1-lasso	1.00	1.00	1.00	1.00	0.96	1.00	0.92	1.00
SP.2-mars.vi	1.00	1.00	1.00	1.00	0.96	0.98	0.96	1.00
SP.3-rf.avg.vi	0.44	0.50	0.52	0.46	0.44	0.46	0.48	0.48
SP.4-rf.scaled.vi	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SP.5-Boruta.conf.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SP.6-Boruta.tent.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
MC.1-avg.cor	0.00	0.73	0.75	0.77	0.79	0.83	0.84	0.90
MC.2-condition	1.17	5.66	16.21	5.66	14.21	6.26	13.19	6.16
MC.3-vif	1.01	4.01	10.65	5.34	15.93	6.78	15.72	8.37

Result table of meta-features in Multicollinearity Setting 3.