Improving Accuracy in Real Estate Valuation: A Study on Data Imputation and Prediction Models

Computing Science Masters Thesis

Hou, Y. (Yuxuan Hou)

First supervisor
Dr. A.A.A. (Hakim) Qahtan

Second supervisor
Dr. H. (Heysem) Kaya

Universiteit Utrecht

Department of Information and Computing Sciences
Utrecht University
Netherlands
August 2023

Abstract

In this study, we aim to assist real estate evaluators in avoiding serious errors during the evaluation process due to carelessness or subjective biases. To achieve this, we primarily employ real Dutch real estate valuation data provided by KATE Innovations and build a data prediction framework with three key components: data imputation, feature selection, and data prediction. Within this framework, we build a data imputation method called "Bucket4Imp" and an improved version of the Fast Correlation-Based Filter (FCBF) as an optional feature selection preprocessing step. For data imputation and data prediction, we adopt a bucket-like ensemble model, incorporating three models: K-Nearest Neighbors, Decision Trees, and Random Forest. Through these model and module configurations, we comprehensively address classification or regression problems involving high dimensional and missing data.

Experimenting with diverse datasets demonstrates the significant enhancement in predictive accuracy achievable through our data imputation preprocessing method. Furthermore, our improved feature selection technique contributes to enhanced predictive performance. However, the decision to employ feature selection should be based on the characteristics of the feature set. The study also conducts a comparative analysis between the machine learning-based Bucket4Imp method and the statistics-based Mode method. The results show that our Bucket4Imp method performs significantly better than the statistical approach, particularly in multi-class scenarios. Moreover, the findings highlighted the challenges of predicting multi-class features, particularly when class distributions are even, and the number of class labels is large.

Keywords— Missing Data, Data Prediction, Feature Engineering, Data Imputation

Acknowledgements

I would like to thank my supervisors, Dr. A.A.A. (Hakim) Qahtan and Dr. H. (Heysem) Kaya, for their full support and academic guidance. Much of the work in this study was inspired by discussions I had with Professor Hakim. I am also grateful to my daily supervisor, Roald Neuteboom, for his patience and helpful hints, both technically and in life. I want to express my appreciation to KATE Innovations for providing me with this amazing project and for their understanding, care, and trust. I would like to thank my mom and dad, Ms. Li and Mr. Hou, for their generous love and support, and their unconditional encouragement have enabled me to focus on my interests and study. Lastly, I am grateful to Gustav Hubert for being there with me during stressful nights and for his understanding, which has been my biggest support.

Contents

Chapter 1

Introduction

As artificial intelligence (AI) continues to gain prominence across various domains, there is an increasing interest in leveraging intelligent tools to enhance daily work processes. The real estate industry, renowned for its significant investments, has witnessed the development of numerous machine learning models aimed at supporting appraisers in their day-to-day operations. For instance, industry giants like Zillow [109] and Redfin [1] have deployed proprietary algorithms to estimate property prices, while HomeUnion's RENTestimate [83] tool assists in determining rental prices. Notably, a McKinsey report titled *"How big data is transforming real estate"* [5] highlights the efficacy of machine learning models in accurately predicting three-year rents per square foot for multi-family buildings in Seattle with over 90% accuracy, by utilizing traditional and alternative data sources.

The focus of predictive modeling in real estate valuation has primarily been on predicting the overall sale price, rather than analyzing the various influencing factors affecting the property from multiple aspects. However, neglecting the critical aspects of real estate valuation analysis can have serious consequences. For instance, in the case of I & I Securities Pty Ltd v R S Melloy Pty Ltd [2002], a developer sought a loan from a lender and hired a valuer to assess the project's value. The developer provided specific information to the valuer, who, unfortunately, failed to verify its accuracy. As a result, the lender relied on the flawed valuation and incurred significant losses. The court held the valuer accountable for negligence in neglecting to conduct independent inquiries. Similarly, in Hann Nominees Pty Ltd v National Australia Bank [2000], another valuer presented a rental valuation. However, this valuer was found negligent for not recognizing that some of the rents paid did not truly represent comparable rental values.

In both cases, it becomes evident that negligence in real estate valuation can lead to adverse outcomes and emphasizes the importance of accurate and thorough analysis by valuers. Thus, analyzing the data itself, beyond predicting the final target, holds intrinsic value, especially in real-world scenarios, where the quality of the data itself may not be guaranteed. When individuals enter data, it often includes human biases or inadvertent errors. This situation can yield two outcomes. Firstly, if the input data is riddled with subjective bias, it can significantly degrade the overall data quality. Consequently, the output generated may be influenced by this flawed data, potentially resulting in inaccurate answers. This could lead to financial losses for the company. Therefore, verifying the data before predicting the final values, such as house prices, is meaningful. Besides, given the disparities in house prices across locations and countries, using a single model to value all real estate properties is impractical.

An important aspect of analysis pertains to the intrinsic correlation among features. In the comprehensive valuation process, numerous features are considered, sometimes exceeding hundreds in real-world cases. These diverse perspectives on real estate analysis contribute to a high-dimensional feature set. Within this feature set, certain features exhibit strong interconnections, while others may contribute minimal information to the analysis. This necessitates a feature selection process that can streamline subsequent model training, mitigate the curse of dimensionality, and enhance the compatibility of the data with a learning model. By adopting this analytical perspective, we can effectively narrow down the feature set and consequently uncover more robust data patterns.

Another essential aspect of analysis lies in understanding the data values themselves. The accuracy of feature values cannot always be confirmed initially, and critical data may be missing due to various reasons. Garcia et al. [32] highlight the occurrence of missing data and potential human biases or mistakes during real estate valuation. Exploring and recovering these missing data points can contribute to improved data prediction and mitigate biases in the real estate valuation process.

Nonetheless, understanding the features of the data can pose various challenges. Valuers may forget to enter valued data into the system or consider certain attributes unnecessary and skip providing the information. Alternatively, they may perceive certain data points as inapplicable to a specific real estate object and omit it. Consequently, the overall quality of the dataset is compromised. This is not an isolated case; it occurs in real-world situations. For example, KATE Innovations, a real estate valuation software company in the Netherlands, possesses numerous real estate object valuation records from various valuers. Some of these records contain features that are important for prediction and hold significant meaning for valuation. Thus, from the company's perspective, ensuring the accuracy of data entered into these features is crucial. Yet, upon analysis, it becomes evident that the data quality in the company's dataset is unsatisfactory, with numerous missing values.

Improving the data quality and recovering missing feature values are crucial tasks in data analysis. In a learning process, we should benefit from the maximum amount of information which should be handled carefully otherwise the learned model could be inaccurate or even incorrect [42].

In this study, our main research task is to analyze various aspects including, data quality, data missing mechanisms, and data recovery methods using real-world case data provided by KATE Innovations. As mentioned above, the significance of this research lies in the fact that real estate valuers must provide accurate and consistent information regarding the Real Estate Object (REO) attributes during the valuation process. Inconsistencies or inaccuracies in the data can diminish trust in the valuation company, hindering transactions. For example, customers may be disappointed if a property is not appraised accurately, such as when the number of bedrooms is incorrectly stated. Such discrepancies may lead to customer complaints during property visits and erode trust in the appraisal company. In extreme cases, significant discrepancies may result in legal actions against the valuation company, leading to substantial financial liabilities as two cases that we show before.

In order to avoid these problems, our goal is to obtain more standardized data and reduce the impact of subjective biases in the source data. We intend to analyze and predict meaningful feature values, which will enable us to identify effective strategies for accurately retrieving missing data. Additionally, we seek to explore appropriate missing data imputation methods to obtain more useful information from the dataset. By utilizing this information, we can provide reminders to the evaluators, thereby minimizing human biases and errors in the process. We also want to build one solution that can be used on different data types. We hope that the subjective data analysis and predictive model can be used to standardize and avoid such issues. Besides, we are studying how the use of machine learning can improve results over simpler methods. In this study, we build, train, and test our data model using the real estate valuation data provided by KATE Innovations.

1.1 Research Questions

Based on the motivation described above, the primary focus of this thesis is to conduct a comprehensive analysis of the data and feature landscape. The aim is to perform essential data imputation and prediction using selected models to forecast significant feature values. These crucial features have been carefully chosen by real estate appraisal experts for their potential value in providing meaningful information. In this study, each attribute of a real estate object (REO) requiring evaluation will be considered a feature, and our goal is to leverage data imputation and predictive techniques to enhance the accuracy of the assessments.

Based on this scope, we set the following main research question.

Main Research Question (MRQ): Given a high-dimensional dataset with missing values (training data) and a set of target features (test data), what techniques can we use to predict the values of the target features accurately?

Based on this question, we can set the following sub-questions:

1. How to deal with missing values in the training data.
2. How to handle the high dimensional dataset in the training data.
3. What models can be used to predict the target features accurately?

1.2 Thesis Organization

The main structure of this thesis is organized as follows: In Chapter 2, Related Work, we introduce the research and relevant models involved in constructing the model for this experiment. In Chapter 3, Theoretical Analysis, we begin with a formal definition of the primary task we are addressing and establish the research objectives. Furthermore, we present further analysis of the approaches taken to address the problems and objectives, providing a clear delineation of the functional modules for problem resolution. We also conduct a more in-depth analysis of related models, explaining their applicability to the current experiment.

In Chapter 4, Methodology, we construct the experimental model based on the feasible results obtained from the theoretical analysis in Chapter 3. Chapter 4 provides a detailed description of the construction process of each module within our model and outlines the algorithmic flow of these modules.

In Chapter 5, Experiment, we evaluate our algorithm and model using the primary dataset from KATE Innovations, along with three other distinct datasets. We meticulously record the experimental results and present them graphically for clarity. Additionally, we analyze the results of each experiment, speculate on the reasons behind the outcomes, and draw conclusions. Due to space limitations, we include some experimental results in Appendix A.

Finally, in Chapter 6, Conclusion, we reiterate and answer the main research question and three sub-research questions mentioned in the Introduction (Chapter 1). We summarize and discuss the findings from the experiments and analysis. Moreover, we highlight potential limitations that might affect the model's performance and suggest avenues for future research.

Chapter 2

Related Work

In this chapter, work related to the research questions and the project process is presented. We start with the techniques that deal with missing data - 2.1.2, followed by work on selecting feature subsets to narrow down dimensionality 2.2, as well as some prediction models in 2.3 that are frequently used in the real estate field.

2.1 Missing Data Imputation

2.1.1 Missing Data Mechanism

In reality, the data results we obtained all have a common drawback - missing or unknown data (e.g. incomplete feature vectors) due to various reasons. Missing data mechanisms describe how data is missing and explain the relationship between the measurement variables and the missing data scenario. Numerous studies have provided detailed descriptions and empirical examinations of missing data mechanisms [86][65]. Generally, missing data mechanisms can be classified into three types: Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Not at Random (MNAR) [27]. MCAR refers to the situation where the probability of missing data in a feature variable is unrelated to the values of other observed variables in the analytical model or the feature variable itself. MAR implies that the missingness of the feature variable is related to other variables but unrelated to the variable itself. MNAR indicates that the probability of missing data is related to the variable itself. In datasets with missing data, all three types of missingness may occur simultaneously, or the missing data mechanism may correspond to only one type.

The missing data pattern provides information about the location of missing data or at least the pattern of missingness, which simply represents the configuration of observed and missing values (MVs) in a dataset [27]. Generally, missing data patterns can be categorized as univariate, monotone, or arbitrary [90]. The univariate missing pattern occurs when only one feature variable contains missing data. The monotone missing pattern occurs when missing values in one variable imply missing values in subsequent variables, forming a staircase pattern of missingness [27]. An arbitrary missing pattern, also referred to as a general missing pattern, indicates that the missing data is completely random - this is the most common scenario of missing data.

Nevertheless, determining the threshold for handling missing data (whether through imputation or direct deletion) has been a recurring question among researchers in various fields. In some studies, a missing percentage of 5% is considered a low threshold, below which the impact of data imputation on the final data analysis is considered negligible [89]. In another article [2], the authors indicate that a missing value percentage of 5% represents the upper limit threshold for large datasets, and exceeding this value may introduce biases in the analysis. Furthermore, some authors have suggested that if the proportion of missing data in a variable exceeds 40%, the results should be treated as hypothetical rather than an accurate representation of the missing data after imputation [25].

As the proportion of missing data increases, only a few studies have investigated the bias and efficiency in datasets.

These studies have typically been conducted with a maximum of 50% missing data, and the results suggest that the estimated variability increases with an increasing proportion of missing values [70][71][23].

2.1.2 Prediction of Missing Data

Handling missing data can be accomplished through two primary methods: one involves directly deleting the missing data, while the other entails employing imputation techniques to estimate and replace the missing values.

The direct approach of dealing with missing values - simply deleting the missing data - is often used when there is a significant number of missing values in a data record or a data feature column. It involves directly removing rows or columns containing missing data, retaining only the available data. However, this method also means discarding available data, which may have adverse effects on the prediction results [61]. Furthermore, in certain cases, some observed data cannot be discarded due to their critical importance. Many such instances occur in the field of medicine. For example, in clinical settings, patients may choose not to disclose certain personal information, or insufficient data may be obtained due to equipment malfunction. In many situations, it is nearly impossible to completely prevent or avoid missing data, and since all observed values are highly valuable as they contain a wealth of information, it is crucial to handle missing data effectively, minimizing the loss of data information caused by its presence [105].

Imputation can be classified into two main forms: single imputation and multiple imputation. Single imputation refers to the process of filling in missing data using a certain method and directly using the obtained results for subsequent calculations and analyses. Some methods include using statistical measures for computation and imputation, such as the mean [41], last observation carried forward, and random imputation. However, these methods can lead to biased results and suboptimal outcomes [59]. For instance, experimental results by Jamshidian and Bentler [40] demonstrate that the direct use of mean for imputation can result in severely biased estimates, even when the data exhibit missing completely at random (MCAR) patterns. This means that if the data has a substantial amount of missing values and these missing values are estimated using the mean of the observed samples, there may be significant discrepancies between the estimated values and the actual values of the variable. Multiple imputation (MI) is a technique used to replace each missing value with a set (greater than one) of plausible values. This approach involves generating several alternative imputations or plausible values to reduce the uncertainty associated with missing data. It entails constructing multiple versions of a unified dataset and calculating optimal estimates by combining results from different imputed versions. By imputing missing data, MI aims to minimize bias in the analysis and account for the potential impact of missing values [91][61] as it better handles missing data by estimating and replacing missing values many times [59].

In order to better analyze how the missing data situation could influence the data imputation, Madley et al. [66] provide guidelines for using multiple imputation (MI) or complete case analysis (CCA) in datasets with a high number of missing values.

Because there is no control over missing data for the researcher or experimenter, the ability to handle missing data has become a fundamental requirement for basic data analysis in order to reduce the impact of improper handling of missing data that may lead to large errors or direct misclassification. Garcia et al. [32] divided the data imputation techniques into two main categories, based on statistical imputation- and machine learning based imputation techniques.

Statistical-based imputation techniques are mean imputation, in which the missing component of a vector is filled using the mean of that component over all observations, and regression imputation, in which the missing component is filled using the predicted values from a regression analysis using the vector components present; however, these two methods do not reflect the uncertainty in the prediction of the unknown values, while multiple imputation does not fill a single value for each missing value, but uses a set of plausible values to replace each missing value, using these values to represent the uncertainty of the correct value to be imputated.

Machine learning-based imputation methods, on the other hand, are numerous and typically involve creating predictive models to estimate the values that will replace the missing values. These methods compute other available information in the dataset to model the missing data. The most commonly used method is K-NN, which is very straightforward - selecting the K nearest neighbors from the complete case, minimizing their similarity

measure, and ranking them in increasing order of distance. We then estimate the alternative values that will replace the missing attribute values. But again, this approach represents a high computational cost, this is because it requires a search of the entire dataset for each record with a missing value in order to find the K-nearest neighbors to complete the data imputation. Fessant and Midenet extend the Self-Organising Mapping (SOM) method to the interpolation of missing data in their work [29], a mapping from a high-dimensional input data space to a low-dimensional mapping space. For each input vector with missing values, we select its image nodes to measure only the distances with known attributes and estimate each missing value based on the weights of the node activation groups in the incomplete attributes. Bengio and Gingras propose the use of recurrent neural networks (RNNs) feedback to input units and estimate missing data [8]. RNNs have an architecture of feedback joins from their units, so in their method, the missing values are first initialized using the mean, and then estimated using the feedback connections from the hidden neurons. The experimental results show that the recurrent network performs significantly better than the standard network. Marwala and Chakraverty[69] use Automatic Associative Neural Networks (AANN) for missing data imputation, using the network to learn from the full case in order to replicate all inputs as outputs. The network does not update the weights when unknown values are detected, but replaces the missing values with the network outputs.

In addition, tree models are frequently used as one of the model types for data imputation methods, such as decision trees and random forests. The advantage of trees lies in their ability to handle different types of data, such as categorical, discrete, and continuous variables. Another benefit of trees is their ability to capture important dependencies and interactions. Furthermore, ensemble models based on trees, such as random forests, can easily handle high-dimensional feature problems and consistently achieve high performance even without fine-tuning their parameters [102].

In the study by Valdiviezo and Aeist [102], they examined the performance of tree-based data imputation techniques in the presence of missing data. They explored two approaches for dealing with missing data: using surrogate decisions with the trees or combining imputation methods with tree-based methods. Their experiments involved different missing data mechanisms and various missing data rates. The experimental results indicated that for smaller proportions of missing data, using ensemble methods combined with surrogates or single imputation is sufficient. However, for moderate to large proportions of missing data, ensemble methods based on conditional inference trees combined with multiple imputation yield the best performance. Their findings demonstrate the potential of ensemble methods with multiple imputation for achieving better predictive performance. Jenhani et al. [42] proposed a non-specificity based possibilistic decision tree (NS-PDT) approach. That is, we use a non-specific based gain ratio as an attribute selection metric which is more appropriate than the standard gain ratio based on Shannon entropy, which allows multiple attributes to be selected in each node to provide a more flexible construction process.

Bayesian Principal Component Analysis (BPCA) method integrates the Expectation-Maximization (EM) algorithm of PCA [82] with Bayesian modeling, introducing a novel missing value imputer [77]. In standard PCA, data points that are far from the training set but close to the main subspace may exhibit similar reconstruction errors. The BPCA method defines a likelihood function that reduces the likelihood of data points far from the training set, even if they are close to the main subspace. This algorithm, specifically developed for missing value estimation, does not emphasize orthogonality among factor loadings. Consequently, the factor weights used to predict missing values differ from those in traditional PCA, leading to an improvement in missing value imputation. Additionally, Bayesian Principal Component Analysis-based MVI BPCAMVI[6] as an unsupervised method, uses probabilistic latent variables applying PCA regression, Bayesian estimation, and EM algorithms. However, this method is not computationally efficient as it applies those three algorithms simultaneously.

Nikfalazar et al. [76] proposed a new method called DIFC by integrating the advantages of decision trees and fuzzy clustering into an iterative learning approach in the context of tree-based data imputation. The imputation technique of DIFC involves iteratively learning new estimates from records with similar attributes determined by the decision tree, using the fuzzy clustering iteration. This missing data imputation method combines supervised learning (decision tree) and unsupervised learning (fuzzy clustering) and iteratively fills in the missing values. Experimental results demonstrate the high performance and robustness of this method. Furthermore, other decision tree models such as Classification and Regression Trees (CART) are compared to Boosted Regression Trees (BRT). Through experimentation and discussion, the authors conclude that the CART model is the preferred choice for

exploratory analysis and predicting missing variables, while the BRT model can explain how the values of other variables influence missing values [100]. Other methods for estimating missing values using decision trees and forests include segmenting the dataset using decision trees and forests to identify segments with higher similarity and attribute correlation. Then, the similarity and correlation of the data are utilized to estimate the missing values within the segments [81]. The authors suggest that high correlation among the features of the dataset may affect the accuracy of data imputation. Therefore, they merge small-sized data segments with appropriate segments to enhance the similarity between records and the correlation between attributes within the merged segments, and then use the records within the segments to estimate numerical and categorical missing values. From their experiments, this method shows better performance than other methods such as Bayesian principal component analysis (BPCA) [77], LLSI [50] and ILLSI [15].

The method of using random forest for imputation is widely used in the handling of missing data. Its ability to handle mixed-type missing data makes it an attractive approach for data imputation. Tang et al. [98] compared Random Forest (RF) imputation with KNN methods and analyzed different RF imputation procedures. They conducted experiments using various missing data mechanisms and found that RF procedures' accuracy depended on the missing data mechanism, decreasing from MCAR to MAR and NMAR. RF imputation showed promising results even with high levels of missingness, reducing error rates compared to a baseline method. The study highlighted the importance of considering data correlation, where RF imputation performed better than KNN in low to medium correlation scenarios. They also compared computational speeds and found that KNN was the fastest, while multivariate RF algorithms improved speed compared to mRF (missForest) algorithms.

Pantanowitz and Marwala compared different paradigms for missing data imputation. They employed five methods for data imputation, namely random forest, auto-associative neural networks with genetic algorithms, auto-associative neuro-fuzzy configurations, and two hybrids based on random forest and neural networks [79]. Experimental results demonstrated that random forest exhibited excellent performance in terms of accuracy and computation time for imputing missing data.

Batista et al. [7] compared the performance of K-Nearest Neighbor (KNN) imputation with CN2 and C4.5 models. They experimented with artificially implanted missing values in large datasets, using cross-validation for error rate estimation. KNN showed good performance in predicting missing values, but its limitations include scalability issues on large databases. They found that attributes with strong correlations could affect the accuracy of KNN predictions, as the inducer might choose alternative attributes with high correlation. The study highlighted the importance of considering attribute correlations when applying KNN imputation.

The Multivariable Imputation by Chained Equations (MICE) method [103] is a regression-based conditional model that uses information from other variables to predict the missing values of interested variables. This method is mainly used by invoking the context between the investigated variable, other measured values and contexts within the investigated variables to make predictions of missing values. However, this method's prediction of missing data is highly dependent on predetermined weights, which are often not optimized, as well as ignoring features from cross-sections with adjacent timestamps (e.g. laboratory test values over a certain period of time). To address these limitations and effectively combine contextual signals within or between the variables under research, while avoiding the reduced effectiveness of learning from available data and the possibility of constructing effective predictive models caused by missing data, Zhang et al. used XGBoost for regression prediction [113]. This is a method that combines an unsupervised pre-filling strategy with a supervised machine learning approach in the form of Extreme Gradient Boosting, to leverage both types of context for imputation purposes. Their interpolation framework starts with unsupervised learning to pre-fill missing values, and they evaluate three pre-filling strategies: global mean, local mean, Iterative Singular Value Decomposition (SVD) and soft impute. From the experimental results, the pre-fill strategy enhanced the predictive ability of their model, and using the simplest pre-fill strategy (global mean) resulted in an average performance of 0.8151, which was 78.1% higher compared to the other two. They also found that a larger window size did not significantly improve imputation. In summary, the expected value-based pre-fill approach provides evidence that XGBoost regression is assisted by better initialization, resulting in faster convergence and better performance, with the model achieving an average improvement of 20% compared to the state-of-the-art approach.

Hasan et al. observed the impact of Missing Value Imputation (MVI) methods on machine learning models in their study. They trained various ML models, including KNN, decision tree, random forest, AdaBoost (AdB),

Naive Bayes, and XBoost, both with and without missing data imputation. Their experimental results showed that when using the XGB classifier, applying data imputation methods led to a 10.9% improvement in the Area under the ROC curve (AUC) [36]. On the other hand, Wang et al. combined a supervised imputation method based on Naive Bayes with oversampling and a random forest classifier in their imputation approach. Their results demonstrated a 5.8% improvement in AUC compared to using a standalone random forest classifier, using 5-fold cross-validation for evaluation [107]. In another study by Christobel and SivaPrakasam, KNN imputation was combined with a KNN classifier, resulting in a 1.5% increase in accuracy when compared to not using data imputation [21]. Maniruzzaman et al. discovered that using median imputation and a random forest classifier with other methods increased accuracy by 1.0% [68]. Furthermore, Kandhasamy et al. performed experiments combining a decision tree-based imputation method with mean imputation, achieving a relative improvement of 12% in accuracy compared to not using data imputation [49].

Of course, although the use of data imputation in the real estate field is not as extensive as in the medical field (for example, Jerez etl al. [44] analyzed the performance of two different data imputation methods in a task, aiming to predict the probability of breast cancer relapse), researchers have still made some interesting findings regarding its application in this domain. For instance, Sanjar et al. [87] employed the K-nearest neighbors algorithm based on the most correlated features (KNN-MCF) to handle missing data in large datasets, and they found that the prediction accuracy of the random forest algorithm was 92.01%. In their experiments, they used the correlation coefficient of two variables to select the most important features. They also discovered that the number of trees required to achieve the desired testing accuracy was 20, and the performance of the algorithm fluctuated between 91% and 92% when using more than 20 trees. Cheng et al. [20] reanalyzed previously obtained data using multiple imputation statistical methods due to significant data differences between groups caused by many participants missing the scheduled visits during the interview records. Through a comparison between standard repeated measures and multiple imputation methods, they successfully identified statistically significant differences in clinical outcomes among three intervention groups that had not been recognized before and proposed intensive community services and housing subsidies as potential factors that could improve clinical outcomes for housing and patients with certain mental illnesses.

Lastly, Thomas and Rajabi [99] conducted a review of all ML-based imputation methods from 2010 to 2020. They found that cluster-based and instance-based algorithms were frequently proposed methods for missing data imputation. The percentage of correct prediction and root mean square error were the most commonly used evaluation criteria in these studies. A common approach was to set the complete dataset as the baseline to assess the effectiveness of imputation on the test dataset presumed to be missing. The type of missing data and the mechanism of missingness were related to imputation capabilities, and the computational cost on large datasets also needed to be considered.

2.2 Feature Selection

With complex feature sets, there are several approaches for feature processing. Accurately finding useful feature sets helps in the construction and calculation of data models. However, complex feature sets also imply problems caused by high dimensionality.

One of the mainstream approach for handling large feature sets is feature selection. The primary focus of feature selection is to choose a subset of variables from the input that effectively describes the input data while reducing the impact of noise or irrelevant variables [18]. In cases where there are numerous features, many variables may be highly correlated with each other, resulting in the use of these feature variables for prediction providing little additional information about the target class. Consequently, this can introduce noise into the prediction process and lead to less accurate results. This can be likened to seeking advice from someone who knows you well rather than from a random stranger on the street.

Generally, feature selection can be categorized into three main types: filter methods, wrapper methods, and embedded methods. Filter methods serve as a preprocessing step that ranks features, selecting the top-ranked features to be utilized by the predictor [54]. Wrapper methods, on the other hand, prioritize feature selection based on their impact on the performance of the predictor. These methods involve constructing the predictor architecture on top of the feature selection algorithm, aiming to find the subset of features that yield the highest

predictive performance. Embedded methods, as the name suggests, incorporate variable selection as part of the training process, eliminating the need to split the data into training and testing sets [11].

In the case of filter methods, this approach primarily employs variable ranking techniques as the criterion for sequentially selecting variables. Variables are scored based on appropriate ranking criteria, and any variables with scores below a threshold are filtered out. In order to identify and discover correlations between features, in statistics, we can use the Pearson product-moment correlation coefficient [80]. This coefficient is used to measure the degree of linear correlation between variables X and Y of two sets of data. The Pearson correlation coefficient between two variables is defined as the score of the covariance of the two variables divided by their standard deviation. However, there are some problems with the commonly used Pearson's correlation coefficient: it only measures linear relationships and does not detect linear relationships between the data, the relationships measured are symmetric, and it cannot handle relationships between non-numerical vectors [104].

To solve this problem, the PPS (Predictive Power Score) [108] is proposed to mine the relationship between features. Suppose we have two columns and want to calculate the predictive power score of A for predicting B. In this case, we have B as our target variable and A as the only feature. We can now choose the appropriate evaluation metric and use a decision tree to perform a cross-validation evaluation. When the target variable is a number, we can use the decision tree regressor and calculate the Mean Absolute Error (MAE). When the target variable is a category variable, we can use the decision tree classifier and calculate the weighted F1 score. Other scores, such as ROC, can also be used. To normalize the results between (0, 1), PPS uses the baseline scores for the cross-validation assessment. When the target variable is a number, we have: $score = 1 - \frac{cv_{score}}{baseline_{score}})$, where cv_score is the cross-validated MAE, and baseline_score is the MAE calculated using the median of the target variable as the predicted outcome. When the target variable is a class, we have: $score = \frac{(cv_{score} - baseline_{score})}{(1 - baseline_{score})}$, where cv_score is the cross-validated F1 value, and baseline_score is the F1 value calculated using the median of the target variable as the prediction result, where both the prediction algorithm and the metrics can be chosen as required.

Kiral et al. proposed a feature selection algorithm based on the filter method, Relief [51], which requires linear time for a given number of features and training instances, uses statistical methods, and avoids heuristic search. This approach devises a correlation statistic to measure the importance of features, which is a vector, each component of which is the evaluation of one of the initial features. The importance of a subset of features is the sum of the correlation statistics corresponding to each feature in the subset, so it can be seen that this correlation statistic can also be considered as the weight of each feature. One can specify a threshold value and simply choose the feature value corresponding to the correlation statistic that is larger than that threshold, or specify the number of features k, and then choose the k features with the largest correlation statistic component. The problem then becomes how to obtain a valid weight or class of relevant statistics to measure the features, and Relief borrows the idea of the hypothesis margin, i.e. the maximum distance that the decision surface can move while keeping the sample classification unchanged. When an attribute is beneficial for classification, the closer the similar sample is to the attribute and the farther the dissimilar sample is to the attribute. Assuming that the interval can evaluate the classification of features on each dimension, it is possible to approximate the subset of features that are most useful for classification, and Relief uses this property for feature selection. Although the algorithm does not guarantee that the smallest subset of features will be found, the subset of features found is still small compared to the original dataset because only statistically relevant features are selected, which greatly improves the learning rate and quality of the results.

However, since the Relief algorithm is only for binary classification problems, in order to solve multi-classification problems, Kononenko et al. made some improvements to Relief and proposed the ReliefF algorithm [56]. The algorithm deals with the multiclass problem by taking one sample at a time from the training sample, then finding k near hits of that sample from the sample set species of the same kind as that sample, and k near misses from a different sample set species, and then updating the weights of each feature. In other words, the algorithm treats multiclassification as a straightforward one-class-to-many-classes solution - finding the nearest neighbors of each class of the current sample and computing them together. Currently, there are many variants of this class of algorithms, such as the one proposed by Robnik and Kononenko, which can be adapted to the regression problem using RRELIEFF [85]. A deterministic neighbour selection method and a new method for incomplete data handling, Relieved-F [54], were introduced by Kohavi et al..

To better determine the relationship between features, we can use mutual information (MI) to effectively represent

the dependence of features, which, as a metric representation of the interdependence of two random variables, is one of the widely used metrics in feature selection [31]. However, since the mutual information is estimated over the entire sampling space, it cannot accurately represent the correlation between features. To address this problem, Liu et al. proposed a new dynamic mutual information-based feature selection algorithm, DMIFS, which is estimated only on unlabelled instances [62]. In order to obtain the exact values of mutual information of candidate features, new labeled instances induced by the previous selected feature should be preserved. The algorithm will re-estimate the mutual information of candidate features with category labels on the set of unlabelled instances before each stage of identifying the desired features. It estimates the mutual information of each candidate feature with a class label for a subset of candidate features. If the candidate feature has zero mutual information, it is immediately discarded from the subset of candidate features and does not contribute to the prediction of unlabelled instances at this point. We then select the feature with the highest mutual information. As changes in the selected features will generate new instance labels, these new instance labels should be temporarily kept down and eliminated from the original instance labels later. This step is intended to prevent the new instance labels from being recalculated at the next estimation of the mutual information. We repeated the process until there were no candidate features in the candidate feature subset or the number of untagged instances was equal to the sum of all the inconsistency counts of partitions. In their experiments, they compared the DMIFS algorithm with the Relief algorithm and used the Information Gain (IG) and Symmetrical Uncertainty (SU) [111] as the baseline, and tested the predictiveness of the selected subsets using four classifiers: Naive Bayesian, 1-NN, C4.5 and RIPPER. From the experimental results, the performance of the DMIFS algorithm used on the classifiers outperformed that of Relief, IG and SU in most cases.

Yu and Liu's Fast Correlation-based Filter (FCBF) feature selection method [112][111] also utilizes symmetrical uncertainty (SU). This approach calculates the association between features and classes using SU and filters the feature values through a threshold to obtain a minimal-optimal subset. The authors conducted experiments to demonstrate that this algorithm outperforms ReliefF in terms of both time complexity and the quality of the obtained optimal feature subset. Senliol et al. extended this algorithm with an alternative search method, introducing the FCBF algorithm [92]. This extension allows FCBF to select feature subsets of any given size and choose features in a different order compared to the original FCBF method. According to their experimental results, this approach yields more accurate classifiers.

Feature selection is also applied in the field of real estate. For instance, Yalpir et al. conducted an investigation in certain regions of Turkey to address the features involved in determining large-scale real estate stock prices. They employed frequency analysis (FRA) for feature selection, along with principal component analysis (PCA), factor analysis (FA), and analytic hierarchy process (AHP) methods. They also utilized multivariate regression analysis (MRA) for the construction of a large-scale real estate valuation system model [110]. Furthermore, Chansit et al. discovered the analytical constraints imposed by high-dimensional data in real estate. They performed feature selection using an improved Garson algorithm, incorporating a combination of enhanced strategies and input sensitivity analysis. This method adjusts its selection criteria with each iteration of the artificial neural network model while maintaining sensitivity coefficients for each informational feature [17]. On the other hand, Horino et al. adopted a different perspective in analyzing real estate-related data and also use the feature selection method from the other perspective. They conducted an analysis of online comments in the Japanese real estate bulletin board system (BBS) and employed an entropy-based approach for keyword extraction and selection [38].

2.3 Data Prediction

One challenge in the field of artificial intelligence and machine learning is that it is difficult to be very sure that the optimal solution can be obtained for a particular type of problem or for a particular feature using a particular model. Scholars are still experimenting, preprocessing data appropriately, finding suitable models, modifying their parameters, and training them to obtain optimal solutions. For general problems, we can divide prediction problems into two main categories, one for classification problems and one for regression problems. Moreover, in the prediction and training process, instead of training a model using all features simultaneously, we can also train a separate model for each individual feature, where this method is known as univariate modeling. In this approach, each model is responsible for predicting the target variable based on a single feature.

For example, Ali Soltani et al. [95] explored the impact of various features on housing price variations using four machine learning models. They found that incorporating the spatiotemporal lag variable significantly improved the predictive accuracy of the models. Lu sifei et al. [64] compared the performance of two regression algorithms and highlighted the benefits of feature selection using Lasso. They discussed various prediction models suitable for multi-featured situations. Feldman et al. [28] utilized the nonparametric Classification and Regression Tree (CART) algorithm to analyze real estate mortgage data, citing its advantages over traditional regression models in handling large datasets, high dimensionality, mixed data types, and missing data. However, they noted some limitations related to the stability of CART trees and its suitability for multivariate predictions. Additionally, Konjes et al. [22] introduced the C4.5-CART model for predicting residential estate prices, which combines numerical and categorical results. They achieved promising results with high accuracy and identified important features for price estimation.

Ensemble learning has been extensively explored in the domain of real estate appraisal. Graczyk et al. [34] conducted experiments using bagging, boosting, and stacking methods with six algorithms to optimize models. While stacking showed low prediction errors, its performance was inconsistent. Bagging proved more stable but yielded comparatively inferior results. Additive regression produced similar outcomes. Consequently, no single algorithm was identified as the best ensemble approach, motivating the search for an optimal hybrid multi-model solution. Furthermore, Paireekreng and Choensawat [78] demonstrated the effectiveness of ensemble techniques and voting algorithms in predicting real estate project outcomes in Thailand, outperforming single models such as neural networks, decision trees, naive Bayes, KNN, and SVM. Similarly, Locurcio et al. [63] highlighted the advantages of ensemble learning, particularly bagging and random forests based on decision trees. These studies collectively emphasize the promise of ensemble learning for real estate prediction tasks.

Chapter 3

Theoretical Analysis

3.1 Problem Statement

A model is proposed to process data in the real estate appraisal process. The whole dataset D is the record of different real estate valuation results. Given a valuation $V = \{X, Y\}$, where $X = \{x_1, x_2, ...x_i\}$ is a given non-target feature set that contains most basic information (e.g, the basic location address); $Y = \{y_1, y_2, ..., y_j\}$ is the target feature set that was manually picked by the dataset owner and that reflects the most important information (e.g, how long a given REO would take to rent out). Lastly, the dataset D is missing many values, both in X and Y.

Given a set of selected prediction models M, and methods for computing the distance between each feature, we create subsets F_j, from the original dataset using these measurements. Each of these subsets targets a specific feature within Y, with every F_j being a subset of X. When a feature y_n is given from Y, we then find those features from the F_j subset that have the highest correlation score with y_n.

The best model M_n from M, which exhibited the highest performance on the validation set for feature X_m in X with missing data x, is selected as the model used to calculate the missing values x within the feature set X using X_n, where X_n is the set of all available subsets of X except for X_m. We then get the updated dataset as X'.

The prediction model M_n demonstrates the highest accuracy in predicting each y_n in Y compared to other models when utilizing this subset F_j.

Thus, we have:

$$\forall X_m \in X, \exists X_n = \{A \subseteq X \mid A \neq X_m\}, M_n \subseteq M, \text{ where}$$
$$argmax_{M,X}(accuracy(M_n(X_n) \rightarrow X_m))$$

also:

$$\forall y_n \in Y, \exists F_n \subseteq X', M_n \subseteq M, \text{ where}$$
$$argmax_{M,X'}(accuracy(M_n(F_n) \rightarrow y_n))$$
$$\wedge$$
$$argmax_{X'}(correlation\_score(F_n \rightarrow y_n))$$

.

3.2 Goals

The goals of this research are to:

- Calculate and impute the missing data for each feature using the most accurate predictions from the best model among all the selected models.
- Identify the subset of sub-features exhibiting the highest correlation coefficient with the target feature.
- Utilize the predictive model with the highest accuracy among all the prediction models to compute and forecast the target features.

3.3 Data Pre-processing

One important preprocessing step for data involves encoding categorical variables. Most predictive models are unable to handle string data due to their inherent nature. However, the presence of such data is inevitable, necessitating the need to convert it into numerical form for better prediction. Additionally, computers or models cannot comprehend data represented by terms like "excellent", "average", or "poor", thus converting them into progressively numeric values can facilitate data analysis and evaluation [57].

Generally, there are several methods available for data encoding, such as one-hot encoding, binary encoding, frequency encoding, target encoding, and label encoding. Among these, Gnat compared several methods and found that large-scale evaluation results vary depending on the encoding technique for categorical variables. Notably, one-hot encoding was proven to be the optimal choice [33]. It is one of the most commonly used encoding methods, where each category is represented by separate binary variables. The value of the corresponding variable is set to 1 if the category is present, otherwise 0. However, one limitation of this method is its inapplicability to high-dimensional data, as it requires encoding each feature with respect to other features, making the computation of all feature dimensions more challenging.

Label encoding is the simplest method for encoding features in machine learning algorithms. In this approach, categorical labels are converted into numerical values, where each category is assigned a specific number. Target encoding, on the other hand, is a technique that encodes categorical features based on their relationship with the target variable. The underlying idea is that if a feature can effectively predict the target, its encoded value should be closer to the target value. Other common methods include frequency encoding, which represents categorical data based on its occurrence count, indicating the frequency of a category within the dataset.

In summary, one-hot encoding is generally the most commonly used method, but it has limitations in terms of dimensionality. Label encoding is suitable when the order of categorical variables is important, the labels are reversible, and it does not increase data dimensionality. Target encoding captures information about labels and potential predictive features without increasing the feature space dimensionality. However, target encoding may lead to overfitting, which can affect subsequent calculations and analysis. Frequency encoding also does not increase feature space dimensionality, but it can only be used when referencing the target variable's count; otherwise, categories with similar cardinality will be assigned the same label [16].

Overall, label encoding, with its non-expansion of data dimensions, sensitivity to the order of categorical variables, and label reversibility, is more suitable for our current research.

3.4 Feature Selection

Using feature selection before the experiment can help us to simplify the model by reducing the number of parameters, decrease the training time, reduce overfilling by enhancing generalization, and avoid the curse of dimensionality [19]. In general, there is a tendency for accuracy to decrease due to feature limitations, but if we select important features through feature selection, we can obtain better model accuracy, i.e., we can perform feature selection in advance to eliminate unimportant variables and improve classification accuracy and performance. Chen et al. [19] conducted experiments using three datasets with numerous variables and compared various feature selection algorithms, including varImp, Boruta, and Recursive Feature Elimination (RFE), in combination with classification

models such as Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbour (KNN), and Linear Discriminant Analysis (LDA). They evaluated 17 different methods, including combinations of classifiers with different feature selections. The results showed that RF methods consistently achieved high accuracy, reaching 98.57% with 561 features in the smartphone dataset. The study highlights the advantages of using Decision Trees, like RF, due to their interpretability and ability to handle both nominal and continuous attributes, making them effective tools for classification tasks.

In our framework, we employed a filter-based feature selection method, which offers the advantages of low computational complexity and the ability to avoid overfitting. Furthermore, this approach is independent of biased learning algorithms, which can be interpreted as adapting the data to fit the learning algorithm [18]. However, this method may encounter issues related to the calculation process of correlation scores and the determination of threshold values, leading to the problem of redundant subsets.

In recent years, state-of-art feature selection methods include Minimum-Redundancy–Maximum-Relevance (MRMR), Fisher Score, Fast Correlation-Based Filter (FCBF), etc.

When selecting relevant features for a target, it is important to consider a particular scenario where two features are both relevant to the target class and provide nearly identical information. In such cases, two methods are commonly employed: 1. The all-relevant method, exemplified by an algorithm called Boruta, selects all the features in question. As depicted in the accompanying figure on the right [115], this method chooses both the mother's IQ and the father's IQ. 2. The minimal-optimal method, represented by an algorithm known as Maximum Relevance, Minimum Redundancy (MRMR), follows a different approach. As the name suggests, this method selects only one of the two features and discards the other. In the figure, it is evident that the IQ feature is considered for the entire set of features [24]. FCBF algorithm also uses the minimal-optimal solution.

From the big picture, for the existing approaches, feature selection can be approached in two main ways: individual evaluation and subset evaluation. In individual evaluation, each feature is assessed and assigned a weight based on its relevance. The top-ranked features from this evaluation are often chosen as a subset of relevant features.

While individual evaluation is efficient for high-dimensional data due to its linear time complexity, it has a limitation. It cannot effectively eliminate redundant features because such features tend to have similar rankings. Consequently, even if features are considered relevant to the target class, many of them may be highly correlated with each other. In cases where high-dimensional data contains numerous redundant features, this approach may yield suboptimal results.

The Figure illustrates the process of subset generation in feature selection. Candidate subsets of features are generated using a specific search strategy. Each candidate subset is evaluated and compared to the previous best subset based on a certain measure. If a new subset is found to be superior, it replaces the previous best subset. This subset generation and evaluation process continues until a predetermined stopping criterion is met.

However, this traditional feature selection framework encounters a challenge when searching through feature subsets during the subset generation step. Most heuristic search strategies, such as greedy sequential search, best-first search, and genetic algorithms, have a time complexity of $O(N^2)$, making them unsuitable for datasets with thousands of features as they do not scale well.

Fast Correlation-based Filter (FCBF) is a filter-based feature selection algorithm proposed by Yu and Liu, specifically designed for handling high-dimensional data [112] [111]. This algorithm employs symmetrical uncertainty (SU) as the correlation measure, replacing the conventional information gain (IG), to assess the relevance or redundancy of a feature with respect to a given classification task. Through experimental evaluations conducted by the authors, it has been demonstrated that FCBF outperforms other algorithms in terms of performance.

To identify a minimal subset of features where the probability distribution of different classes given the feature values closely approximates the original distribution, we need to differentiate and eliminate irrelevant feature sets. John provides definitions for strongly relevant, weakly relevant, and irrelevant feature sets in relation to the target feature [45]:

Let F be a full set of features, F_i a feature, and $S_i = F - \{F_i\}$

Definition 1 (Strong relevance): A feature F_i is strongly relevant iff $P(C|F_i, S_i) \neq P(C|S_i)$.

Definition 2 (Weak relevance): A feature F_i is weakly relevant iff $P(C|F_i, S_i) = P(C|S_i)$, and $\exists S'_i \subset S_i$ such that $P(C|F_i, S'_i) \neq P(C|S'_i)$.

Definition 3 (Irrelevance): A feature F_i is irrelevant iff $\forall S'_i \subseteq S_i$, $P(C|F_i, S'_i) = P(C|S'_i)$.

Furthermore, Yu and Liu proposed a definition for redundant features utilizing the concept of Markov blanket [55] as follows:

Definition 4 (Redundant feature): Let G be the current set of features. A feature is redundant and hence should be removed from G iff it is weakly relevant and has a Markov blanket M_i within G.

Where the Markov blanket is:

Definition 5 (Markov blanket): Given a feature F_i, let $M_i \subset F$ ($F_i \notin M_i$). M_i is said to be a Markov blanket for F_i if and only if $P(F - M_i - \{F_i\}, C|F_i, M_i) = P(F - M_i - \{F_i\}, C|M_i)$.

To provide a more comprehensive representation of the relationships among these sets and the target set of optimal subsets in our feature selection process, we employ the illustrative diagram by Yu and Liu [111] (see Figure 3.1) to enhance the visual presentation.

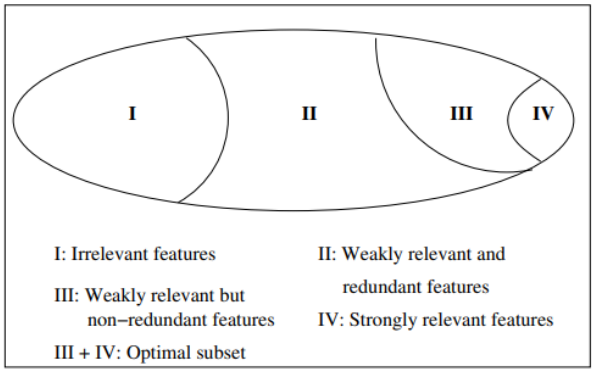

Figure 3.1: A view of feature relevance and redundancy from Yu and Liu

FCBF is a feature selection framework that addresses the issue of feature redundancy by explicitly handling it, thereby enabling efficient elimination of redundant features. This framework consists of two main sections. Firstly, relevance analysis is employed to identify a subset of relevant features by eliminating irrelevant ones. Secondly, redundancy analysis is conducted to identify and eliminate redundant features from the subset of relevant features, resulting in the final subset.

The key advantage of FCBF over traditional subset evaluation frameworks lies in its ability to decouple the processes of relevance and redundancy analysis. This decoupling allows for an efficient and effective approach to finding a subset that closely approximates an optimal subset, bypassing the need for an exhaustive subset search.

FCBF framework uses Symmetrical Uncertainty (SU), which is a measure used to quantify the amount of mutual information shared between two variables. It is commonly used in feature selection and information theory.

The symmetrical uncertainty between variables X and Y is given by:

$$SU(X, Y) = \frac{2 \times MI(X, Y)}{H(X) + H(Y)}$$

where $MI(X, Y)$ is the mutual information between X and Y, and $H(X)$ and $H(Y)$ are the entropies of X and Y, respectively.

The utilization of the Symmetrical Uncertainty (SU) metric by the FCBF algorithm, as opposed to Information Gain (IG), is primarily attributed to SU's characteristic as a normalized representation of information gain. When SU = 1, it indicates a perfect correlation between variables X and Y, implying a causal relationship between X→Y and Y→X. When SU = 0, X and Y are deemed independent. IG, on the other hand, is a non-normalized measure, and its range is uncertain. Additionally, comparing $IG(f_i; f_j)$ and $IG(f_i; C)$ involves different types of comparisons, making a direct comparison challenging. Utilizing SU offers the advantage of normalization, enabling straightforward comparisons between $SU_{i,c}$ and $SU_{i,j}$.

When comparing FCBF with other feature selection algorithms, such as ReliefF [52], CFS-SF [35], and FOCUS-SF [3], which employ both individual and subset evaluation approaches, the results indicate that the FCBF method excels in finding the optimal feature subset. Notably, FCBF(0) consistently outperforms CFS-SF and FOCUS-SF in terms of computational efficiency. This advantage becomes more evident as the dimensionality of the data increases, highlighting the superior computational efficiency of FCBF over CFS-SF and FOCUS-SF. Furthermore, when comparing FCBF(0) with ReliefF, it is observed that ReliefF's performance is unexpectedly slow, primarily due to the computationally expensive distance calculations involved in searching for nearest neighbors. On the other hand, when comparing FCBF(0) with FCBF(log), it is found that setting a larger relevance threshold γ significantly accelerates the FCBF algorithm.

The FCBF feature selection method has been widely utilized in the field of medicine learning, as demonstrated in accurate early prediction of diabetes [53]. In this experiment, the authors employed five machine learning classifiers, namely logistic regression, K-Nearest Neighbors, naive Bayes, Random Forest, and support vector machines, and utilized the FCBF algorithm to eliminate irrelevant features. Through analysis and evaluation of the results, it was observed that a small set of relevant features improved the accuracy of the developed models, with the Random Forest classifier achieving the highest accuracy, sensitivity, specificity, and AUC. Additionally, Muhammad et al. [73] utilized the FCBF feature selection algorithm to construct a preventive model for predicting heart disease, achieving an accuracy of 94.41%. Ridok et al. improved an artificial immune recognition system using the FCBF method as a preprocessing technique, and their proposed method achieved a 100% accuracy for all K settings from 1 to 30 on the Breast-Cancer-Wisconsin dataset through 10-fold cross-validation testing [84].

Although the FCBF feature selection algorithm has been applied in various domains, particularly in medical research as mentioned in the research above, no relevant applications or studies were found in the context of real estate valuation in the present research.

3.5 Prediction Models

3.5.1 K Nearest Neighbor (KNN)

The K-Nearest Neighbors (KNN) algorithm is a non-parametric supervised learning method [30]. This method can be applied to both classification and regression problems. In both cases, the input consists of the K closest training instances from the dataset. However, in classification problems, the output is the class membership of the object. The object is classified based on the majority vote of its neighbors and is assigned to the most common class among its K nearest neighbors (where K is a positive integer defined by the user). If K = 1, the object is simply assigned to the class of the single nearest neighbor. In regression problems, the output is the attribute value of the object, which is the average value of the K nearest neighbors. If K = 1, the object is assigned the value of the single nearest neighbor. Since this method relies on distance calculations, different distance metrics can lead to different classification results and performance. If the features represent different physical units or have specific distance scales, normalization can be applied to the training data to significantly improve the accuracy of the computed results [37].

Specifically, the KNN algorithm assigns unlabeled vectors (query or test points) to a class by assigning the most frequent label among the K nearest training samples that are closest to the query point. For continuous variables, the commonly used distance calculation method is the Euclidean distance.

However, the basic "majority vote" classification has a disadvantage when the class distribution is skewed. Examples from more frequent classes can have a greater influence on the prediction of new instances because they are

more common among the K nearest neighbors due to their larger number. To address this issue, we can introduce weights for the classification.

Moreover, the KNN algorithm can also be used for feature extraction and dimensionality reduction. The dimensionality reduction process can be performed using methods such as Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), or Canonical Correlation Analysis (CCA) as data preprocessing. Feature extraction and dimensionality reduction can be combined into a single step, and then KNN can be used to cluster the features in the space [94].

In summary, KNN is a very intuitive and simple algorithm that is easy to implement. It does not require training because new data entries are labeled based on the values of their nearest neighbors. It can quickly adapt to new data and is applicable to multi-class problems as well as the generalization of classification and regression problems. However, on the other hand, KNN is limited by feature dimensionality and is suitable only for a small number of input variables. As the number of variables increases, the algorithm may struggle to perform well. Additionally, during computation, the features must have the same distance calculation scale, and imbalanced data can lead to misclassification. Moreover, KNN does not handle missing values, meaning the algorithm must be executed on a complete dataset with no missing values.

In practical applications, KNN is often used in real estate-related fields such as house price prediction. For example, Sun et al. employed KNN for regression and classification analysis of household energy efficiency [97]. Mukhlishin et al. compared and utilized fuzzy logic, artificial neural networks, and K-Nearest Neighbors algorithm to find the most suitable model for predicting land and house values in Indonesia [74].

3.5.2 Decision Tree (DT)

Decision Trees are a type of supervised learning method in machine learning that can be applied to both classification and regression problems using classification trees or regression trees, to increase their generalizability. We can select a set of discrete values from the target variable and construct a tree model. In the tree structure, the leaves represent labels, while the branches represent connections between these class label features [75]. A Decision Tree that can take continuous values as the target variable is referred to as a regression tree, which is used to solve regression problems. On the other hand, classification trees are used to predict the class membership of data [96].

Decision Trees have several advantages, for example, they are simple to understand, interpret and visualize, as they implicitly perform variable screening or feature selection. They can handle both numerical and categorical data, as well as multi-output problems, and require relatively little effort from the user for data preparation. Besides, their tree characteristic naturally handles missing values in the data, which allows the implementation of a Decision Tree without requiring additional preprocessing to handle missing values, unlike many other machine learning algorithms. Also, Decision Trees have non-parametric features, which do not make assumptions about the underlying data distribution. This flexibility allows them to be applied to various problem domains without concern about violating distribution assumptions. Their tree-based structure combines features to make predictions, this enables the algorithm to consider interactions and combinations of features, potentially capturing more complex relationships in the data.

However, we can also very easily create over-complex trees that do not generalize the data well, this is also known as overfitting, and can lead to unstable results because small variations in the data might result in a completely different result [72].

In practical applications, Decision Trees have also demonstrated good performance in many usage scenarios. Zhang [114] designed a Decision Tree-based model for predicting house prices, aiming to address the issue of price unfairness arising from different individuals purchasing houses at different prices. They compared this Decision Tree-based model with Support Vector Regression (SVR) and linear regression models, and experimental results indicated that Decision Trees provide an effective solution for house price prediction. Malliaris et al. [67] utilized the Decision Tree approach to study the price trend of gold, incorporating variables such as stock returns and oil prices from the traditional finance industry.

3.5.3 Random Forest

Random Forest (RF), as one of the most commonly used tree-based machine learning models, is a combination of tree predictors, where each tree depends on the values of independently sampled random vectors, and all trees in the forest have the same distribution. As the number of trees in the forest increases, the generalization error of the forest converges to a limit. The generalization error of a tree classifier forest depends on the strength of individual trees and their correlation with each other.

RF, has been highly successful in handling both regression and classification problems [13]. For classification tasks, the output of a Random Forest is the majority-selected class. For regression tasks, the algorithm returns the average or mean prediction of the individual trees. This algorithm follows a simple yet effective "divide and conquer" principle: it samples fractions of the data, grows a randomized tree predictor on each small subset, and then aggregates these predictors together [10].

Compared to Decision Trees, Random Forest can mitigate overfitting on the training set [37]. In high-dimensional feature spaces, the performance of Random Forest is typically better than that of a single Decision Tree, and it also improves upon bagging by decorrelating the trees through random feature subset selection. This means that at each split of the tree, the model considers only a small subset of features rather than all the features in the model. Consequently, this method is widely recognized for its accuracy and can be applied to a wide range of prediction problems without the need for parameter tuning or handling small sample sizes and high-dimensional feature spaces. Currently, this method has been applied to various practical problem domains.

However, due to the black-box nature of Random Forest, there is currently no complete mathematical characterization of the entire process. The most famous result is Breiman's theoretical framework, which provides an upper bound on the generalization error of the forest based on the correlation and strength of individual trees [13]. Within the basic components of Random Forest, bagging [12] and the splitting criteria of Classification and Regression Trees (CART) play crucial roles [14] [10].

Lin and Jeon pointed out the relationship between Random Forests and the K-Nearest Neighbor (KNN) algorithm [60]. They showed that both algorithms can be viewed as so-called weighted neighborhood schemes.

They also demonstrate that a Random Forest's neighborhood shape adjusts according to the local significance of each feature.

Sawant et al. employed the methods of Random Forests and Decision Trees to construct a novel model for predicting housing prices in India. They selected diverse features from the dataset as inputs and demonstrated through experimental results that the performance of Random Forests surpasses that of Decision Tree models [88]. Levantesi and Piscopo applied the Random Forest algorithm to London property data and analyzed the local variables that influence the interaction among housing demand, supply, and prices. They examined the housing prices in central London along with explanatory variables, and the results indicated that Random Forests outperform the traditional regression method of GLM in terms of predictive improvement [58].

3.5.4 Bucket of models

The model bucket is an ensemble technique that utilizes model selection algorithms to choose the best model for each question. The most common method used for model selection is cross-validation, sometimes referred to as a "bake-off contest."

The general process of this approach is as follows: For each model in the bucket, we perform multiple iterations on the training-evaluation set. We train the model on the training set and then test it on the validation set. Subsequently, we select the model with the highest score on the validation set from all the models as the final predictive model. In other words, we attempt all methods on the training set and choose the most effective one [26]. When tested with only one problem, a bucket of models can produce no better results than the best model in the set, but when evaluated across many problems, it will typically produce much better results, on average, than any model in the set.

3.5.5 Bucket Kind of Ensemble Model Analysis

Ensembling multiple machine learning models, such as K-Nearest Neighbors, Random Forest, and Decision Trees, can provide several benefits compared to using a single model. The benefits of ensembling these models include:

Improved prediction accuracy: Each individual model has its strengths and weaknesses. By combining different models, we can leverage their diverse approaches to prediction and improve overall accuracy when using the best performance model in different cases. Different models can capture different patterns and make predictions that may not be possible with a single model alone.

Robustness: Ensemble models are often more robust compared to single models. By always taking the best prediction result from multiple models, the data can always use the best-suited model to do the later prediction.

Handling different types of data and problems: The combination of KNN, Random Forest, and Decision Trees allows for flexibility in handling various types of data. KNN is suitable for numeric data and clustering tasks, while Decision Trees and Random Forest can handle both numeric and categorical data for classification tasks. This ensemble can handle a wide range of data types, making it applicable to diverse prediction problems. Besides, since they can all be used on classification and regression, we do not need to train specific models that suit for only classification or regression questions.

Tackling complex decision boundaries: Decision Trees and Random Forest are capable of modeling complex decision boundaries. By combining them with KNN, which can handle complex decision spaces, the ensemble can effectively capture intricate patterns in the data. This makes the ensemble well-suited for prediction problems with complex relationships and non-linear decision boundaries.

Suitable for various kinds of dimension data: The properties of Random Forests dictate that it can be better insulated from high-dimensional features while still maintaining high feature accuracy; whereas KNN and Decision Tree both perform better on low-dimensional datasets. Therefore, by using them together in a use case, we can train and predict more widely and directly regardless of whether the original data is high or low dimensional. This also means that the model can perform well on unseen data by reducing bias and variance. By aggregating the predictions from different models, the ensemble can provide a more reliable and generalized solution.

The ensemble of KNN, Random Forest, and Decision Trees can find applications in various fields and prediction problems, including but not limited to: Image recognition, financial forecasting (ensembles can be employed to predict stock prices or financial market trends by leveraging the diverse approaches of the individual models. Each model can capture different patterns and contribute to a more accurate prediction), healthcare diagnosis, fraud detection, or customer churn prediction.

In summary, bucket kind of ensembling models combined with KNN, Random Forest, and Decision Trees offer benefits such as improved accuracy, robustness, and flexibility in handling different data types. This ensemble can find applications in various fields and prediction problems, particularly those that require improved accuracy, handling of complex relationships and different prediction problems, or diverse data types.

3.6 Machine Learning-based Imputation

Machine learning-based methods have been compared to statistical methods for data imputation by Jerez et al., and the results demonstrate that machine learning-based approaches are better suited for imputing missing data and can significantly improve subsequent predictive performance [43]. In certain situations, machine learning-based imputation methods can outperform simple imputation techniques or multiple imputation methods. These situations include:

Nonlinear relationships: Machine learning algorithms are capable of capturing complex nonlinear relationships between variables. When missing values are related to intricate patterns or depend on multiple features, machine learning imputation methods can leverage these relationships to make more accurate imputations compared to simple methods relying on summary statistics or predetermined rules [39].

High-dimensional data: Machine learning imputation methods can effectively handle missing values when dealing

with datasets containing a large number of features. These methods can utilize the underlying structure of the data to infer missing values based on patterns and relationships between variables, leading to more reliable imputation results [39].

Complex missing patterns: Machine learning inference techniques can handle missingness exhibiting complex patterns, such as missing data dependent on other variables in the dataset. By considering relationships between variables, machine learning models can more accurately infer missing values in such scenarios [39].

Non-monotonic missingness: In cases where the probability of missingness varies with the values of other variables, machine learning imputation methods can capture these non-monotonic relationships. Simple imputation methods assuming monotonic relationships may fail to accurately impute missing values, whereas machine learning models can handle such complex scenarios [106].

Unknown missing mechanisms: Machine learning imputation methods can be beneficial when the missing mechanism is unknown. These methods do not require explicit assumptions about the missing process and can adapt to different missing patterns and mechanisms [106].

Performance improvement: Machine learning imputation methods have the potential to enhance the overall performance of downstream tasks such as predictive modeling or clustering. By accurately imputing missing values, these methods can provide more complete and reliable datasets, leading to better model performance and more accurate results [39].

However, the machine learning-based imputation methods also depend on the data quality and representativeness of the available data, as well as the imputation algorithm that we picked. Besides, machine learning-based data imputation models may require more computation in terms of time and space due to the computational complexity of the models themselves compared to other single or multiple imputations, and could be sensitive to overfitting if not appropriately validated.

Therefore, careful selection and evaluation of suitable machine learning imputation methods are crucial based on the specific characteristics of the dataset and the desired objectives.

Chapter 4

Methodology

In this section, we will present a brief solution to the problem statement introduced in chapter 3.1. Our proposed solution will consist of two main parts: the calculation of the missing data task and the feature engineering task. A framework of the proposed solution can be found below in Fig.4.1.

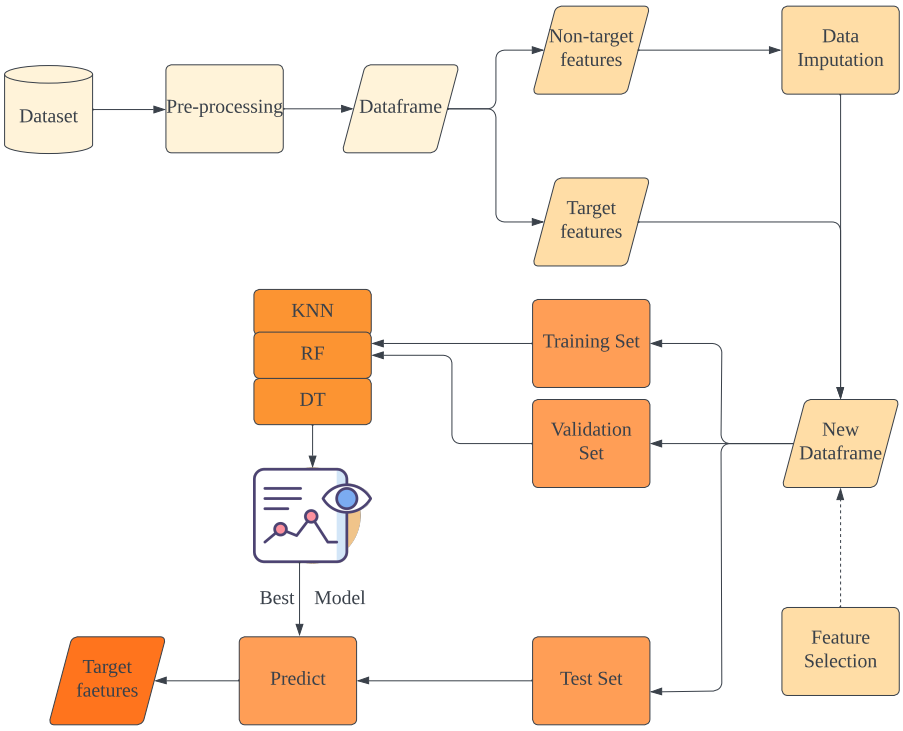

Figure 4.1: The Framework of the Proposed Solution

4.1 Data Pre-processing

In this experiment, we employed the label encoding method for categorical data. This method involves assigning numerical values to categories. The values can be automatically generated by an algorithm in a random manner, or specific numbers can be designated for each category, such as 3 for "excellent", 2 for "good", 1 for "moderate", and 0 for "poor". The advantage of this algorithm is that it allows for assigning values in a progressively meaningful manner, enabling the computer to better understand the order of categories, such as excellent > good > moderate > poor. However, a drawback of this method is that, since the assigned numerical values are limited, different features may end up with the same results. For example, if the convenience of parking for a certain real estate object (REO) is assigned a value of 3 and the floor material of the same REO is also assigned the same value of 3, we lose the specificity of the individual features. To address this issue, in this study, before fully encoding the data, we generate a data dictionary that captures the encoding process within. Specifically, we create a reference dictionary where each data category for a particular feature serves as the key, and the automatically generated encoding number serves as the value. We then save this encoded document for future use.

Furthermore, we also handle missing cells by assigning them specific values. Specifically, we replaced all missing values with "-111". The purpose of this approach is to ensure data consistency for features with missing values. For instance, for feature A, which has only 3 class labels, it is highly likely that the encoding for its missing value would be 4. On the other hand, feature B may have multiple class labels, such as 7, resulting in a missing value being encoded as 8. Such inconsistent encoding can complicate the identification of missing values in subsequent analyses. Our procedure involves replacing all missing values in categorical data with "MISSING_DATA" and then assigning this "MISSING_DATA" category to the value label "-111". Additionally, for numerical data, we directly replace missing values with "-111". After encoding all the data, we replace all cells with a value of "-111" with NAN, effectively restoring the positions of missing values. The separate encoding for different data categories ensures the integrity of the original data and facilitates traceability throughout the process.

For a detailed illustration of the steps, we refer to the diagram 4.2 below:

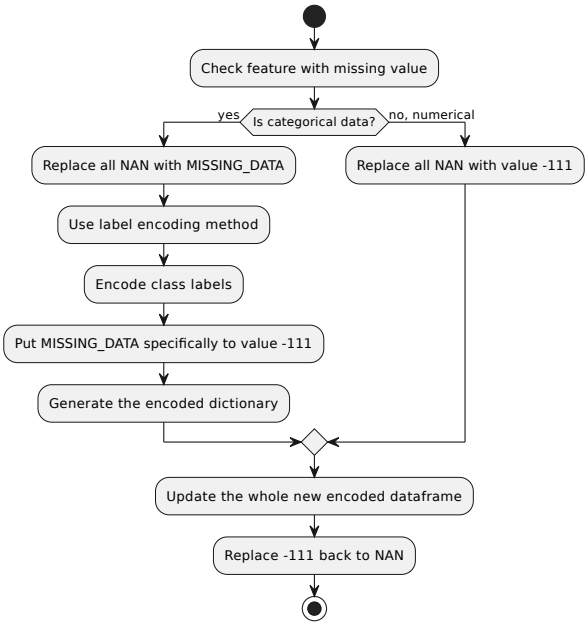

Figure 4.2: The Process of Encoding Data

4.2 Data Imputation - Bucket4Imp

4.2.1 Work Flowchart for the Imputation Process

Figure 4.3 presents an intuitive overview of the two main processes used for data imputation. We name this data imputation method **Bucket4Imp** (Bucket for Imputation).

In simple terms, based on the workflow diagram, the general process of our data imputation can be summarized as following two major steps. Firstly, we prune the dataset to obtain sufficient training and validation sets to learn data patterns. Then, we use a bucket ensemble model to select the best predicted output as the imputed values for missing data and get the imputed new dataframe.

More specifically, we examine all features containing missing values to determine if there are enough data rows to support training with an adequate amount of data for that specific feature. If we are unable to obtain sufficient training data due to missing values in other data, we discard columns with a high number of missing values until we have enough data records to support learning specific data patterns for the target feature. We believe that even though this process results in the loss of some features that can be used when we do the training on our target feature column, these features with excessive missing values do not contain enough information for the data model and hinder the overall learning of the model. Therefore, temporarily discarding them has minimal impact on the overall learning performance of the model. Subsequently, we evaluate the learned models on the validation set by calculating their performance scores using different models. We then select the model represented by the highest performance score and retrain it on the entire validation and training sets combined, using the model to compute the missing values for that specific feature.

In the following sections, we will use a simple case to further explain this process.

4.2.2 Dataset Pre-organization

In order to better explain our data imputation method, we use a simple dataframe as an example.

Table 4.1: Original Table with Missing Values

Col1	Col2	Col3	Col4	Col5	Col6	Col7	Col8	Col9	Col10	Col11	Col12	Col13	Col14	Col15
5.0	NaN	0.0	4.0	NaN	6.0	4.0	NaN	4.0	2.0	8.0	7.0	2.0	9.0	8.0
8.0	5.0	4.0	2.0	2.0	1.0	6.0	9.0	3.0	3.0	2.0	7.0	7.0	7.0	7.0
9.0	4.0	4.0	7.0	1.0	NaN	7.0	3.0	9.0	0.0	1.0	NaN	5.0	9.0	9.0
4.0	1.0	4.0	3.0	3.0	4.0	0.0	2.0	3.0	6.0	0.0	NaN	9.0	5.0	0.0
5.0	8.0	1.0	8.0	8.0	3.0	1.0	5.0	6.0	5.0	6.0	9.0	2.0	5.0	6.0
0.0	5.0	1.0	NaN	2.0	NaN	9.0	0.0	3.0	5.0	7.0	6.0	1.0	8.0	NaN
8.0	9.0	8.0	5.0	NaN	5.0	4.0	4.0	3.0	0.0	8.0	4.0	0.0	1.0	5.0
6.0	8.0	2.0	6.0	5.0	NaN	6.0	7.0	NaN	4.0	4.0	8.0	1.0	2.0	3.0
6.0	0.0	3.0	9.0	3.0	6.0	9.0	6.0	4.0	NaN	7.0	NaN	3.0	7.0	1.0
1.0	6.0	2.0	NaN	5.0	0.0	8.0	1.0	6.0	9.0	0.0	3.0	5.0	9.0	0.0

Firstly, Table 4.1 illustrates a simple randomly generated dataframe, where certain data points have been deliberately set as missing to simulate our real case dataframe. The first step within Bucket4Imp involves arranging the columns in ascending order based on the number of missing values. It is reasonable to start imputing from the column with the fewest missing values. Hence, we initially arrange the columns of the dataframe based on the quantity of missing data, as shown in Table 4.2 (the rightmost column - Col12 - has the most amount of missing data). Subsequently, we seek all available records to assess the feasibility of using the data for calculating and predicting these missing values.

However, in this case, we are unable to find an adequate number of records with full data for training the model. Clearly, only two out of the total ten rows do not provide sufficient information and data patterns to train the model, as observed in Table 4.3. Trying to solve this problem, consequently, we consider *"ignoring"* certain columns

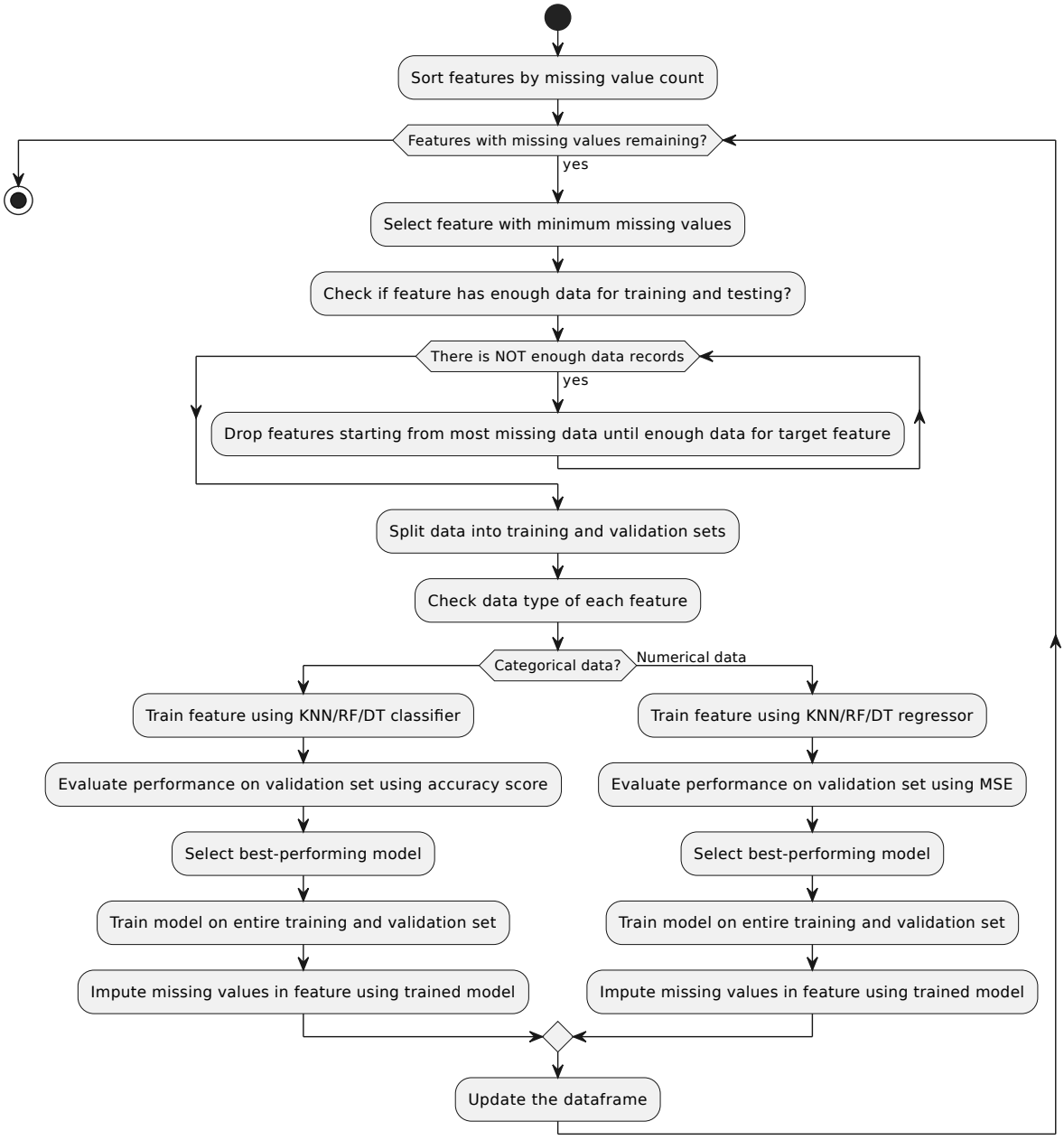

Figure 4.3: The Data Imputation Method - Bucket4Imp

Table 4.2: Feature Sorted by Amount of Missing Values

Col1	Col3	Col7	Col11	Col13	Col14	Col2	Col8	Col9	Col10	Col15	Col4	Col5	Col6	Col12
5.0	0.0	4.0	8.0	2.0	9.0	NaN	NaN	4.0	2.0	8.0	4.0	NaN	6.0	7.0
8.0	4.0	6.0	2.0	7.0	7.0	5.0	9.0	3.0	3.0	7.0	2.0	2.0	1.0	7.0
9.0	4.0	7.0	1.0	5.0	9.0	4.0	3.0	9.0	0.0	9.0	7.0	1.0	NaN	NaN
4.0	4.0	0.0	0.0	9.0	5.0	1.0	2.0	3.0	6.0	0.0	3.0	3.0	4.0	NaN
5.0	1.0	1.0	6.0	2.0	5.0	8.0	5.0	6.0	5.0	6.0	8.0	8.0	3.0	9.0
0.0	1.0	9.0	7.0	1.0	8.0	5.0	0.0	3.0	5.0	NaN	NaN	2.0	NaN	6.0
8.0	8.0	4.0	8.0	0.0	1.0	9.0	4.0	3.0	0.0	5.0	5.0	NaN	5.0	4.0
6.0	2.0	6.0	4.0	1.0	2.0	8.0	7.0	NaN	4.0	3.0	6.0	5.0	NaN	8.0
6.0	3.0	9.0	7.0	3.0	7.0	0.0	6.0	4.0	NaN	1.0	9.0	3.0	6.0	NaN
1.0	2.0	8.0	0.0	5.0	9.0	6.0	1.0	6.0	9.0	0.0	NaN	5.0	0.0	3.0

Table 4.3: Full Rows that Could Be Used for Training

Col1	Col3	Col7	Col11	Col13	Col14	Col2	Col8	Col9	Col10	Col15	Col4	Col5	Col6	Col12
8.0	4.0	6.0	2.0	7.0	7.0	5.0	9.0	3.0	3.0	7.0	2.0	2.0	1.0	7.0
5.0	1.0	1.0	6.0	2.0	5.0	8.0	5.0	6.0	5.0	6.0	8.0	8.0	3.0	9.0

that contain a significant number of missing values. Specifically, due to the presence of these columns, we cannot obtain enough information/patterns from the data records. In order to solve this problem, we compare the columns based on their minimum impact degree, intending to transform the problem into a set cover problem, where the problem was proven to be an NP-complete problem in 1972. The set cover problem is given as follows: Given a universe U consisting of n elements, and a collection S of m subsets of U such that the union of S is equal to U, the set cover problem aims to find the smallest sub-collection $C \subseteq S$ that covers all the elements in U [93].

Now we further explain and analyze this process. Firstly, regarding the data imputation on column **Col9**, as seen from Table 4.3, we do not have enough rows of records to aid in model training. When we set the threshold for the number of specific data rows to be 67% of the total data (2/3 of total data). Here, we selected a threshold of 67% because, as part of our data pre-processing procedure, we excluded feature columns with missing data exceeding 30%. Consequently, the highest missing data percentage among the remaining features is less than or equal to 30%. Therefore, we can only set the threshold to be less than 70% (i.e., the completeness of a feature), as exceeding this value would result in an inability to impute the most (i.e., the last) missing feature. After setting the threshold, we need to discard some rows that contain excessive (or too scattered) data in order to obtain more data pattern information (more data records/rows). In this process, we utilize a **greedy algorithm** to discard columns that contain missing values. The application of the greedy algorithm in the set cover problem is based on a rule that selects a set with the maximum number of uncovered elements at each stage. This method has a linear relationship with the sum of the sizes of the input sets in terms of time, and it uses a bucket queue to determine the priority of sets.

After running the algorithm, since we require at least 67% of the data records we iteratively check each column to determine if the minimum required number of data rows can be achieved without considering that particular column. Firstly, we drop column **Col6** according to the algorithm. This still leaves only 2 full rows. We then keep dropping columns until reaching our required threshold of full rows (7 here). This means additionally dropping columns **Col12**, **Col4**, **Col5**, **Col2** and **Col8**. The result is shown in Table 4.4.

The remaining dataframe is shown in Table 4.5. From the table, it can be observed that although there are some missing values in **Col10** and **Col15** (where **Col9** serves as our imputation target), these omissions do not significantly impact our model training and predictions. Moreover, they contribute to a richer set of available feature information.

Table 4.4: Sufficient Full Rows

Col1	Col3	Col7	Col11	Col13	Col14	Col9	Col10	Col15
5.0	0.0	4.0	8.0	2.0	9.0	4.0	2.0	8.0
8.0	4.0	6.0	2.0	7.0	7.0	3.0	3.0	7.0
9.0	4.0	7.0	1.0	5.0	9.0	9.0	0.0	9.0
4.0	4.0	0.0	0.0	9.0	5.0	3.0	6.0	0.0
5.0	1.0	1.0	6.0	2.0	5.0	6.0	5.0	6.0
8.0	8.0	4.0	8.0	0.0	1.0	3.0	0.0	5.0
1.0	2.0	8.0	0.0	5.0	9.0	6.0	9.0	0.0

Table 4.5: Remaining Dataframe

Col1	Col3	Col7	Col11	Col13	Col14	Col9	Col10	Col15
5.0	0.0	4.0	8.0	2.0	9.0	4.0	2.0	8.0
8.0	4.0	6.0	2.0	7.0	7.0	3.0	3.0	7.0
9.0	4.0	7.0	1.0	5.0	9.0	9.0	0.0	9.0
4.0	4.0	0.0	0.0	9.0	5.0	3.0	6.0	0.0
5.0	1.0	1.0	6.0	2.0	5.0	6.0	5.0	6.0
0.0	1.0	9.0	7.0	1.0	8.0	3.0	5.0	NaN
8.0	8.0	4.0	8.0	0.0	1.0	3.0	0.0	5.0
6.0	2.0	6.0	4.0	1.0	2.0	NaN	4.0	3.0
6.0	3.0	9.0	7.0	3.0	7.0	4.0	NaN	1.0
1.0	2.0	8.0	0.0	5.0	9.0	6.0	9.0	0.0

4.2.3 Machine Learning-based Data Imputation

Due to the similarities between the process of handling the user's entered data and the learning and computation process of machine learning models, both involve decision-making processes, and the results of each feature may be mutually influenced. When evaluators input evaluation results, they also need to engage in decision-making processes under different conditions to determine the evaluation outcomes. For example, if the location of a real estate object (REO) is more than 5010 km away from the city center, we classify the proximity to the city center as "far". Furthermore, under such circumstances, if the REO has fewer than 10 parking spaces in its vicinity, we consider its parking capacity to be "poor". This process bears a striking resemblance to the construction of a Decision Tree.

Hence, it is reasonable to simulate decision processes to predict missing values during the computation of real estate data. Through nearest neighbor imputation, we can interpolate missing values using the numerical results of other REOs with similar overall conditions. Similarly, by utilizing Random Forests or Decision Trees, we enable the learning model to acquire knowledge from previous evaluations conducted by evaluators. Consequently, when encountering missing values in an evaluation record, we simulate the evaluator's thought process to compute the missing values.

From this point of view, we then utilize (in total) six data imputation models for the calculation and prediction process. Different prediction problems should use different prediction models, so if the target column's data type is categorical, it is the classification problem. We will compare the **Decision Tree Classifier, Random Forest Classifier** and **K Nearest Neighbor Classifier**; And if our target column's data type is numerical (int or float), then our prediction problem is more likely a regression problem, so we will compare the **Decision Tree Regressor, Random Forest Regressor** and **K Nearest Neighbor Regressor**.

The process is as follows: Firstly, we separate the whole dataset into 80% for a training set and 20% for a testing set. Then, we partition 20% of the training set as the validation set. Next, we proceed to train the three models

using the training set to predict the target column and taking into account the data type. We then assess their performance on the validation set to identify the optimal model. For different data types, we employ different performance metrics to compare and choose the best model on the validation set. For categorical data, we compute balanced accuracy and F1 scores (where a higher score indicates better performance), while for numerical data, we calculate the Mean Squared Error (where a lower score indicates better performance). Subsequently, we combine the training set and validation set and retrain the best performing model. We then utilize this new-trained model to compute the missing values in the target column, replacing the NaN values with the computed results, thereby completing the data imputation for that column. Taking the example of column **Col9**, **Random Forest Regressor** exhibits superior performance among the three regression models. Hence, we employ this model for data imputation. Furthermore, We use Figure 4.4 to give a more direct and clear work flowchart for the use of the Bucket of models integration method to select the most suitable model for each feature.

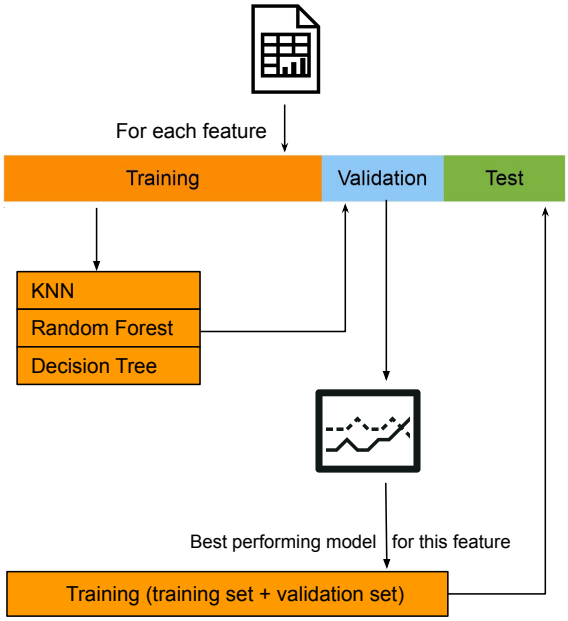

Figure 4.4: Training, Validating, and Testing Process for Each Features

The data obtained for this column (row) are shown in Table 4.6, where this data row is the imputation target for **Col9**. Upon completion of the computation for **Col9**, we obtain the initial round of data imputation results in the dataframe shown in Table 4.7.

Table 4.6: Predicting Missing Values for Column: Col9

Col1	Col3	Col7	Col11	Col13	Col14	Col10	Col15
6.0	2.0	6.0	4.0	1.0	2.0	4.0	3.0

Afterward, we proceed with the same procedure for the next column as the same following process: utilizing the updated dataframe, which incorporates the newly imputed missing data, we perform a search for the minimum usable data records. During this process, we also discard columns that hinder the attainment of sufficient data. Subsequently, using the obtained data records, we train a model specific to the current target column. Following model training using the validation set, the best model is determined, and subsequently, the training set and validation set are combined to train the selected optimal model. Subsequently, this model is employed to calculate, predict, and replace all missing data for the respective column. The pseudocode for this process can be outlined

Table 4.7: Imputed Results for Col9

Col1	Col3	Col7	Col11	Col13	Col14	Col2	Col8	Col9	Col10	Col15	Col4	Col5	Col6	Col12
5.0	0.0	4.0	8.0	2.0	9.0	NaN	NaN	4.00	2.0	8.0	4.0	NaN	6.0	7.0
8.0	4.0	6.0	2.0	7.0	7.0	5.0	9.0	3.00	3.0	7.0	2.0	2.0	1.0	7.0
9.0	4.0	7.0	1.0	5.0	9.0	4.0	3.0	9.00	0.0	9.0	7.0	1.0	NaN	NaN
4.0	4.0	0.0	0.0	9.0	5.0	1.0	2.0	3.00	6.0	0.0	3.0	3.0	4.0	NaN
5.0	1.0	1.0	6.0	2.0	5.0	8.0	5.0	6.00	5.0	6.0	8.0	8.0	3.0	9.0
0.0	1.0	9.0	7.0	1.0	8.0	5.0	0.0	3.00	5.0	NaN	NaN	2.0	NaN	6.0
8.0	8.0	4.0	8.0	0.0	1.0	9.0	4.0	3.00	0.0	5.0	5.0	NaN	5.0	4.0
6.0	2.0	6.0	4.0	1.0	2.0	8.0	7.0	4.93	4.0	3.0	6.0	5.0	NaN	8.0
6.0	3.0	9.0	7.0	3.0	7.0	0.0	6.0	4.00	NaN	1.0	9.0	3.0	6.0	NaN
1.0	2.0	8.0	0.0	5.0	9.0	6.0	1.0	6.00	9.0	0.0	NaN	5.0	0.0	3.0

as algorithm 1:

Algorithm 1 Bucket4Imp: Imputation of Missing Values

procedure IMPUTEMISSINGVALUES(DataFrame D with missing values)
  $S \leftarrow$ Sort columns in D with missing values in ascending order of missing amount.
  while S is not empty **do**
    $c \leftarrow$ first column from S
    $c_{\text{full}} \leftarrow$ full data records (rows) from c
    while size(c_{full}) $\leq$ threshold and S is not empty **do**
      Drop the last column from S
      $c \leftarrow$ first column from S
      $c_{\text{full}} \leftarrow$ full data records (rows) from c
    Train_models $\leftarrow$ [KNN, Random Forest, Decision Tree]
    if c is numerical **then**
      Best_Model $\leftarrow \arg\min_{\text{model} \in \text{Train\_models}}$ MSE(Validation_set[model])
    else
      Best_Model $\leftarrow \arg\max_{\text{model} \in \text{Train\_models}}$ Balanced accuracy (F1)(Validation_set[model])
    Best_Model $\leftarrow$ Train_val_set
    Imputation $\leftarrow$ Test_set(Best_Model(S) $\rightarrow c$)
    Update D with newly imputed missing values on c
    Delete c from S

After a few rounds, we obtain the final imputed framework, as shown in Table 4.8 with all NaN values replaced.

4.2.4 Unit Test for Data Imputation

Despite using the best-performing models to calculate missing values for each feature, the effectiveness of this method for filling randomly missing data in the entire dataframe (accuracy of performance) remains unknown. This approach presents two potential issues: firstly, the pursuit of a certain amount of training data may lead to the loss of too much feature data; secondly, as we actually train the target column twice, first by searching for the best-performing model among the three models, and secondly by using all the full row records to retrain the target column and calculating the missing data in the test set, the introduction of additional rows may result in a change in the optimal model for the target column - for example, as the number of data records increases, Random Forest may be more effective than Decision Trees in learning pattern information, thus rendering the originally optimal model not truly optimal.

Table 4.8: Final Imputed Dataset

Col1	Col3	Col7	Col11	Col13	Col14	Col2	Col8	Col9	Col10	Col15	Col4	Col5	Col6	Col12
5.0	0.0	4.0	8.0	2.0	9.0	7.61	7.0	4.0	2.0	8.0	4.0	6.5	6.0	7.0
8.0	4.0	6.0	2.0	7.0	7.0	5.0	9.0	3.0	3.0	7.0	2.0	2.0	1.0	7.0
9.0	4.0	7.0	1.0	5.0	9.0	4.0	3.0	9.0	0.0	9.0	7.0	1.0	6.0	5.69
4.0	4.0	0.0	0.0	9.0	5.0	1.0	2.0	3.0	6.0	0.0	3.0	3.0	4.0	5.16
5.0	1.0	1.0	6.0	2.0	5.0	8.0	5.0	6.0	5.0	6.0	8.0	8.0	3.0	9.0
0.0	1.0	9.0	7.0	1.0	8.0	5.0	0.0	3.0	5.0	0.5	5.65	2.0	1.0	6.0
8.0	8.0	4.0	8.0	0.0	1.0	9.0	4.0	3.0	0.0	5.0	5.0	6.5	5.0	4.0
6.0	2.0	6.0	4.0	1.0	2.0	8.0	7.0	4.93	4.0	3.0	6.0	5.0	3.0	8.0
6.0	3.0	9.0	7.0	3.0	7.0	0.0	6.0	4.0	0.0	1.0	9.0	3.0	6.0	6.9
1.0	2.0	8.0	0.0	5.0	9.0	6.0	1.0	6.0	9.0	0.0	6.62	5.0	0.0	3.0

However, these two issues do not significantly impact our data imputation process for the following reasons: first, in the previous section, we explained some basic data preprocessing steps, including the exclusion of feature columns with more than 30% missing data and the exclusion of records (rows) with more than 30% missing data in a row. Such data cleaning and preprocessing allow us to retain data that can provide sufficient information and preserve more useful features during the data imputation process. Additionally, we believe that a larger number of data records will have a greater impact on the training of the target feature since a smaller number of records may mislead the model's learning. Second, our validation set accounts for only 20% of the training set - a moderate proportion. While the choice of the best model may vary due to differences in the number of training data rows, the problems associated with retraining three models or not calculating performance scores on the validation set are more significant. The former may result in lengthy computation time or increased computational space, while the latter is necessary to assess whether each model can achieve satisfactory scores.

In order to obtain a comprehensive assessment of the overall performance of our Bucket4Imp methodology, we employ a dataset consisting of complete information. This dataset, as documented in [101], exhibits similarities with the dataframe sourced from KATE Innovations, though without missing data (NAN). We will introduce this dataset in Chapter 5 as it is also used in our experiment. In this chapter we will only use this dataset as a unit test to our Bucket4Imp method. For this, we conduct a comparative analysis to ascertain the alignment between the imputed outcomes and the original data, focusing specifically on the positions of the dropped data (randomly generated missing data). The resultant concurrence rate is calculated to be **0.98**, representing an accurate imputation ratio.

4.3 Feature Selection

To perform feature selection, we employ the fast correlation-based filter algorithm proposed by Yu and Liu [111] [112]. This algorithm aims to identify a subset of features from a given dataset that provides informative insights for predicting a target variable. The algorithm operates by measuring the symmetrical uncertainty (SU) between each feature and the target variable, considering their mutual information and individual entropies.

The algorithm consists of the following functions:

Entropy Calculation

The "entropy" function computes the empirical entropy $H(X)$ of an input vector using a specified base. It calculates the probability distribution of values within the vector and derives the entropy based on this distribution.

Conditional Entropy Computation

The "conditional_entropy" function determines the conditional entropy $H(X|Y)$ by considering two input vectors, denoted as x and y. It identifies the unique values and their frequencies within the vector y. For each unique value, the function calculates the entropy of the corresponding subset of x where y takes on that particular value. Finally, it obtains a weighted average of these entropies based on the probabilities associated with the unique values in y.

Mutual Information Calculation

The "mutual_information" function quantifies the information gain or mutual information between two random variables, x and y. It subtracts the conditional entropy of x given y from the entropy of x, thereby measuring the shared information between the variables.

Symmetrical Uncertainty Calculation

The "symmetrical_uncertainty" function computes the symmetrical uncertainty (SU) between two random variables, x and y. It utilizes the mutual information between x and y, as well as the individual entropies of x and y. SU is obtained by calculating the ratio of twice the mutual information to the sum of the entropies.

Core Feature Selection

The "feature_selection" function performs the core process of feature selection. It takes three inputs: a dataset denoted as X, a target variable represented by y, and a predetermined threshold related to symmetrical uncertainty. The function initializes an empty list, S', to store the indices of the selected features. It iterates over each feature within X, computing the symmetrical uncertainty between the respective feature and y. If the symmetrical uncertainty exceeds or equals the defined threshold, the feature index is appended to S'.

The function then rearranges the elements of S' in descending order based on their associated symmetrical uncertainty values. It further iterates over the elements of S', removing any feature that fails to satisfy specific conditions based on the symmetrical uncertainty in relation to other selected features. Consequently, S' represents the indices of the selected features.

Finally, the function generates a new dataframe, denoted as S_{best}, by selecting the columns of X corresponding to the indices present in S'. This dataframe contains the chosen features and serves as the output of the feature selection algorithm.

In general, the algorithm calculates the symmetrical uncertainty between each feature and the target variable, and selects a subset of features based on a predetermined threshold. The selected features exhibit a significant degree of mutual information with the target variable while minimizing redundancy among themselves.

4.4 Data Prediction

Graczyk et al. [34] compared bagging, boosting, and stacking ensembles models using real estate evaluation data and found that there is no single algorithm that produces the best ensembles. Hence, it is worth seeking an optimal hybrid multi-model solution.

Therefore, we attempt to construct a multi-model solution. Similar to the methods used in data imputation process, here we have sufficient data - because of the data imputation process. Our goal is to find the best model applicable to each feature. We still employ a bucket-like ensemble approach: training multiple models on the same feature set and target feature, and then using the best-performing model to retrain the dataset and make predictions on the test set. We made some improvements to the bucket ensemble method in both data imputation and data prediction. After selecting the best-performing algorithm, we merge the validation set and training set to retrain the model on multiple data records, instead of use the best model immediately on the test set.

The workflow of our process is illustrated in Figure 4.5.

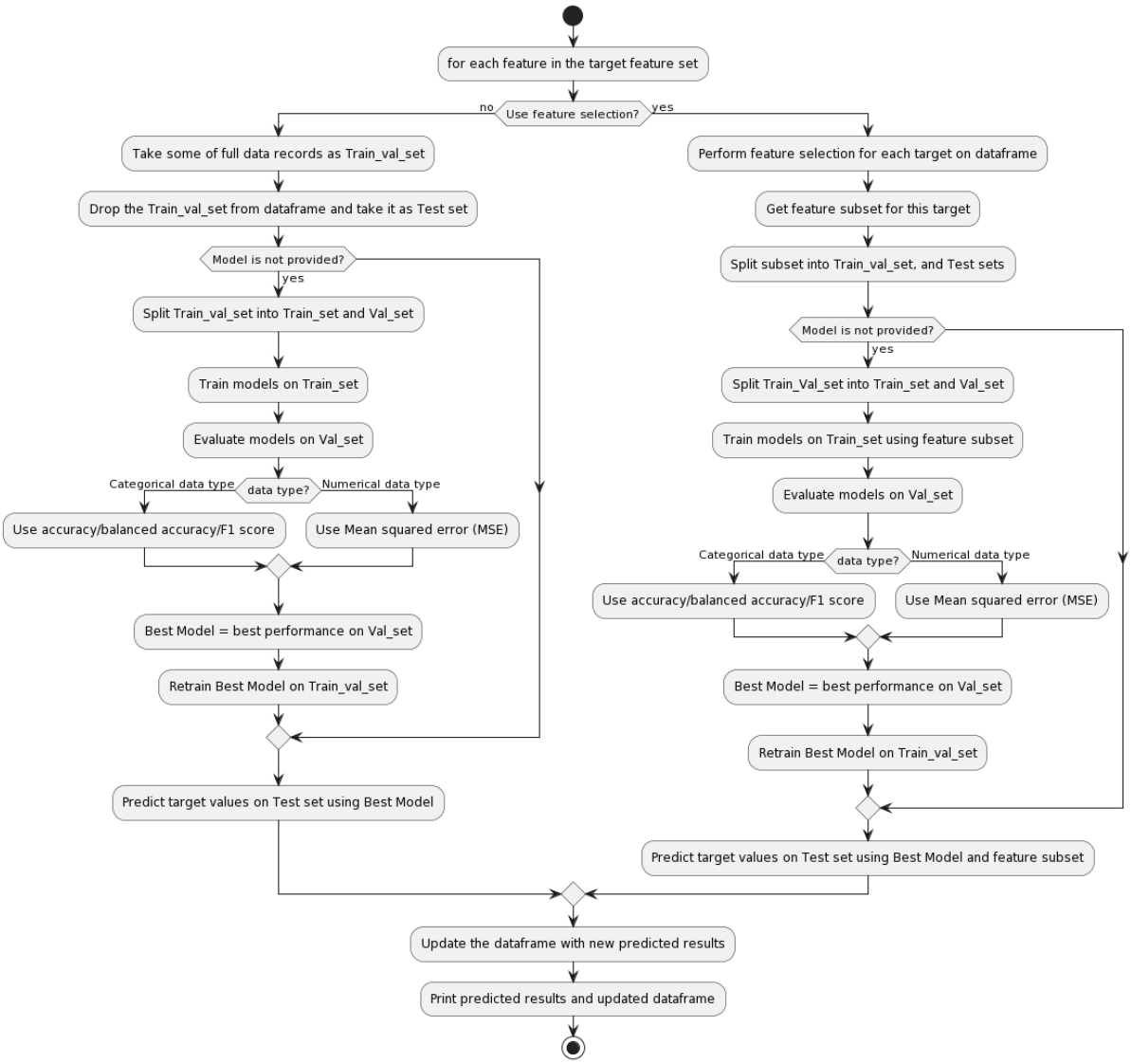

Figure 4.5: The Workflow for Data Prediction Process

The approach employed in this conclusive prediction process entails identifying the optimal model that is well-suited for each feature. Additionally, the option to implement feature selection with the aim of reducing feature dimensionality is also considered.

The prediction process first checks if we want to use feature selection to narrow down the high dimensional feature set. We set the feature selection to switch here because we want to apply our model to a more general usage scenario, and some data scenarios do not contain high-dimensional feature sets, thus using feature selection on a smaller set of features is relatively unnecessary.

When feature selection is disabled (we won't use feature selection to generate feature subsets, i.e, do the data prediction directly), we generally split the data into a training set and a testing set, with a 70% and 30% split respectively. We then find out the optimal model that well-suit for this feature. We use the same bucket ensemble

method to find the best model when using trained model on validation set. Evaluation metrics, such as balanced accuracy or other appropriate metrics depending on the problem domain, are computed to compare the models. The best model is selected based on these evaluation metrics.

Once the best model is determined, it undergoes retraining employing the complete training and validation dataset. This retraining procedure is designed to leverage the entirety of the available data, thereby enhancing the model's performance. After this retraining process, the resulting model is applied to the test set for final predictions.

In the case where feature selection is enabled (i.e, we use feature selection to do the pre-processing before the data prediction, and then use the selected feature in the prediction model), the function performs feature selection to identify the best features for each target column using a specified threshold value. The selected features are then used to train the model. The rest of the prediction process is similar to the process described earlier.

In conclusion, the data prediction function follows a methodology that combines feature selection and model training to predict the target feature in a dataframe.

Some of the main processes of data prediction described above are shown in Algorithm 2:

Algorithm 2 Data Prediction and Feature Selection

Require: D_{di} (dataframe after data imputation for non-target set), T (target set), FCBF feature selection model

Ensure: *Prediction* (Prediction results for target set T)

for T_i in T **do**
  if use feature selection **then**
    $D'_{di} \leftarrow FCBF(D_{di})$ ▷ Generate feature subset using FCBF
  else
    $D'_{di} \leftarrow D_{di}$
  $Train\_val\_set$, $Test\_set \leftarrow$ Split(D'_{di})
  $Train\_set$, $Validation\_set \leftarrow$ split($Train\_val\_set$)
  $[KNN, \text{Random Forest}, \text{Decision Tree}] \rightarrow Train\_set$
  $Best\_Model \leftarrow \min(\text{MSE}(Validation\_set[KNN, \text{Random Forest}, \text{Decision Tree}]))$ ▷ If T_i is numerical
  $Best\_Model \leftarrow \max(\text{accuracy}(Validation\_set[KNN, \text{Random Forest}, \text{Decision Tree}]))$ ▷ If T_i is categorical
  $Best\_Model \rightarrow Train\_val\_set$
  $Prediction \leftarrow \text{Test\_set}(Best\_Model(D'_{di}) \rightarrow T_i)$
  $T_i \leftarrow$ Prediction

Chapter 5

Experiment and Results

In this chapter, we will utilize the imputation, feature selection and prediction methods mentioned in the methodology framework to give prediction results for the valuable feature targets provided by KATE Innovations. Additionally, we will incorporate two additional datasets with similar sizes and one dataset with larger data records to further evaluate the performance of our framework. This approach will enable us to thoroughly assess the effectiveness of our methodology.

In the content of this chapter, the following topic will be discussed: 1) An introduction and analysis of the experimental dataset; 2) Experimental details and some experimental settings under different model configurations; 3) Experiments conducted based on the framework mentioned in the methodology and the corresponding results; 4) The experimental examination of the relationship between the number of feature subset features used in the data prediction process and the resulting performance scores; 5) Methodological improvements for performance enhancement and improved results; 6) All experiments and their results performed on large datasets; 7) The performance comparison between statistically based data imputation and the Bucket4Imp method.

5.1 Datasets and Pre-processing

In our main study, we utilized three datasets for conducting the experiment. Our objective was to address the problem raised by KATE Innovations, which involves predicting interesting and valuable feature values. We used KATE Innovations' dataset as our primary experimental dataset and obtained two other datasets of similar size from Kaggle [46] [48] [4] for validating the effectiveness of our proposed method compared to not using any pre-processing methods.

In this section, we provide a brief introduction to the three datasets, which include the dataset provided by KATE Innovations, in what follows is called the **KATE dataset**. We then perform some basic observations on each dataset, as well as some dataset pre-processing due to its characterization.

5.1.1 KATE Dataset

The dataset, obtained directly from KATE Innovations, a Dutch real estate valuation software company, consists of 12 data tables that are downloaded from their database. Each table is connected through foreign keys, creating complex relationships between them. We have performed necessary table associations and preprocessing, such as linking the values of two tables using unique foreign key values to generate a larger consolidated dataset, only using finished valuation records, as well as removing duplicate data entries, etc. Reorganizing these disparate data tables allows us to gain a better understanding of the overall data situation. Additionally, constructing a single table that includes all the relevant data will better serve subsequent experiments and operations.

As a result of the interaction between the multiple internal correlation constraints for all these 12 data tables, we have reduced the table to a more informative and formatted dataset, and a reduction of 89% from the original dataset.

Before conducting the experiment, we conducted a basic assessment of the data in the KATE dataset. The benefits of doing this are twofold. Firstly, we can gain a better understanding of the data, allowing us to determine the necessary steps for processing and analyzing it, thereby minimizing errors and biases in subsequent analyses. This, in turn, enables us to gain valuable insights and make better decisions. Secondly, assessing the data quality early on provides significant advantages in terms of cost and time efficiency. By proactively addressing any quality issues at the outset, we can avoid potential downstream problems that may require rework or corrections. As a result, the overall efficiency of the project is improved while costs are reduced.

The KATE dataset is composed of **1134** data record rows and **692** feature columns, with 178 float64 data type features and 514 object data type features. In order to ensure consistency, a thorough examination of the basic characteristics and distribution of the data is undertaken. Specifically, three features pertaining to monetary valuation, namely the rounded valuation results and the unrounded valuation fees, are selected from the dataframe. The analysis reveals that the rounded monetary values exhibit a reduced number of unique values, which is to be expected since rounding can result in numerical convergence. Conversely, when examining the unrounded columns, it is observed that although some values may be identical, the majority of them are unique. Both the graphical representation (Figure 5.1) and the tabular data (Table 5.1) confirm the overall consistency of the dataset, as no duplicate records are found, and most of the data align with the expected patterns. This outcome is highly advantageous because it ensures the integrity and reliability of the data, facilitating accurate analyses and avoiding potential distortions that duplicates could introduce.

Table 5.1: KATE dataset: Basic Data Information (using three features as an example)

	count	unique	top	freq
valuation_value_mv_amount	1034	411	270000	19
valuation_value_mv_amount_unrounded	870	844	523076	3
valuation_value_mv_before_fees	703	684	566230	3

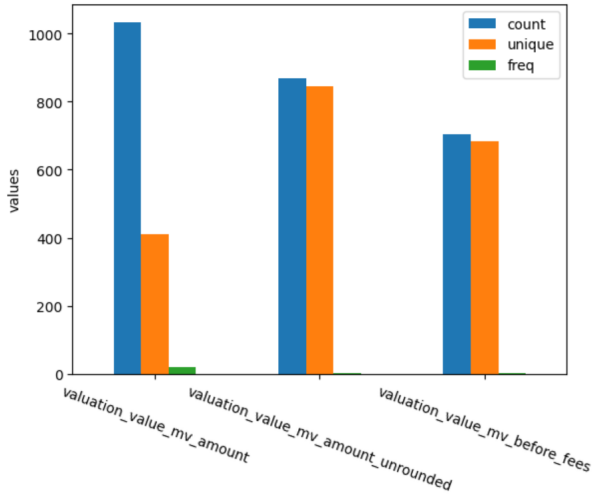

Figure 5.1: KATE Innovations: The Bar Chart of Three Features

We then analyze the most important characteristic of data quality - data completeness. We use the following Figure 5.2 and Figure 5.3 to illustrate the basic situation.

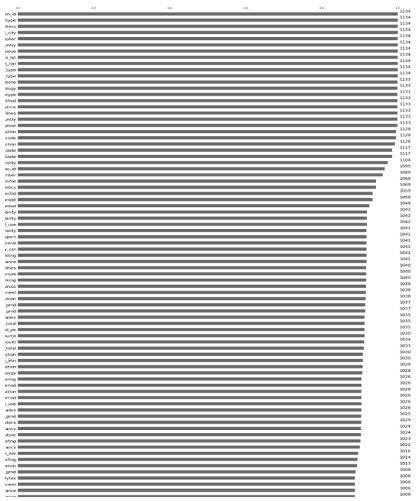

Figure 5.2: Each Feature's Data Completeness Starting from the Most Complete One

Figure 5.3: Each Feature's Data Completeness Ending with the Most Incomplete One

The analysis of data completeness is visually depicted in Figure 5.2 and Figure 5.3. Each bar in the figures represents the completeness of a specific feature and we sort them in descending order. In Figure 5.2, the bars correspond to the most complete features, where a significant amount of values are present. Here, the actual feature names are omitted in the figure due to company data confidentiality. On the other hand, Figure 5.3 displays the most incomplete features. Between these two figures, there are additional bar charts representing the completeness situation of various features. These features fall between the two depicted in Figure 5.2 and Figure 5.3, as they are sorted in descending order of data completeness. However, due to the large number of features and the space limitations of the figure, we will only display the top and bottom portions, omitting the intermediate feature's data completeness situation. Upon examining the descending order of existing data for these features, it is evident that certain features lack any data, as the number of available values is zero.

Table 5.2: Data Completeness Situation

Completeness Threshold	**Completeness Percentage**
80%	19.075145%
70%	20.231214%
60%	21.820809%
50%	26.300578%

Given the significant amount of missing data, we calculate the percentage of feature counts surpassing a certain threshold of completeness across the entire dataset. Table 5.2 shows that roughly 19% of features in the complete dataset exhibit a data completeness level of 80% or more. Furthermore, only about a quarter of the features have half or more of their data available within the dataset as a whole.

Overall, the present situation is considerably poorer compared to normal datasets, as a significant portion of the features in this dataset cannot be utilized due to insufficient information provided.

Several reasons account for the occurrence of missing data, as discussed with experts within the company. For instance, certain features may not be applicable to specific real estate situations, resulting in valuers being unable to complete these fields and thus leading to missing data. This type of missing data can be classified as Missing At Random (MAR), wherein the missingness of the data is systematically linked to the observed data rather than the unobserved data. In other words, the absence of a particular variable is related to other variables. For example,

when an appraiser selects the property type as a small apartment, the corresponding option for whether there is a backyard may be left blank, and this empty cell is reflected in the dataset as missing data.

Another reason for the occurrence of missing data could be that during the valuation process, certain fields are deemed less important and are not mandatory for valuers to fill in. Consequently, data is missing for these valuation records, resulting in a lack of data records for the entire feature. From this perspective, this type of missing data could be considered Missing Not At Random (MNAR). In this case, the fact that the data is missing is systematically related to the unobserved data, indicating that the missingness is connected to events or factors that are not measured by the researcher.

Although we have already organized the entire KATE dataset, further preprocessing is required, as indicated in Table 5.2, due to the presence of significant missing data. Schafer [89] suggests that a missing rate of 5% or less is inconsequential, while Bennett [9] argues that statistical analysis may be biased when more than 10% of the data is missing. In our case, we aim to retain as many features as possible and employ data imputation techniques to recover missing values. Therefore, we have set a data completeness threshold of 70%. This means that if a feature's data have less than 70% of values, the column cannot provide sufficient information for subsequent training and prediction processes, potentially leading to biased results. Consequently, we apply the same threshold to data records. If a record has less than 70% of its data available, it will not be considered for use.

After dropping features that have more than 30% of their data missing, as well as data records with more than 30% missing data, we assessed the remaining dataset. The size of the KATE dataset after pre-organization has **1027** data record rows and **104** feature columns. The analysis revealed that 78 features still had missing values, accounting for 75.00% of the total features.

We then picked some interesting features from whole feature set as our target set, which also checked by the company's expert. These targets are: **car accessibility, general accessibility, public transport accessibility, location position, marketability conclusion, the lettability of marketability, the lettability period of marketability, the saleability of marketability, the saleability period of marketability, construction condition, facades condition, floors grnd condition, foundation condition, loadbearing condition, roofing condition, state maintenance condition, window frames grnd condition, parking facilities, parking location, location position current use, object assessment of marketability, object functionality**, and the **rental situation**

The missing data situation for these target sets is showed in Figure 5.4, where the y-axis represents the missing amount of data in each column divided by the total length of the dataframe after pre-processing.

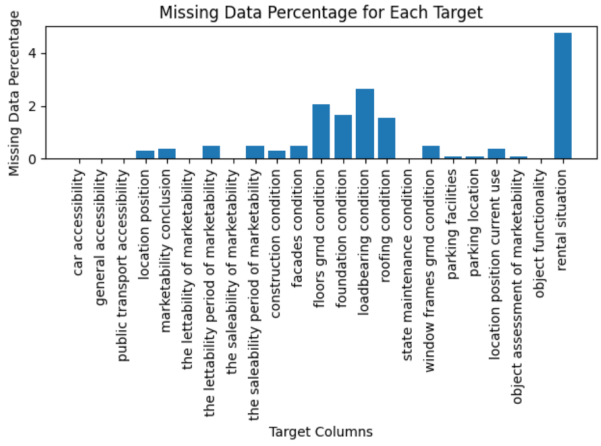

Figure 5.4: KATE dataset: Target Feature's Data Missing Percentage

Besides the missing data situation, we also check the data distribution as shown in Figure 5.5. From the figure,

we can see that for each target feature, their data distributions are not evenly distributed, with most concentrated on a single value and only a very small number of values for the other feature classes. Upon closer examination of this data, we can observe that most of the data is concentrated in evaluation cases that are slightly above average, specifically the "good" category among all corresponding label classes. Subjectively speaking, this doesn't appear to be a favorable phenomenon, as it suggests that the evaluations are mostly neutral, neither good nor bad, or just satisfactory. Other than that, other feature's data distribution is mainly represented as "na" for "not applicable" for the situation that the evaluation criteria are not applicable to this Real Estate Object (REO); or "unknown" which refers to the situation that the situation of this REO cannot get information, for example, the energy labels.

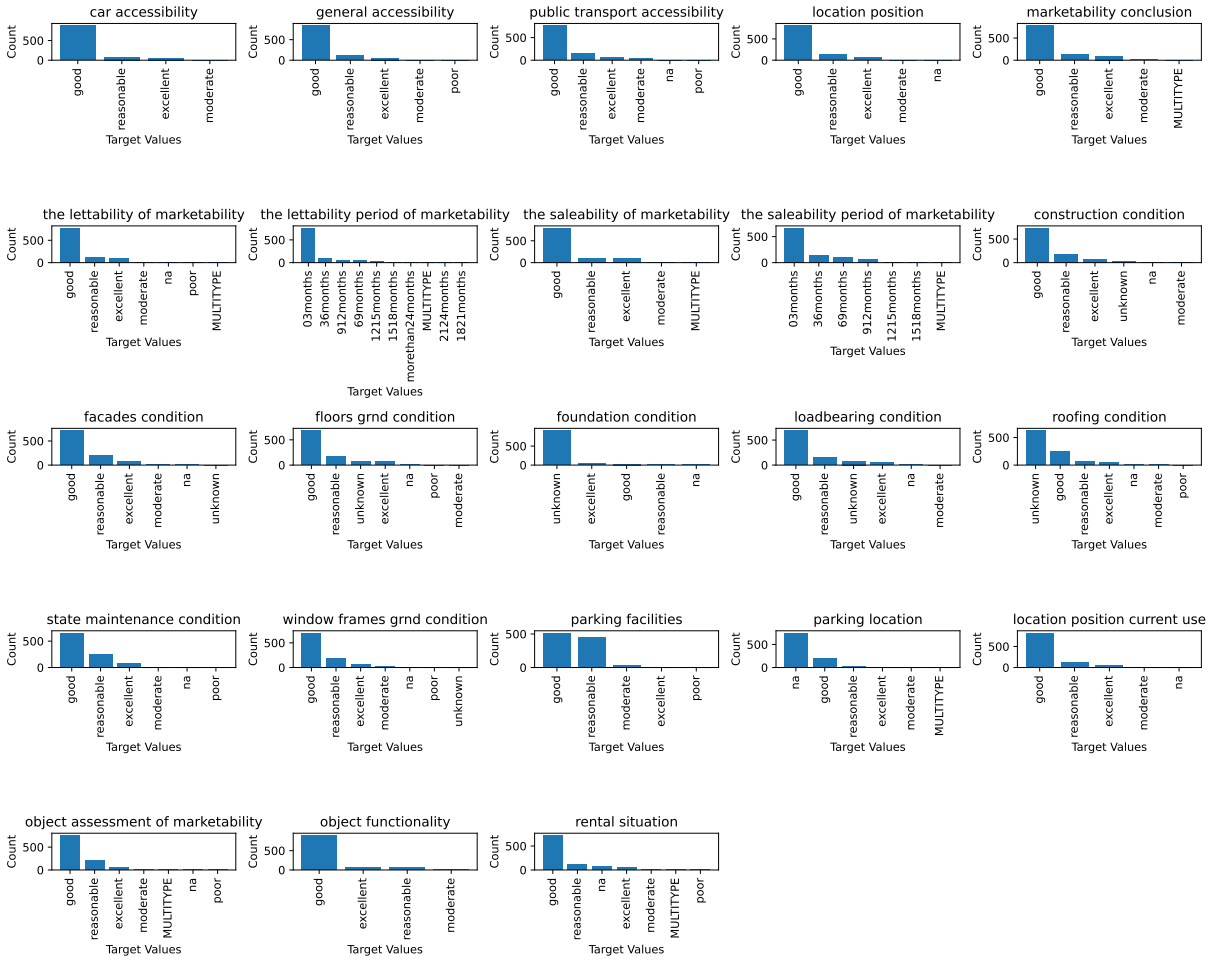

Figure 5.5: KATE dataset: Target Feature Distribution

5.1.2 House Dataset and Insurance Dataset

Our second dataset is from Kaggle, specifically from one of the house price prediction challenge datasets [4], which we denote as the **house dataset** for the later experiments. In this dataset, the total size of the dataframe is similar to the dataset from KATE Innovations (after pre-organization), where they have **1460** data records and **81** feature columns. All the feature sets are derived from various domains of real estate data evaluation, such as the general zoning classification, lot size in square feet, or style of dwelling, etc. We randomly selected several features from this dataset, they are **"LotShape", "LotConfig", "Neighborhood", "BldgType", "HouseStyle",**

"MasVnrType", **"BsmtExposure"**, **"GarageFinish"**, and **"SaleType"**. In Figure 5.6 we show the data distribution of our picked target features, in Figure 5.7 we illustrate the and in Figure 5.8 we present each feature's data missing situation.

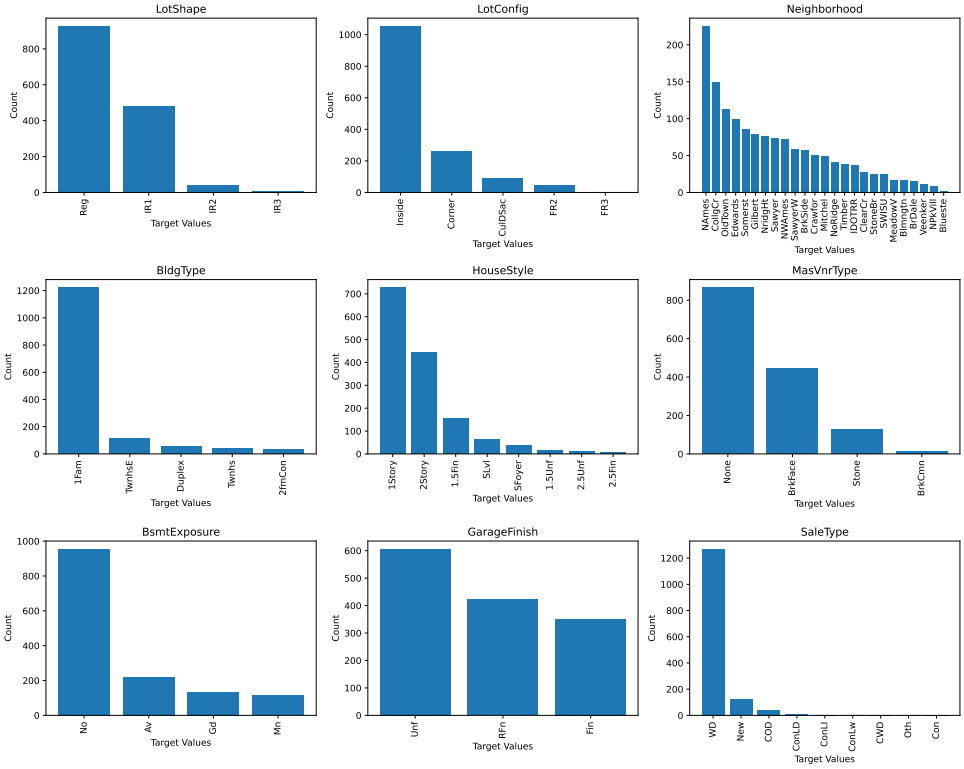

Figure 5.6: House dataset: Target Feature's Data Distribution

From the Figure 5.6, Figure 5.7, and Figure 5.8, we can see that for this dataset, the missing data situation is not as bad as the KATE dataset. Also, for our target variable, in the dataset, only a few instances contain missing data, and the missing values are very few. Furthermore, the data distribution of the target variable appears relatively evenly distributed. However, among the features, we observe some highly imbalanced distributions. For example, in the "BldgType" feature, the majority of the data is concentrated on "1Fam" and in the "SaleType" feature, most of the data is concentrated on "WD".

The third dataset is from Kaggle's Porto Seguro's Safe Driver prediction task [47]. This dataset was originally used to predict whether a driver will file an insurance claim next year, thus, we refer to this dataset as **insurance dataset** in the following work. The dataset consists of more than **10,000** data records and **60** feature columns, from which we will randomly select around 1400 data records of the over 10,000 data records to make the dataset as similar as the other two. Unlike the other two datasets, this dataset appears to have no missing values on the surface. However, missing values in this dataset are represented as "-1". This dataset's overall data missing situation is presented in Figure 5.9. In this experiment, to better evaluate the performance of the model, we replace "-1" with NAN. Additionally, we randomly selected 9 features from the 60 feature columns as our target prediction features. They are: **"ps_ind_16_bin"**, **"ps_calc_16_bin"**, **"ps_calc_10"**, **"ps_ind_03"**, **"ps_calc_06"**, **"ps_car_01_cat"**, **"ps_ind_09_bin"**, **"ps_car_07_cat"**, and **"ps_ind_02_cat"**. The distribution of their data is shown in Figure 5.10.

From Figure 5.10, we can see that the data distribution of most target features is more even compared to the other two datasets, except for the feature "ps_car_07_cat" where a large amount of data is concentrated on the value 1.0.

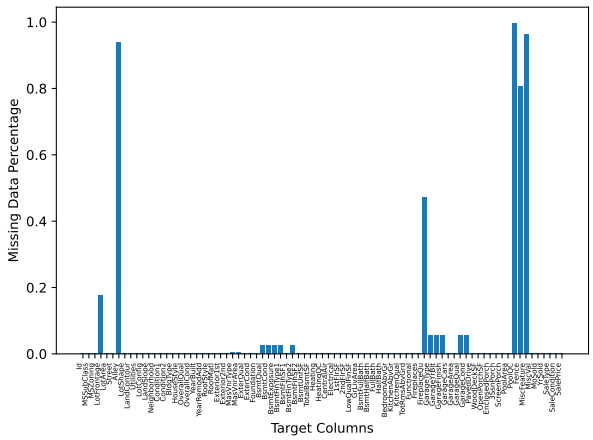

Figure 5.7: House dataset: Overall Missing Situation

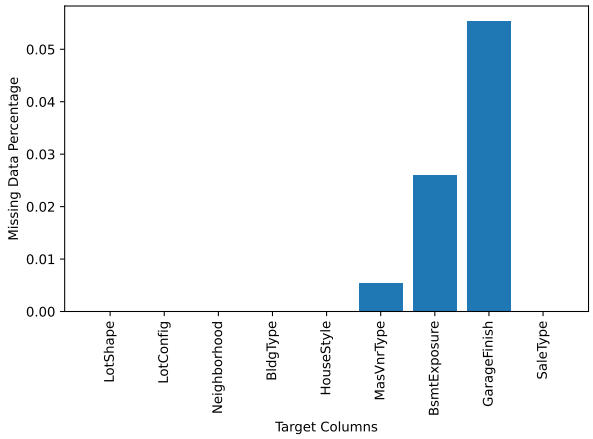

Figure 5.8: House dataset: Target Feature's Data Missing Percentage

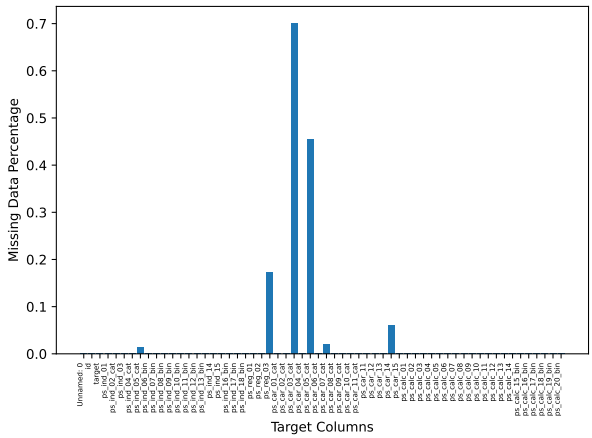

Figure 5.9: Insurance dataset: Overall Missing Situation

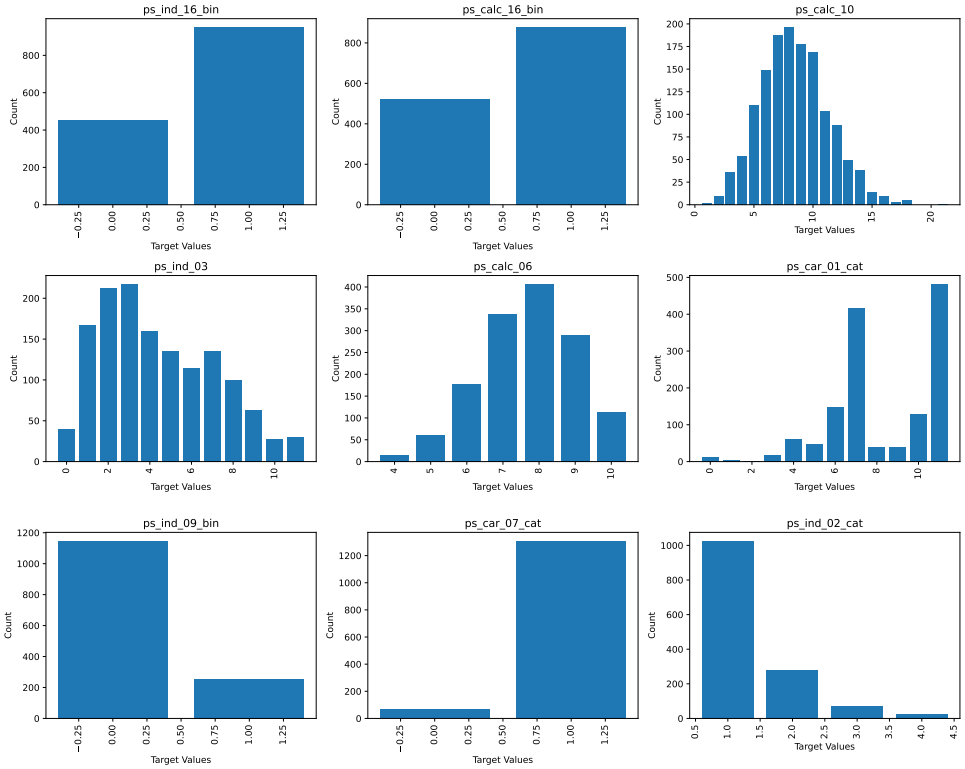

Figure 5.10: Insurance dataset: Target Feature's Data Distribution

On the other hand, the data distribution of the feature "pc_ind_03" is relatively even. Additionally, the binary feature data "ps_calc_16_bin" has a relatively ideal distribution. Furthermore, looking at the occurrences of "-1" in the entire dataset, which indicates missing data, as shown in Figure 5.9, we can see that the dataset only contains a small number of features with missing values (from the figure, it appears to be only 6), and among them, three features have missing values below 15%. Overall, the data quality is good and can be used for further research.

For the house and insurance dataset, we do the same data pre-processing - drop the feature columns and data record rows that have less than 70% of data values. However, due to the relatively high data quality of the house dataset, it is not very meaningful to directly perform data imputation and then do data prediction on it. We believe that the data missingness situation of this dataset would not significantly affect data prediction. Thus, in order to better evaluate the performance of our model and the effect of internal modules on data prediction, we constructed a process to manually generate missing values in the data. This process focuses on generating missing data in a dataset while keeping a specified percentage of rows without missing values.

The method randomly selects columns that do not contain target variables and calculates the percentage of missing values in those columns. It identifies the columns with missing values and determines the desired number of columns to generate missing data. Additional columns are randomly chosen to fulfill this requirement. To better simulate the missing data scenario of the previous KATE dataset, we imposed a restriction on the percentage of columns containing missing values out of the total number of columns. We limited this range to 50% - 70%. The process continues by iteratively checking the number of rows with missing values and randomly selecting rows to meet the desired percentage. Missing data is generated by setting the selected cells to NaN. The method also ensures that neither a row nor a column exceeds a 30% missing value threshold.

After manually generating missing data, we get Figure 5.11 for the new house data with more missing values and Figure 5.12 for the new insurance data with more missing values. These two figures also present that the missing values are missing randomly throughout the dataset, in line with our ideal expectations.

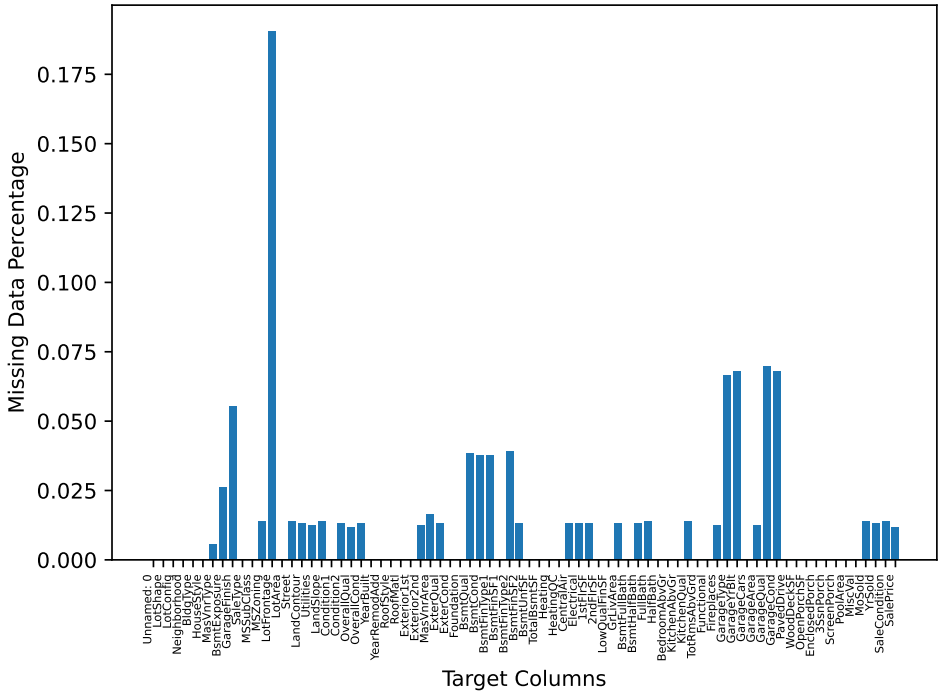

Figure 5.11: House dataset: New Dataset with a Generated Missing Situation

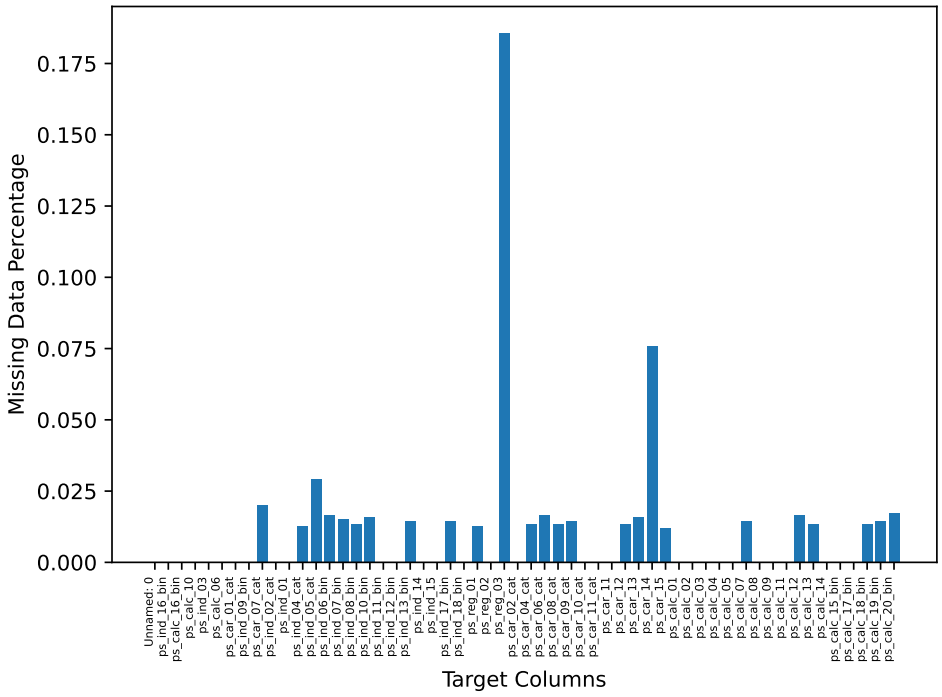

Figure 5.12: Insurance dataset: New Dataset with a Generated Missing Situation

5.2 Experimental Setup

In this experiment, we primarily utilize three machine learning models: **K-Nearest Neighbor (KNN), Random Forest (RF),** and **Decision Tree (DT)**. As outlined in the methodology section, our approach encompasses multiple predictive components, including data imputation to address missing values and enhance data completeness, optional feature selection to potentially reduce dimensionality and improve predictive performance, and culminating in the use of a bucket ensemble model for the final prediction. Thus, for our experiments, we first use all components of our prediction models, and subsequently, we remove certain components and observe the scores of predictive performance to analyze whether the inclusion of those components can assist in more accurate predictions. We repeat these processes with two additional datasets (house and insurance datasets) to validate the non-accidental predictive performance results.

In summary, our experiment includes:

1. KATE dataset
   - 1.1 (Data Imputation + Data Prediction) Use data imputation and then do data prediction based on the imputed dataset.
   - 1.2 (Data Imputation + Feature Selection + Data Prediction) Use data imputation and feature selection, then do data prediction based on imputation - narrowed down data subset. [1]
2. House dataset
   - 2.1 (Data Imputation + Data Prediction) Data imputation on manually generated datasets with a large number of missing values and then do data prediction based on the imputed dataset.
   - 2.2 (Data Imputation + Feature Selection + Data Prediction) Data imputation on manually generated datasets with a large number of missing values and do feature selection to narrow down the feature set that is used in prediction, then do data prediction based on the imputed and narrowed down data subset.
   - 2.3 (Data Prediction) Use the original dataset to do direct data prediction.
3. Insurance dataset
   - 3.1 (Data Imputation + Data Prediction) Data imputation on the manually generated datasets with a large number of missing values and then do data prediction based on the imputed dataset.
   - 3.2 (Data Imputation + Feature Selection + Data Prediction) Data imputation on the manually generated datasets with a large number of missing values and feature selection to narrow down the feature set that is used in prediction, then do data prediction based on the imputed and narrowed down data subset.
   - 3.3 (Data Prediction) Use the original dataset to do direct data prediction.

The settings for using or not using the internal modules of the entire experiment are very clear. We want to test the following:

- Data Imputation + Data Prediction, Data Prediction → Whether pre-filling missing data with data imputation can provide better data for more training, thereby achieving better prediction results.
- Data Imputation + Feature Selection + Data Prediction, Data Prediction → Whether reducing high-dimensional data features by selecting the most relevant feature values for the target feature after filling in missing data can reduce more noise and result in better predictions.
- Data Imputation + Feature Selection + Data Prediction, Data Imputation + Data Prediction → Whether reducing high-dimensional data features by selecting the most relevant feature values for the target feature, after filling in missing data, can reduce more noise and result in better predictions.

[1] Note here, we will not do an experiment on the KATE dataset using direct data prediction (predict the target using the original dataset) because the overall data in the KATE dataset is too incomplete, even after pre-processing, we can only find 20% of the complete data records relative to the total processed data, which is not enough for minimum training set amount.

In our experiments, in order to simulate the data and experiments of KATE Innovation, our testing set will be partitioned from the entire dataset. Regarding the assessment of predictive performance, we employ balanced accuracy and F1 score metrics to evaluate the accuracy of categorical feature predictions. For numerical feature data, the Mean Squared Error (MSE) method inherent to regression models is employed. These performance metrics are leveraged to quantify the alignment between the outcomes of our predictions and the actual data present in the original dataset.

5.3 Experiments and its Results

5.3.1 Experiments

KATE dataset

The following Table 5.3 is the results of using data imputation first to impute missing value and then do the prediction for our target features. Since our target features are categorical data with multi-class, we use the classifier model to do the prediction. In the table, we show the two models' performance when we evaluate them on the validation set, where DT is short for Decision Tree, RF is short for Random Forest, and KNN is short for K-Nearest Neighbor. As mentioned earlier, the best model for each target is the model with the highest balanced accuracy score among the three models. We also presented the balanced accuracy and F1 scores of each feature on the test set using the best model chosen from the validation set.

Table 5.3: Prediction Performance (Data Imputation and Data Prediction)

Feature Name	Performance on Validation Set (Balanced Accuracy)				Performance on Test Set (Using Best Model)	
	DT	RF	KNN	Best Model	Balanced Accuracy	F1 Score
car accessibility	0.8958	0.8889	0.8194	DT	0.5640	0.9223
general accessibility	0.9653	0.9167	0.7778	DT	0.8881	0.9281
public transport accessibility	0.8889	0.8750	0.7222	DT	0.7793	0.9041
location position	0.9653	0.9444	0.7708	DT	0.8962	0.9468
marketability conclusion	0.9236	0.9028	0.6944	DT	0.5956	0.9334
lettability of marketability	0.9236	0.9514	0.7292	RF	0.7536	0.9247
lettability period of marketability	0.8681	0.8958	0.7153	RF	0.6675	0.8222
saleability of marketability	0.9167	0.9792	0.7361	RF	0.8879	0.9151
saleability period of marketability	0.7917	0.7569	0.6111	DT	0.6524	0.7916
construction condition	0.9444	0.9722	0.7153	RF	0.6994	0.9548
facades condition	0.9028	0.9444	0.6944	RF	0.7536	0.9262
floors grnd condition	0.8681	0.9444	0.6528	RF	0.9001	0.9357
foundation condition	0.8889	0.8958	0.8819	RF	0.3481	0.9091
loadbearing condition	0.8264	0.8472	0.6319	RF	0.6892	0.8927
roofing condition	0.6181	0.7361	0.5694	RF	0.6459	0.7796
state maintenance condition	0.8194	0.8958	0.5208	RF	0.7243	0.9123
window frames grnd condition	0.8264	0.8750	0.5833	RF	0.6926	0.9273
parking facilities	0.5972	0.6667	0.5972	RF	0.4121	0.7972
parking location	0.7292	0.8542	0.7917	RF	0.3156	0.8351
location position current use	0.9583	0.9514	0.8194	DT	0.8025	0.9491
object assessment of marketability	0.7292	0.8611	0.7639	RF	0.4859	0.8813
object functionality	0.9097	0.9306	0.8889	RF	0.4583	0.9349
rental situation	0.7917	0.8681	0.7847	RF	0.5444	0.8448

We also incorporated feature selection on the imputed dataset after the data imputation process, aiming to identify highly correlated features with the target variable. We then utilize the selected feature subset to predict the target

feature. The data results with feature selection and data imputation are shown in Table 5.4, the models used and the performance measure scores are the same as in Table 5.3:

Table 5.4: Prediction Performance (Data Imputation, Feature Selection, and Data Prediction)

Feature Name	Performance on Validation Set (Balanced Accuracy)				Performance on Test Set (Using Best Model)	
	DT	RF	KNN	Best Model	Balanced Accuracy	F1 Score
car accessibility	0.9236	0.9097	0.9028	DT	0.7664	0.8835
general accessibility	0.8819	0.8819	0.8819	DT	0.7097	0.9209
public transport accessibility	0.9028	0.8958	0.8958	DT	0.6006	0.8549
location position	0.9514	0.9444	0.9514	DT	0.8792	0.9555
marketability conclusion	0.9722	0.9722	0.9236	DT	0.8668	0.9423
lettability of marketability	0.9167	0.9167	0.9236	KNN	0.7847	0.9313
lettability period of marketability	0.8403	0.8125	0.8403	KNN	0.3628	0.7490
saleability of marketability	0.9444	0.9444	0.9167	DT	0.9479	0.9462
saleability period of marketability	0.8264	0.8125	0.8194	DT	0.4033	0.6826
construction condition	0.9375	0.9375	0.9514	KNN	0.7298	0.8954
facades condition	0.9097	0.9028	0.9167	KNN	0.7830	0.8840
floors grnd condition	0.9167	0.9167	0.9306	KNN	0.4927	0.8518
foundation condition	0.9097	0.9097	0.9097	DT	0.3949	0.9256
loadbearing condition	0.9236	0.9236	0.9306	KNN	0.5596	0.8357
roofing condition	0.6944	0.6944	0.6806	DT	0.3257	0.5134
state maintenance condition	0.8750	0.8750	0.8958	KNN	0.5029	0.8556
window frames grnd condition	0.9236	0.9167	0.9028	DT	0.6567	0.8799
parking facilities	0.6806	0.6181	0.6806	KNN	0.2908	0.6882
parking location	0.6597	0.6667	0.6944	KNN	0.2005	0.6979
location position current use	0.9514	0.9444	0.9514	DT	0.6991	0.9527
object assessment of marketability	0.7778	0.7708	0.7847	KNN	0.3984	0.7951
object functionality	0.8889	0.8889	0.8889	DT	0.2647	0.8632
rental situation	0.7500	0.7639	0.7708	KNN	0.1648	0.2896

House Dataset and Insurance Dataset

For the house and insurance dataset, due to the generation of certain missing data instances, our experimental approach for predicting the target feature directly using a dataset containing missing values allows for a maximum utilization of only 65% of the data for training purposes. However, following the imputation of missing data, we can capitalize on 85% of the dataset for training. To assess the predictive auxiliary performance for the same variable while considering data imputation, we conducted two experiments on our datasets. The first experiment encompassed the application of all three predictive methods for training on the 65% data subset. Conversely, the second experiment involved predicting the target feature using the dataset subjected to data imputation (along with feature selection), employing 85% of the available data for training purposes.

Below are the results of the impact on data prediction with and without feature selection after data imputation, as well as the results of direct data prediction. Similar to the KATE dataset, we employed the same model and performance calculation metrics to observe the outcomes.

The experiments conducted on these two datasets involved training with 65% of the available data, as detailed in the tables provided in the A. Specifically, for the house dataset, refer to Table A.1 for scenarios where only data imputation is employed before the prediction, Table A.3 for cases where feature selection is incorporated after data imputation before prediction, and Table A.5 for utilizing a dataset containing manually dropped missing data for predictive analysis. For the insurance dataset, the results are presented in Table A.2, Table A.4, and Table A.6,

corresponding to situations involving using data imputation, the inclusion of feature selection during the predictive modeling phase, and direct prediction on a dataset with missing values.

Table 5.5: House dataset: Performance Scores with 65% of Data Used in the Training Set

Feature Name	Data Imputation		Data Imputation & Feature Selection		Only Direct Prediction	
	Bal Acc	F1-Score	Bal Acc	F1-Score	Bal Acc	F1-Score
LotShape	0.2981	0.5965	0.2934	0.5909	0.6852	0.7399
LotConfig	0.3152	0.6205	0.2760	0.6463	0.5128	0.7062
Neighborhood	0.5305	0.6107	0.2383	0.3178	0.3997	0.4889
BldgType	0.9350	0.9877	0.5164	0.8935	0.8294	0.9652
HouseStyle	0.7400	0.9376	0.3526	0.7615	0.7600	0.9394
MasVnrType	0.6127	0.8668	0.3337	0.5795	0.8239	0.8973
BsmtExposure	0.4282	0.6607	0.2540	0.5330	0.5032	0.6830
GarageFinish	0.6558	0.6739	0.3861	0.2873	0.8095	0.8160
SaleType	0.3330	0.9085	0.1620	0.8095	0.7377	0.9317

Table 5.6: House dataset: Performance Scores with 85% of Data Used in the Training Set (except direct prediction)

Feature Name	Data Imputation		Data Imputation & Feature Selection		Only Direct Prediction	
	Bal Acc	F1-Score	Bal Acc	F1-Score	Bal Acc	F1-Score
LotShape	0.6109	0.6995	0.4005	0.6023	0.6852	0.7399
LotConfig	0.4049	0.6731	0.2819	0.6230	0.5128	0.7062
Neighborhood	0.5786	0.6835	0.2374	0.3549	0.3997	0.4889
BldgType	1.0000	1.0000	0.2344	0.9284	0.8294	0.9652
HouseStyle	0.7814	0.9730	0.4140	0.6806	0.7600	0.9394
MasVnrType	0.7773	0.8838	0.4541	0.5968	0.8239	0.8973
BsmtExposure	0.4176	0.6691	0.3226	0.5954	0.5032	0.6830
GarageFinish	0.6998	0.7083	0.3255	0.2771	0.8095	0.8160
SaleType	0.6960	0.9128	0.4629	0.8658	0.7377	0.9317

5.3.2 Results Analysis

As for the KATE dataset results, based on the performance score evaluation results on the test set presented in Table 5.3 and Table 5.4, using the best model from the validation set, it can be observed that without feature selection (refer to Table 5.3), our F1 scores reflect excellent prediction results: among all the feature predictions, 65% of the results achieved evaluation scores above 0.9, with the lowest remaining prediction score being 0.78. This is an outstanding prediction outcome. Furthermore, considering the overall results after feature selection (refer to Table 5.4), as a multi-classification problem, our prediction results perform well overall. In terms of performance evaluation results for each feature, approximately 70% of our F1 scores are above an accuracy of 0.85, with around 50% of the features obtaining accuracy evaluation scores above 0.9. The prediction accuracy is very high.

In the context of the housing dataset, it can be observed from the result graphs that the performance of using data imputation surpasses those derived from direct predictive modeling in the absence of data imputation. We refer to Figures 5.19 and 5.20 for balanced accuracy and F1 score in terms of performance assessment, respectively.

Turning to the insurance dataset, an examination reveals that across most of the features, the three aforementioned prediction methods, along with variations in the volume of training data, yield similar performance outcomes. This observation is substantiated by the data illustrated in Figure 5.23 and Figure 5.24 for balanced accuracy and F1 score, respectively.

Table 5.7: Insurance dataset: Performance Scores with 65% of Data Used in the Training Set

Feature Name	Data Imputation		Data Imputation & Feature Selection		Only Direct Prediction	
	Bal Acc	F1-Score	Bal Acc	F1-Score	Bal Acc	F1-Score
ps_ind_16_bin	0.8686	0.9173	0.7372	0.8081	0.9038	0.9340
ps_calc_16_bin	0.5083	0.5284	0.5231	0.5705	0.5286	0.5711
ps_calc_10	0.0672	0.1071	0.0545	0.0971	0.1196	0.1249
ps_ind_03	0.1113	0.1370	0.0629	0.0833	0.1364	0.1335
ps_calc_06	0.1593	0.2268	0.1153	0.1753	0.1433	0.1388
ps_car_01_cat	0.1720	0.4323	0.0719	0.2293	0.2657	0.4992
ps_ind_09_bin	0.9742	0.9872	0.5691	0.7571	1.0000	1.0000
ps_car_07_cat	0.5000	0.9356	0.5000	0.9356	0.5000	0.9049
ps_ind_02_cat	0.2812	0.6904	0.2437	0.6170	0.2771	0.6710

Table 5.8: Insurance dataset: Performance Scores with 85% of Data Used in the Training Set (except direct prediction)

Feature Name	Data Imputation		Data Imputation & Feature Selection		Only Direct Prediction	
	Bal Acc	F1-Score	Bal Acc	F1-Score	Bal Acc	F1-Score
ps_ind_16_bin	0.8577	0.8706	0.7428	0.8147	0.9038	0.9340
ps_calc_16_bin	0.5238	0.5590	0.5054	0.5313	0.5286	0.5711
ps_calc_10	0.0612	0.0866	0.0733	0.0947	0.1196	0.1249
ps_ind_03	0.1053	0.1165	0.0859	0.0904	0.1364	0.1335
ps_calc_06	0.1373	0.1989	0.1610	0.1928	0.1433	0.1388
ps_car_01_cat	0.1769	0.4654	0.1008	0.2651	0.2657	0.4992
ps_ind_09_bin	0.9730	0.9920	0.4890	0.7147	1.0000	1.0000
ps_car_07_cat	0.5000	0.9233	0.4971	0.9156	0.5000	0.9049
ps_ind_02_cat	0.3279	0.6615	0.2256	0.5836	0.2771	0.6710

Figures 5.13 and 5.14 presented below illustrate the performance scores obtained from two different pre-processing prediction methods. The yellow line corresponds to the prediction achieved through a combination of data imputation and feature selection, while the blue line represents the prediction solely based on data imputation. From the balanced accuracy and F1 score's results (see Figures 5.13 and 5.14, we can observe that using feature selection's performance score for each feature performs worse than not using feature selection, as for most of the features, the blue line is always higher than the yellow line on both of the figures).

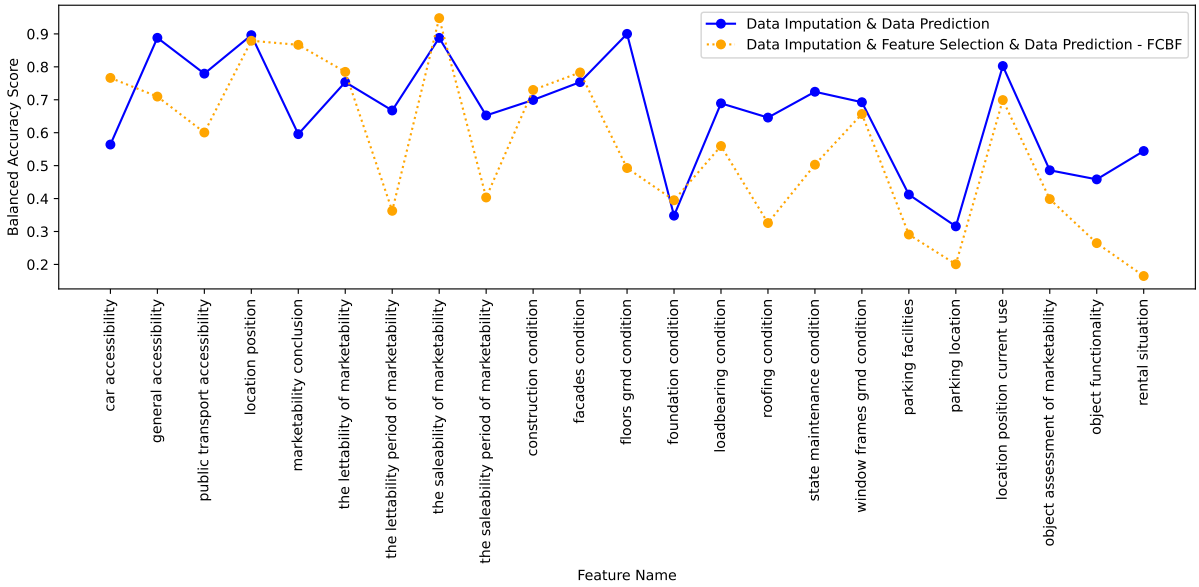

Figure 5.13: Balanced Accuracy Score of Each Feature

When looking at the outcomes of applying the feature selection preprocessing technique, it yielded lower accuracy compared to not using the preprocessing. We found two primary factors that might explain this. Firstly, there's the issue of data distribution along with a multitude of class labels, resulting in a highly imbalanced data distribution where different classes lack even representation. Secondly, the characteristics of the dataset's own features play a role in influencing the outcomes of the feature selection process.

To visually represent the relationship between prediction performance and the number of unique class labels associated with each feature, we generated Figures 5.15, and 5.16 for each performance measurement. These figures illustrate how the performance varies based on the unique class labels for each feature.

We also show the same figure results for the house and insurance datasets. For the house dataset, Figures 5.17 and 5.18 respectively display the balanced accuracy and F1 scores for each target when using 65% of data records as the training set. On the other hand, Figures 5.19 and 5.20 demonstrate the balanced accuracy and F1 performance on the test set when we use 85% of data records as the training set on the dataframe that underwent data imputation. The data imputation process allowed us to increase the available data by handling missing values before conducting predictions.

We also generate the same performance score Figures for the insurance dataset: Figures 5.21 and 5.22 for balanced accuracy and F1 performance score on the test set when only using 65% of data from the dataframe as the training set, and Figures 5.23 and 5.24 for balanced accuracy and F1 score when using 85% of data records as the training set.

Using Figures 5.15 and 5.16, where we plotted the graph of performance scores and the number of category labels, we attempt to analyze and explain why the inclusion of feature selection affects the lower scores in some features.

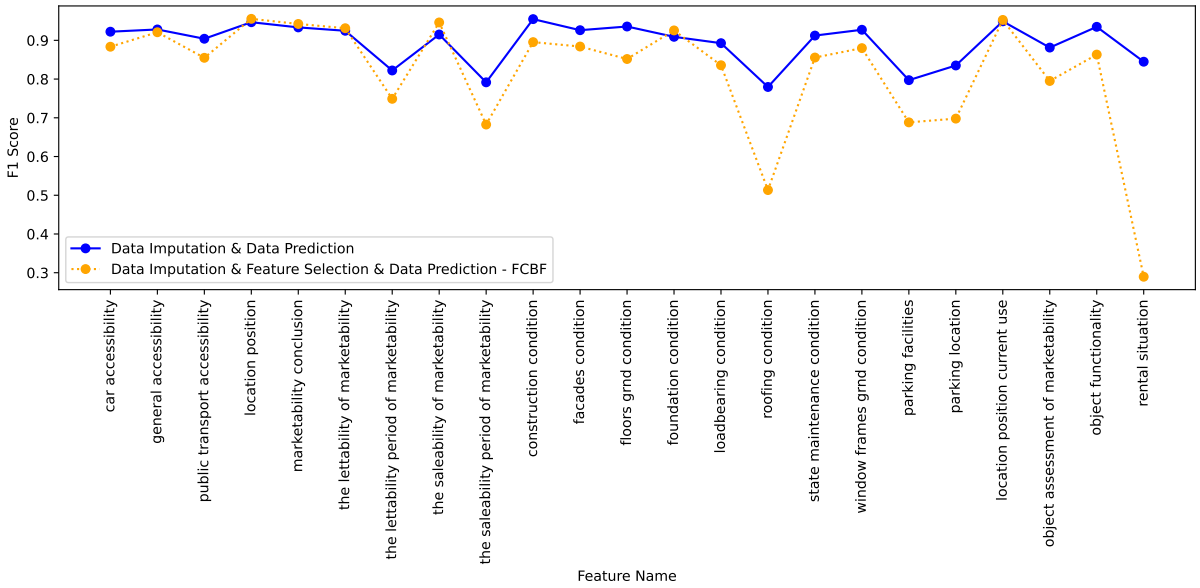

Figure 5.14: F1 Score of Each Feature

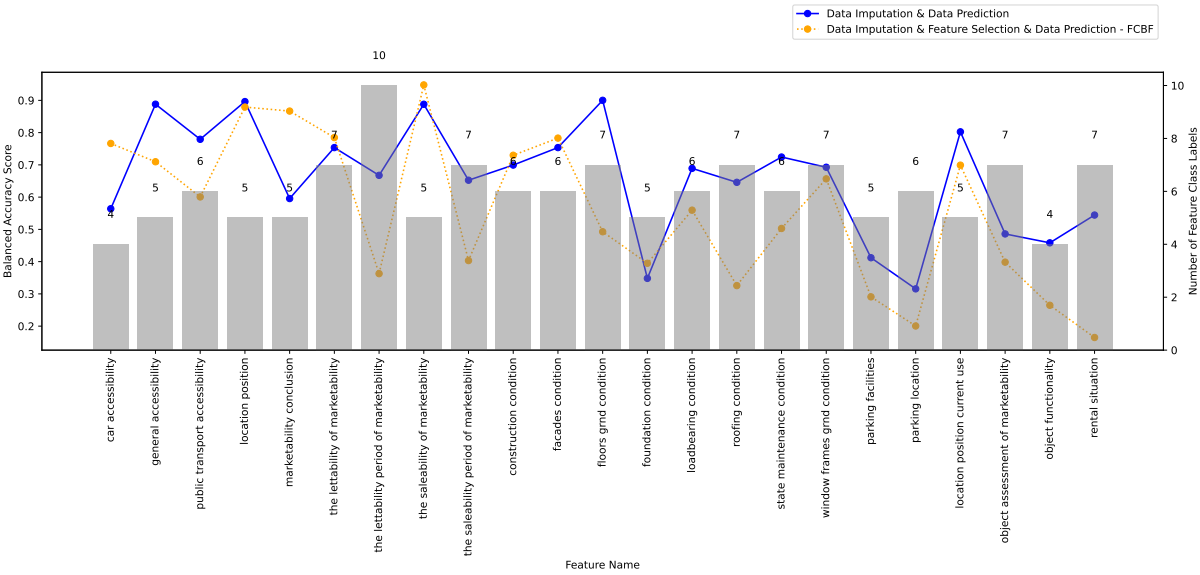

Figure 5.15: Balanced Accuracy Score with the Number of Each Feature's Class Labels

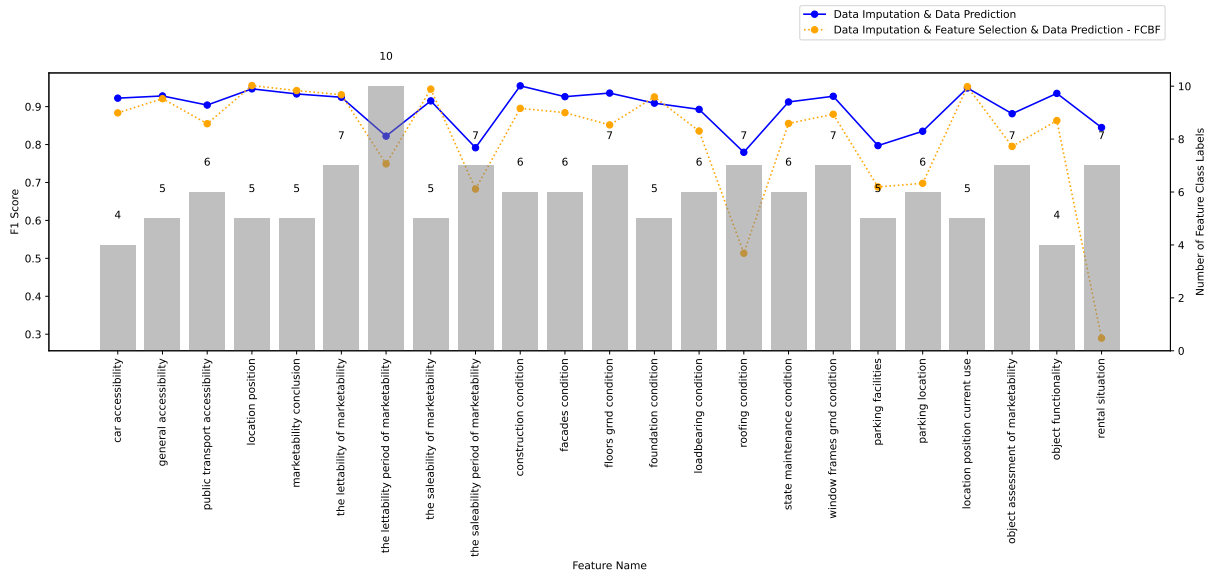

Figure 5.16: F1 Score with the Number of Each Feature's Class Labels

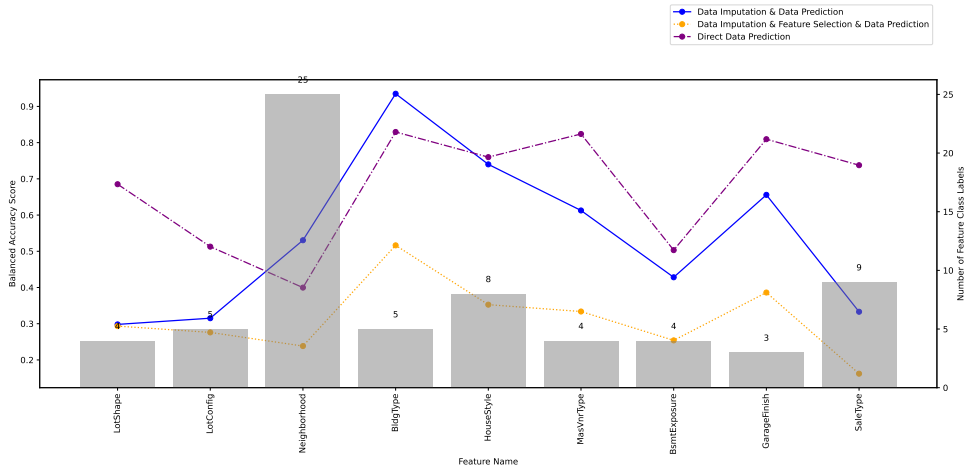

Figure 5.17: House dataset: Balanced Accuracy Scores with 65% of Data Used in Training Set

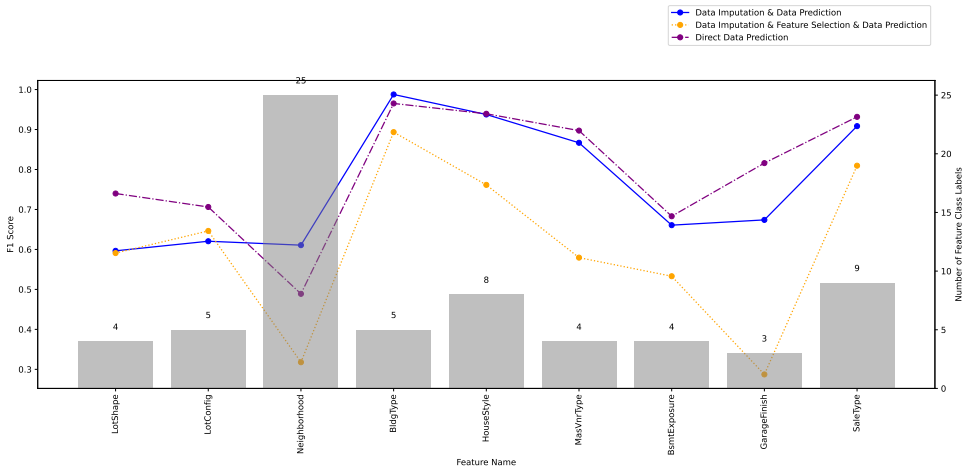

Figure 5.18: House dataset: F1 Scores with 65% of Data Used in Training Set

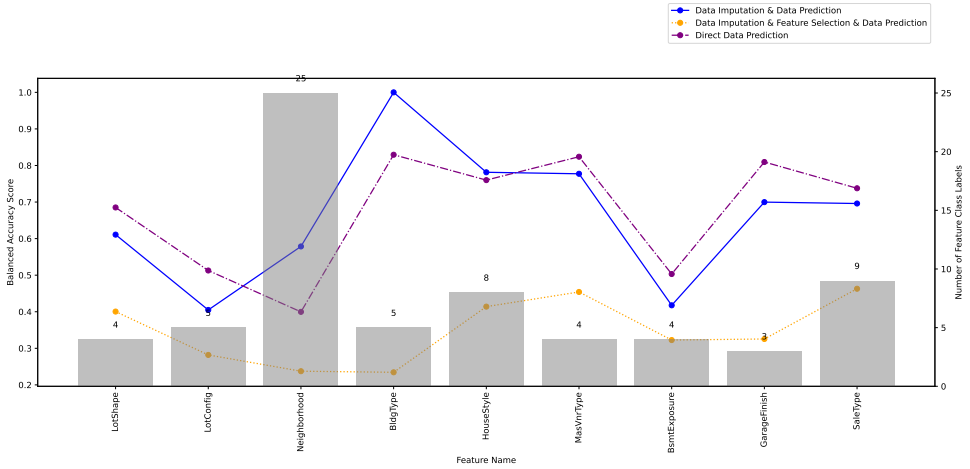

Figure 5.19: House dataset: Balanced Accuracy Scores with 85% of Data Used in (Data Imputation & Data Prediction) and (Data Imputation & Feature Selection & Data Prediction) Training Set

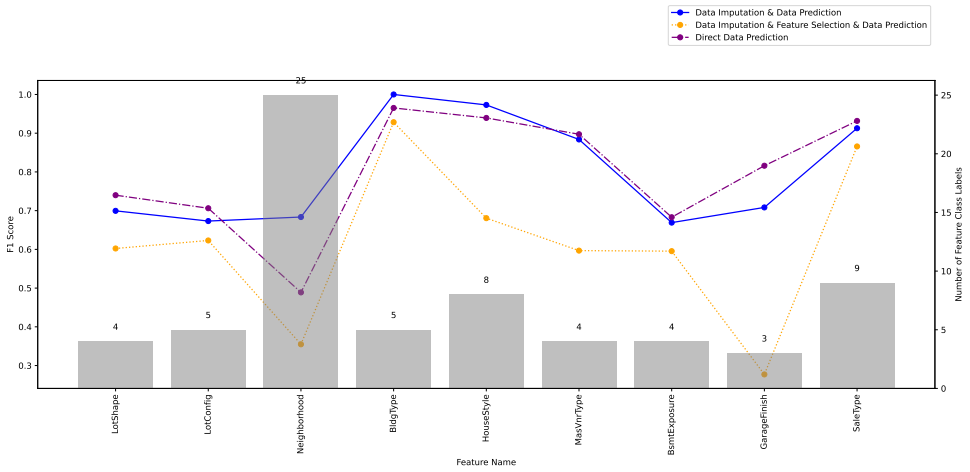

Figure 5.20: House dataset: F1 Scores with 85% of Data Used in (Data Imputation & Data Prediction) and (Data Imputation & Feature Selection & Data Prediction) Training Set

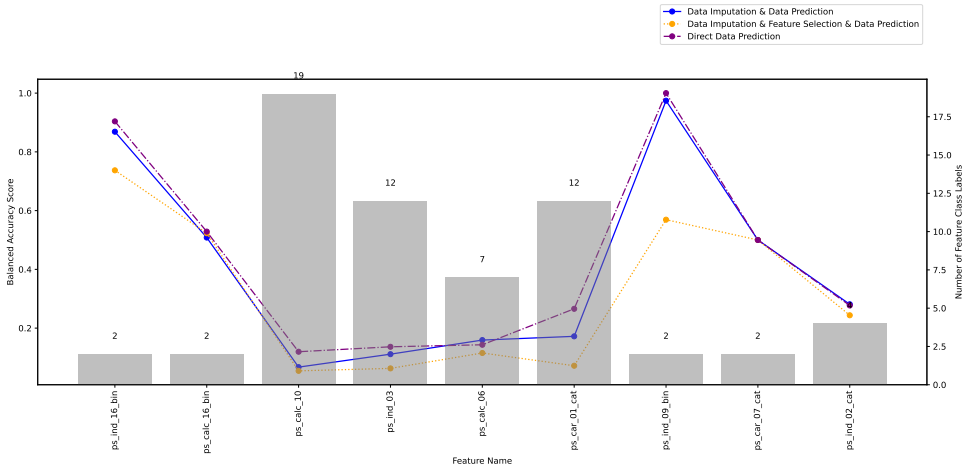

Figure 5.21: Insurance dataset: Balanced Accuracy Scores with 65% of Data Used in Training Set

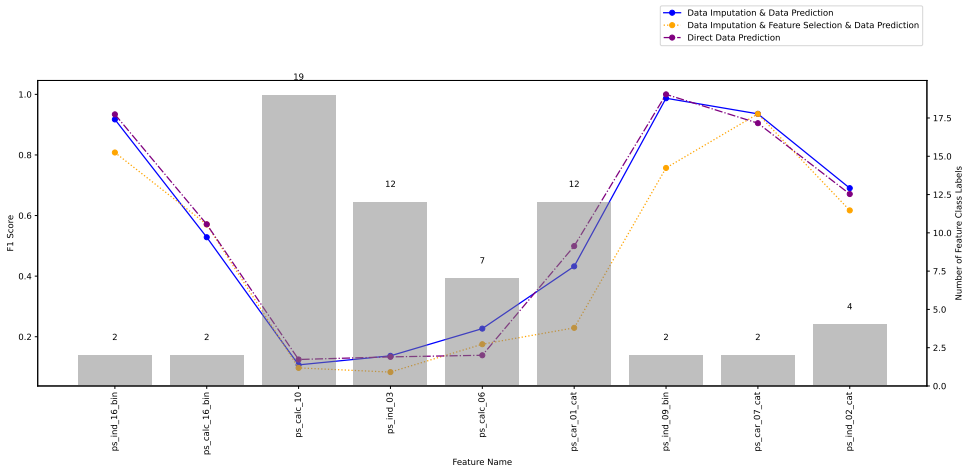

Figure 5.22: Insurance dataset: F1 Scores with 65% of Data Used in Training Set

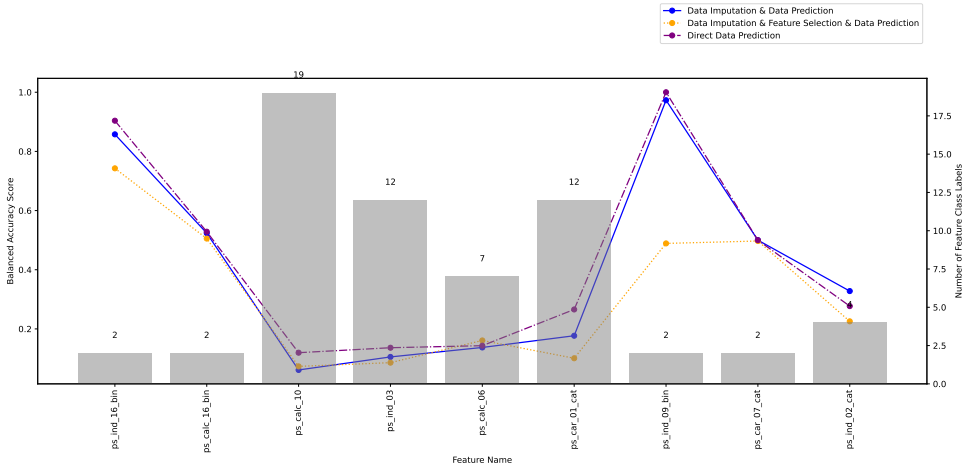

Figure 5.23: Insurance dataset: Balanced Accuracy Scores with 85% of Data Used in (Data Imputation & Data Prediction) and (Data Imputation & Feature Selection & Data Prediction) Training Set

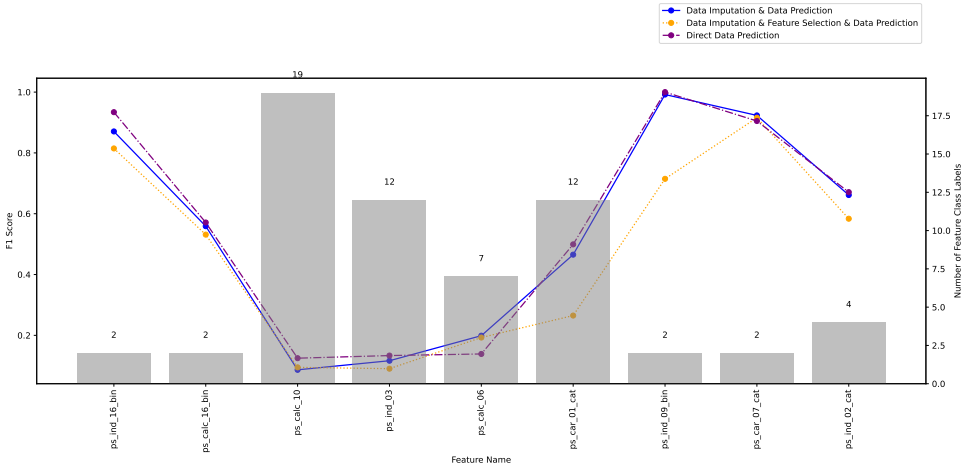

Figure 5.24: Insurance dataset: F1 Scores with 85% of Data Used in (Data Imputation & Data Prediction) and (Data Imputation & Feature Selection & Data Prediction) Training Set

We take the **rental situation** feature for example, where this feature gets the lowest prediction score when using feature selection. We hypothesized that this might be related to the distribution of data in the categorical data. Therefore, we observed the number of category labels for each feature. Since it is a multi-classification problem, this means that if there are too many classes, then the accuracy of the classification will be reduced as well. From these three figures, we found that when there are more classification labels for a feature, such as the **lettability of marketability**, with 10 different classes, the results show that the performance of using feature selection is lower than when not using feature selection, and, we believe that the distribution of the data also affects the accuracy of the prediction. For example, we see that for **object assessment of marketability** and **rental situation**, both features have the same number of class labels, 7, but from Figure 5.5, we can see that the distribution of the feature **rental situation** is not the same as that of **object assessment of marketability**. However, from Figure 5.5, the distribution of data for the feature **rental situation** is more even than that for the feature **object assessment of marketability**. In other words, from Figure 5.5, we can see that more data is distributed in the "na" and "excellent" class labels in the **rental situation**, while in the **object assessment of marketability**, the data are almost only distributed in the three class labels.

This situation also shows in the house and insurance datasets. As a practical illustration of this point, we analyzed the **Neighborhood**, **BlogType**, **MasVnrType**, and **BsmtExposure** features in the house dataset, using balanced accuracy evaluation results with 65% of training data as an indicator of actual performance shown in Figure 5.17. This is because balanced accuracy can more accurately reflect the prediction results under conditions of imbalanced data distribution, and also with unified variants. For the **Neighborhood** feature, even though it had the highest category label count of 25, its performance was subpar. The **BlogType** feature, on the other hand, performed exceptionally well despite an uneven distribution of data and a high concentration of values in the **1fam** category, which likely reduced the variety of options and increased the likelihood of correct prediction.

On examining the **MasVnrType** and **BsmtExposure** features, it was found that the performance of **MasVnrType** was superior to **BsmtExposure**. This difference in performance can be attributed to their data distribution, as seen in the distribution charts (Figure 5.6). Compared to **MasVnrType**, **BsmtExposure** exhibited a more uniform distribution across the **Av**, **Gd**, and **Mn** categories, increasing the likelihood of error in predictions, thereby affecting overall performance scores.

Comparing **LotConfig** and **BldgType**, which have a similar count of category labels, the data distribution for **LotConfig** is more uniform. This results in a greater number of values for other categories and, in turn, can hinder the performance of a multi-classification model. In contrast, for **LotShape**, which has fewer category

labels than **BldgType**, the data distribution is also more uniform than **BldgType**, potentially disrupting the multi-classification model performance. For the **SaleType** feature, which has a similar imbalanced distribution as **BldgType**, the high count of category labels could potentially interfere with the results of the multi-classification model.

The same analysis was applied to the insurance dataset. As illustrated in Figure 5.21, features like **ps_calc_10, ps_ind_03, ps_calc_06,** and **ps_car_01_cat** contain 19, 12, 7, and 12 different label classes respectively. Although **ps_ind_03** and **ps_car_01_cat** have the same number of label classes, the better performance of **ps_car_01_cat** is due to its less balanced distribution compared to **ps_ind_03**, as can be clearly seen in Figure 5.10. The data for **ps_ind_03** are similarly distributed among almost all label classes, while the values of **ps_car_01 cat** are primarily concentrated in a few classes. This result supports our argument: having a balanced distribution of data might create difficulties in predicting multi-class problems. As data imputation serves as the cornerstone of feature selection, incorrect predictions may be amplified after feature selection, leading to a decline in accuracy performance in the final prediction results.

This result appears to align with our expectations. When dealing with a greater number of potential predictions, the results become more reliant on additional features for in-depth analysis. For instance, if we have a wider array of fruits to choose from, we would anticipate requiring more information across various dimensions to make accurate assessments. However, due to the limitation of the threshold for calculating the distance and filtering in feature selection, we may filter out some other features that may have potential information value for this feature, and only select other mainly valid features that mostly pass the threshold, as our feature selection method - FCBF, only selects minimal-optimal feature subsets and discards weakly relevant and redundant features.

Moreover, it is hypothesized that these results might be an outcome of the significant number of missing values in other features of the dataset. Despite our attempts to impute these missing values, inaccuracies in the imputed data are inevitable. These inaccuracies could potentially lead to two problems. Firstly, the feature selection might be erroneous due to these inaccuracies, mistakenly determining strong correlations between the inaccurate feature and the target feature. Secondly, feature selection involves utilizing a limited set of feature values to predict the target. Hence, if substantial deviations exist in these limited features without a plethora of other features to correct these significant deviations, the application of such features could result in erroneous outcomes.

For instance, consider a scenario where we initially have 20 variables to calculate the target value and assume that 5 of these variables contain highly deviated values. In such a case, we still have the remaining 15 variables providing accurate information from other dimensions to compute the target variable correctly. However, after feature selection, we only employ 10 of these variables to compute the target feature. Since these 10 features have a greater influence on the target feature than the rest - as they are most closely related - it leads to two issues. Firstly, any errors in these variables will have a significant impact on the computational results, causing large error values. Secondly, the lack of a multitude of other features to provide information from other dimensions means that these errors cannot be "pulled back" in the correct direction. As such, the use of feature selection might sometimes reduce prediction accuracy.

Besides, another reason could be that this dataset is not suitable for using a minimal-optimal feature subset as FCBF proposed. We believe that since this dataset was provided by the real estate appraisal experts at KATE Innovations, then for them, when they provided the data, they potentially pre-selected the features for us. Since they will only enter data for features that are full of meaning, and we removed all features that did not exceed 70% completeness at the beginning, it means that we have already pre-pruned features that are irrelevant and redundant, while the remaining features are all very highly correlated. Therefore, performing feature selection may overlook some features that provide potential information because of the threshold limit.

Naturally, as the number of features increases, we have more dimensions available for our model's computation and analysis, which can consequently lead to higher accuracy. On the other hand, it is also known that endlessly adding features does not always contribute useful information to the overall prediction; instead, it may introduce more interfering factors.

Since we have already set the threshold for the FCBF feature selection method to the minimum of 0, we cannot change the threshold. In order to analyze this influence factor, and search for potential ways to increase the performance of using feature selection methods, we focused on the feature with the most significant performance

score difference between using feature selection and not: the **rental situation**. We began with a subset of features selected through the feature selection process, gradually added more features to this subset, and then used this expanded feature subset for the prediction of our target feature.

We use correlation score as a forward selection measurement, meaning that for each unselected feature after the feature selection process, we calculate its correlation with the target. We select the most correlated feature to add to the new feature subset and then examine the impact of each added feature on the overall performance metrics.

In this experiment, we use balanced accuracy as our performance measurement because it functions better with imbalanced data distributions. This approach allows us to thoroughly evaluate the impact of additional features and effectively distinguish between beneficial and detrimental inclusions.

Looking at the results from Figure 5.25, from the fitted line (the red line) in the resulting chart, it can be observed that, as we add approximately 0 to 20 features, the performance score increases almost with each addition, and the overall rate of increase is relatively rapid. Once we reach around 20 to 25 features, our balanced accuracy begins to peak and gradually stabilizes. During the period of stability, adding between 30 to 50 features exhibits a declining trend. However, this trend gradually levels off after 50 features, and the performance score fluctuates around 0.55, indicating a state of relative stability.

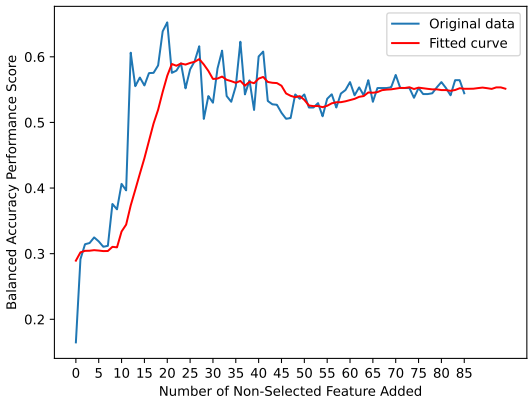

Figure 5.25: KATE dataset: The Relationship Between Balanced Accuracy and The Number of Features Used To Make Predictions on the Rental Situation Feature

This outcome suggests two key points. Firstly, with respect to the **rental situation** feature, the subset of features chosen through our feature selection process may not provide complete support for the calculation of the target feature. Secondly, the most optimal use of the feature subset occurs when using approximately 20 to 25 features.

We conducted the same experiment on the **Neighborhood** feature from the house dataset as we did with the **rental situation** feature in the KATE dataset - we gradually added feature values to the feature subset used to predict the **Neighborhood** feature. As shown in Figure 5.26, we can see that as the number of feature values increases, the resulting balanced accuracy also increases, eventually reaching the result of data prediction using data imputation without using feature selection. Moreover, when the number of features increases to about 40-45, our balanced accuracy starts to reach its peak and gradually stabilizes. Before 40, there is a drastic increase in accuracy when increasing about 0-5 features, approximately around 0.1. These results mean that only when using approximately more than 40 features, our performance score can reach a stable value, and when using more features, the overall prediction performance does not decrease.

Furthermore, on the insurance dataset, in terms of the changes in prediction performance brought about by adding additional features to the feature subset obtained after feature selection, we focus here on the extension of the feature subset for **ps_ind_09_bin**. As shown in Figure 5.27, the obtained results are quite intriguing. From the

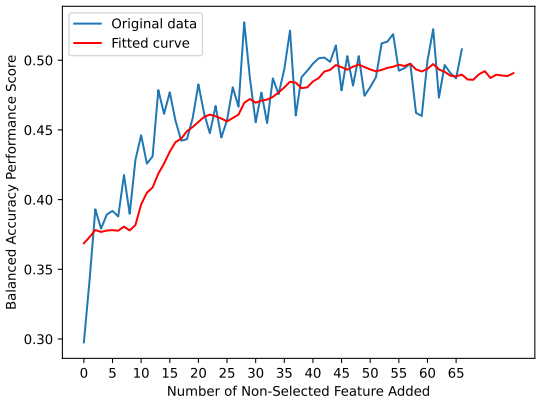

Figure 5.26: House dataset: The Relationship Between Balanced Accuracy and The Number of Features Used To Make Predictions on Neighborhood Feature

figure, it appears that we can only obtain satisfactory prediction performance results after gradually increasing the number of features used for prediction to all feature values. This implies that using feature extraction does not apply to this feature. This result can also be observed from the three performance evaluation result charts: for this feature, all yellow lines (dots) are the lowest (worst performance).

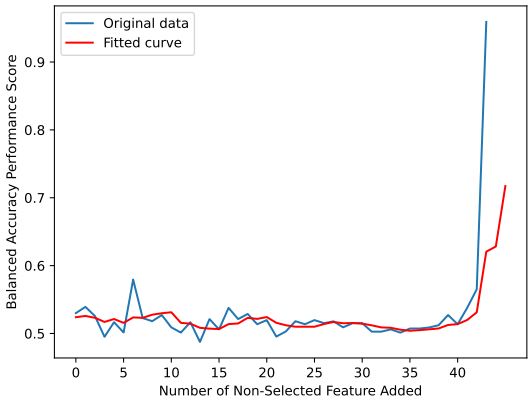

Figure 5.27: Insurance dataset: The Relationship Between Balanced Accuracy and The Number of Features Used To Make Predictions on ps_ind_09_bin Feature

It is important to acknowledge that multiple factors could potentially influence the performance of feature selection, especially the state of data imputation. Therefore, these findings need to be interpreted considering the complexity and variability of the underlying data dynamics.

Besides the performance score, observing from Table 5.4, Table A.3 and A.4 in A, where feature selection is not employed, Decision Tree (DT) and K-Nearest Neighbors (KNN) can outperform Random Forests. This is because of the algorithm's own characteristics. Decision Tree is able to capture complex decision boundaries and interactions between features, while KNN effectively measures distances in the feature space. Random Forests, although known for handling high-dimensional data, may not provide significant improvements in performance

when the number of features is limited. This can be ascribed to the inherent characteristics of the Random Forest as an ensemble model, amalgamating numerous Decision Trees. Random Forest is generally adept at managing intricate interrelations between features and targets, as well as dealing with outliers and noise within the data. Consequently, it exhibits commendable performance when an extensive set of features is utilized for prediction. Moreover, when only using data imputation to do pre-calculating, the best model that is most used is Random Forest as we can check from Table 5.3, A.1 and A.2, which also indicates that with more features, the Random Forest model performs better than the other two models.

5.4 Feature Selection Improvement

Due to our previous analysis, the predictive performance results generated by using feature selection are lower compared to not using it. To analyze the reasons behind this issue, we conducted experiments on the feature subset selected by the feature selection process. We continuously added other features highly related to the target feature to the selected dataset and used the newly selected feature subset to predict the target feature. We observed the trend in predictive performance results as features were incrementally added. Refer to Figure 5.25, 5.26, and 5.27.

We attribute this situation to the nature of the dataset itself. For instance, experts in real estate assessment from KATE Innovations have already assisted us in selecting a highly correlated feature set. The interconnections among the features in the set are highly correlated. However, the FCBF algorithm focuses on seeking a minimal-optimal feature subset and overlooks weakly irrelevant but redundant features. Thus, the computed feature subset doesn't truly serve the purpose of feature selection - reducing irrelevant feature noise and enhancing prediction accuracy. To address this, we made some simple enhancements to FCBF, referred to as FCBF+.

The working principle of FCBF+ is quite similar to the experimental process we previously conducted to observe the impact of increasing features on predictive results. In terms of principle, the improved FCBF+ method combines wrapper and filter approaches for feature selection and employs the forward selection method. Specifically, we gradually incorporate the most correlated features with the target feature into the feature subset already chosen by FCBF. Subsequently, we calculate the performance (keeping the experimental settings consistent as previously) and compare it with the results from the five features before and after (i.e., a total of 10 features). If adding a certain feature leads to the best-calculated result, we stop adding new features to the selected feature subset, thus achieving the best predictive performance. We tested the improved algorithm on the KATE dataset, and the experimental results are presented in Figure 5.28 and 5.29 for balanced accuracy and F1 score, respectively.

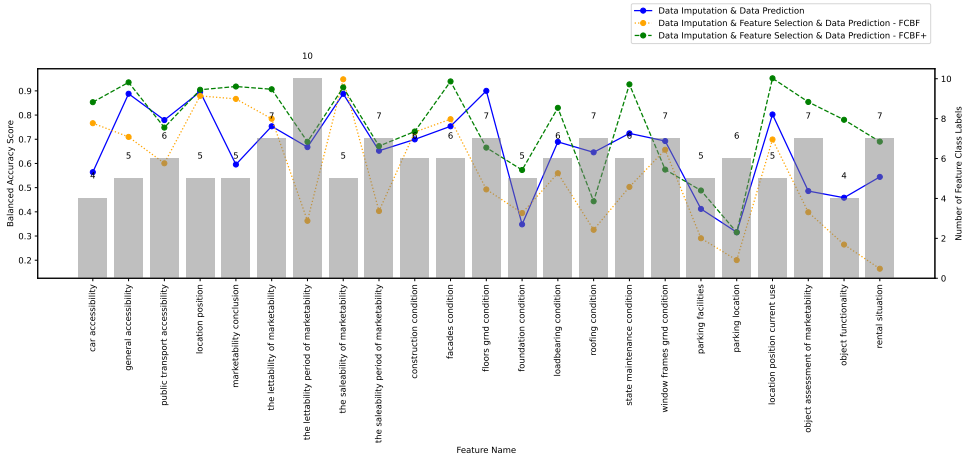

Figure 5.28: KATE dataset: Balanced Accuracy Scores with Improved Feature Selection Method

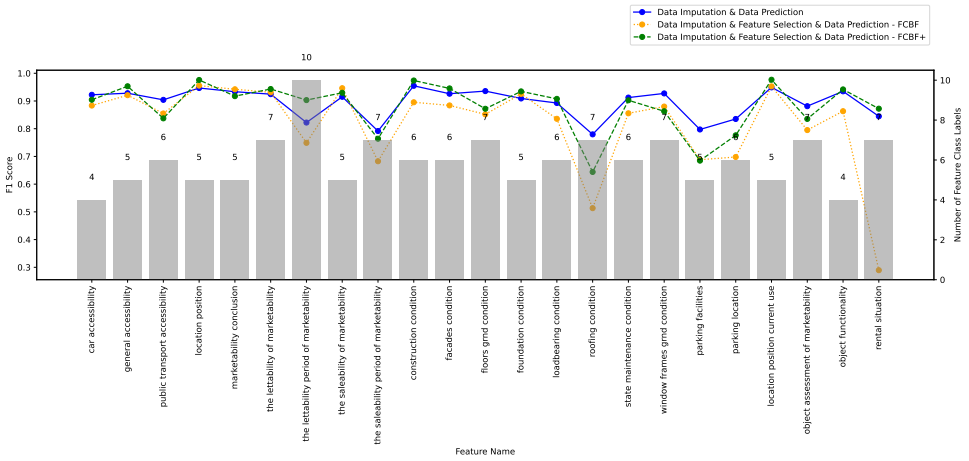

Figure 5.29: KATE dataset: F1 Scores with Improved Feature Selection Method

From the two result figures, we observe that this enhanced feature selection method leads to an improvement in prediction performance scores. The most significant improvement is evident in the case of balanced accuracy, as shown in Figure 5.28. When compared to the original FCBF method, the use of the improved feature selection method demonstrates a pronounced performance enhancement. The results align with the advantages of employing feature selection, as depicted in the graph, where it is evident that, for the majority of features, the FCBF+ method performs exceptionally well (the green line is at its peak). Furthermore, the F1 score benefits from the FCBF+ method, yielding higher accuracy results compared to previous outcomes, as shown in Figure 5.29.

5.5 Experiments on Large Dataset

Since our previous experiments were conducted on relatively small datasets, we aim to apply our model and experiments to larger datasets to observe their performance. In this experiment, we utilize the TIC2000 dataset from the UCI dataset collection [101], referred to as **UCI dataset**. This dataset, used in the CoIL 2000 Challenge, contains information about customers of an insurance company. It comprises 86 variables, and 5822 data records, and includes both product usage and socio-demographic data. The distribution of this dataset, along with the results of preprocessing the dataset for missing data, similar to our previous experiments, are illustrated in Figures A.1 and A.2, as shown in Appendix A.

We have chosen the following features as our target variables (along with their corresponding information): **"MBERHOOG"** (High status), **"MBERZELF"** (Entrepreneur), **"MBERBOER"** (Farmer), **"MBERMIDD"** (Middle management), **"MBERARBG"** (Skilled laborers), and **"MBERARBO"** (Unskilled laborers).

Similarly, we conducted six experiments using three prediction methods on two different training datasets. The experimental results are presented in the following tables and figures: Table 5.9 showcases the performance scores of the three methods when trained on 65% of the data. As before, Bal Acc and F1 scores represent the balanced accuracy and F1 score performance results, respectively, on the test set using the best model. The corresponding graphs are depicted in Figures 5.30 and 5.31. Table 5.10 displays the performance results (excluding direct predictions) when using 85% of the data for training. The graphs illustrating different features, each with balanced accuracy and F1 score results, can be found in Figures 5.32 and 5.33, including the use of improved feature selection method (FCBF+).

From the result charts, we can observe that on large datasets, the accuracy of our data prediction after performing data imputation shows a significant improvement compared to not using it, as shown in Figure 5.33. Additionally, the use of the improved feature selection method yields enhancements in computational performance.

Table 5.9: Performance Scores with 65% of Data Used in Training Set

Feature Name	Data Imputation		Data Imputation & Feature Selection		Only Direct Prediction	
	Bal Acc	F1-Score	Bal Acc	F1-Score	Bal Acc	F1-Score
MBERHOOG	0.8981	0.9263	0.7798	0.7960	0.8422	0.9125
MBERZELF	0.8053	0.9670	0.5421	0.7716	0.7976	0.9537
MBERBOER	0.9093	0.9665	0.2691	0.6956	0.9206	0.9748
MBERMIDD	0.8969	0.9083	0.7210	0.7064	0.9034	0.8966
MBERARBG	0.9065	0.9195	0.6150	0.7645	0.9126	0.8967
MBERARBO	0.8863	0.8902	0.7303	0.7545	0.9068	0.9039

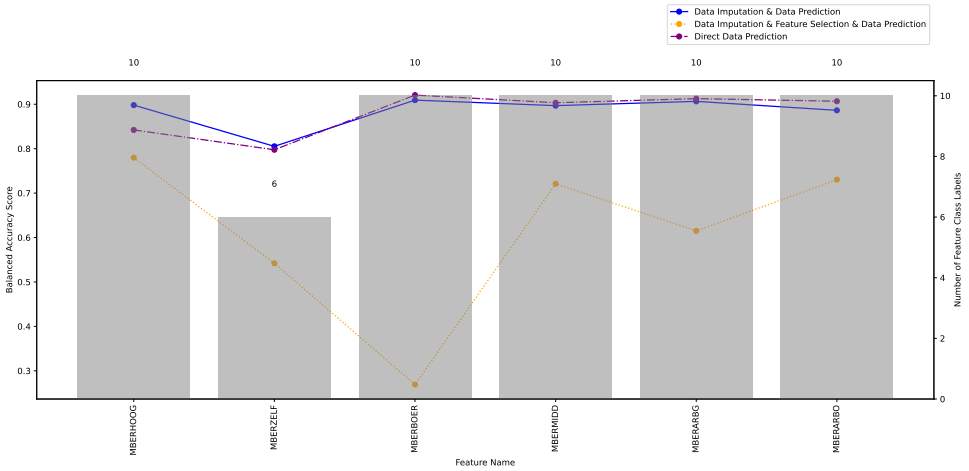

Figure 5.30: UCI dataset: Balanced Accuracy Scores with 65% of Data Used in Training Set

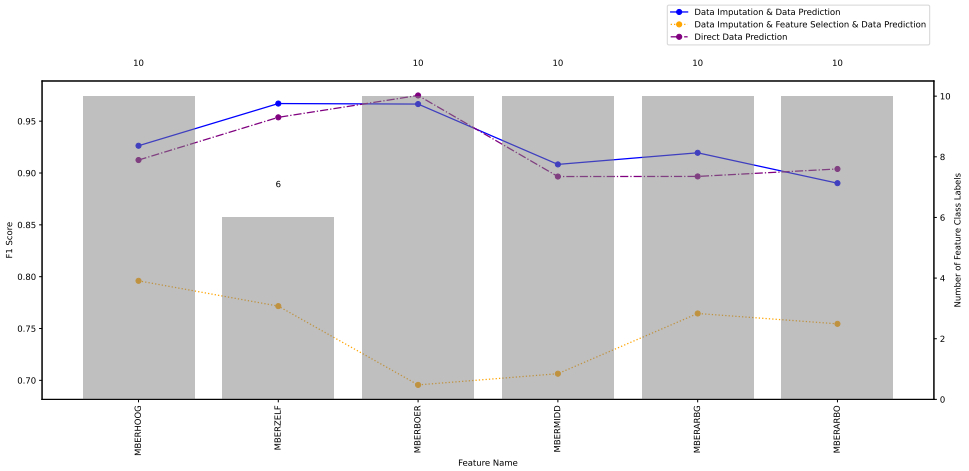

Figure 5.31: UCI dataset: F1 Scores with 65% of Data Used in Training Set

Table 5.10: Performance Scores with 85% of Data Used in Training Set (expect direct prediction)

Feature Name	Data Imputation		Data Imputation & Feature Selection		Only Direct Prediction	
	Bal Acc	F1-Score	Bal Acc	F1-Score	Bal Acc	F1-Score
MBERHOOG	0.9101	0.9346	0.7587	0.7996	0.8422	0.9125
MBERZELF	0.7432	0.9783	0.4598	0.7763	0.7976	0.9537
MBERBOER	0.8418	0.9560	0.2959	0.6923	0.9206	0.9748
MBERMIDD	0.8905	0.9302	0.7146	0.7063	0.9034	0.8966
MBERARBG	0.9307	0.9383	0.6113	0.7152	0.9126	0.8967
MBERARBO	0.9371	0.9120	0.7928	0.7795	0.9068	0.9039

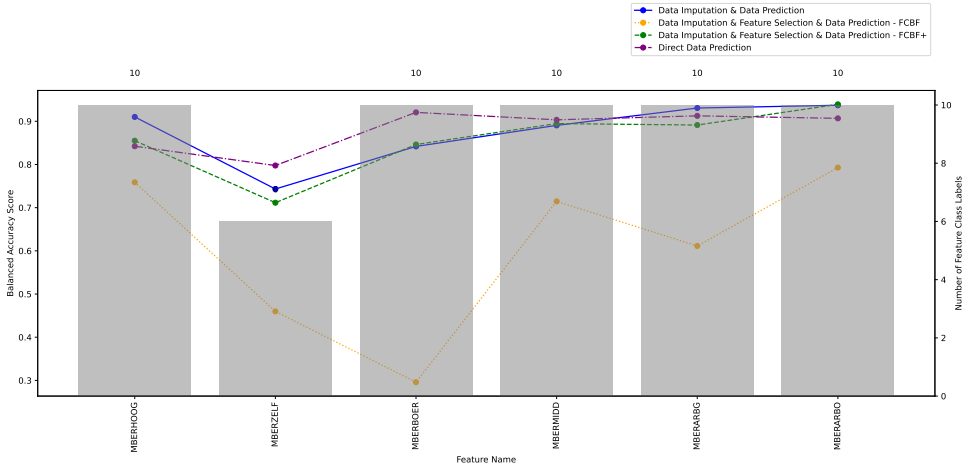

Figure 5.32: UCI dataset: Balanced Accuracy Scores with Improved Feature Selection Method (Using 85% Trained Data)

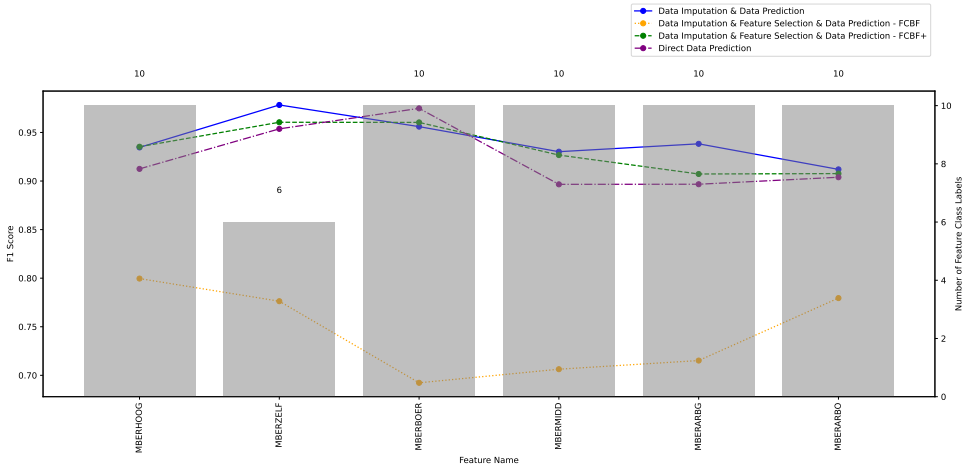

Figure 5.33: UCI dataset: F1 Scores with Improved Feature Selection Method (Using 85% Trained Data)

5.6 Comparison with Statistical Imputation Method

In our previous experiments and their analysis, we mostly focus on the feature selection model, some explanations of its performance, and its improvement method. However, the data imputation is the most important component of our entire prediction methodology and is the cornerstone of the entire experiment conducted, for which we want to compare and compute its performance. We unit-tested our Bucket4Imp method in the previous Chapter 4, and the result shows it performed excellently. As mentioned in the previous section, data imputation methods also include traditional statistical approaches such as using frequent values for imputation. Inspired by the research of Jerez et al. [43], we conducted performance experiments comparing the imputation methods using statistical approaches (mode) with our imputation model. We applied both imputation methods to the house dataset and the insurance dataset, comparing the quantity of different data type labels and the computational capabilities of machine learning-based imputation methods and statistical-based imputation methods. We aim to discuss the significance and applicability conditions of using machine learning-based imputation.

Mode is a statistical concept that represents the most frequently occurring value in a dataset. In the context of data imputation, which is the process of filling in missing values in a dataset, the mode method is used to replace missing values with the mode of the respective variable. The mode method is commonly employed when dealing with categorical or discrete variables, where the missing values can be imputed with the most common category or value. This approach assumes that the mode represents the typical or representative value for the variable and helps maintain the distribution and patterns within the dataset.

Given that the house and insurance datasets inherently have high data completeness, in this experiment, we only discarded features that contained over 30% missing values in the original dataset - the same preprocessing used in all previous experiments. We randomly generated some missing values and applied both our data imputation methods and the statistical mode method to the same dataset with randomly missing data, observing the accuracy of each method under different evaluation standards. To avoid possible chance effects caused by the variability of machine learning model predictions, we repeated our model operations 100 times and took the average of the results to mitigate randomness. The graphs below, 5.34 and 5.35, apply this comparison experiment to the house dataset. The former uses balanced accuracy as the performance evaluation criterion, while the latter uses the F1 score. The balanced accuracy results of the insurance dataset are shown in 5.36, with its F1 score results reflected in 5.37.

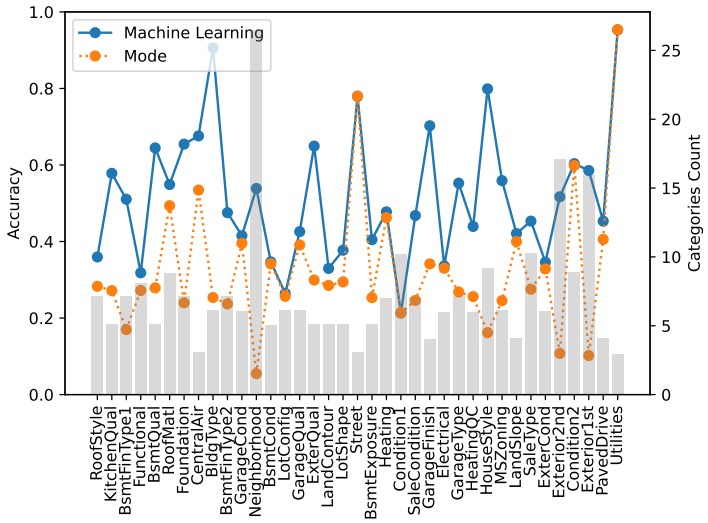

Figure 5.34: House dataset: Comparison of the Balanced Accuracy Score Between our Data Imputation Model and the Statistical Mode Method

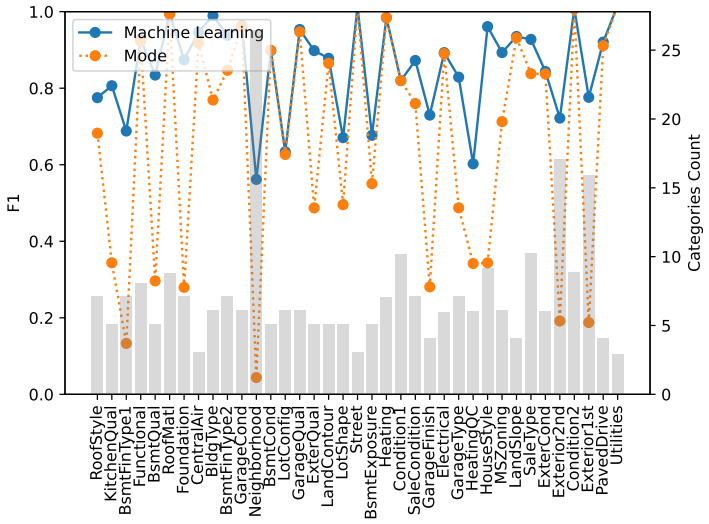

Figure 5.35: House dataset: Comparison of the F1 Score Between our Data Imputation Model and the Statistical Mode Method

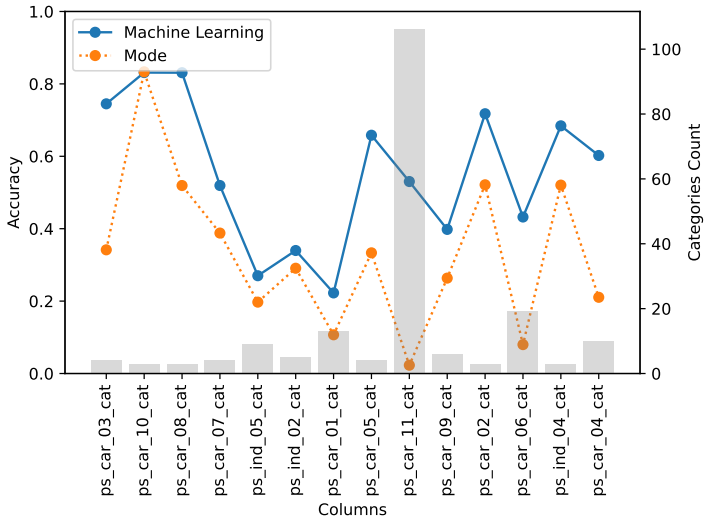

Figure 5.36: Insurance dataset: Comparison of the Balanced Accuracy Score Between our Data Imputation Model and the Statistical Mode Method

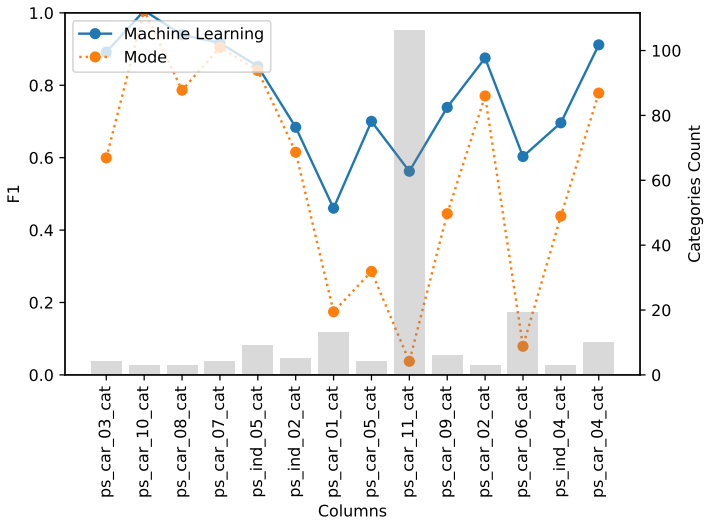

Figure 5.37: Insurance dataset: Comparison of the F1 Score Between our Data Imputation Model and the Statistical Mode Method

From the results, we can see that our method significantly outperforms the statistical mode method, regardless of the performance evaluation standard used. It's visually apparent from the graphs that the results obtained using our method (represented in the graphs as "Machine Learning" and shown with blue lines) provide better predictions than the mode method even in the worst-case scenario (represented by the orange dotted lines).

This is especially noticeable when dealing with features with many category labels. For instance, in Figure 5.34, the feature "Neighborhood" produces poor performance results when the mode method is used. The reason for this, as we have previously analyzed and as demonstrated in the figure, is that the feature contains a large number of category labels. This complicates decision-making in multi-classification models, making the use of the most frequent category label for imputation, as the mode method does, a poor strategy.

Similarly, in Figure 5.36, the feature "ps_car_11_cat" has over 100 category labels. Despite this complexity, our machine learning-based data imputation model can still maintain good results, while the mode-based imputation results are nearly zero.

Of course, we also have some features where both methods perform well, some even achieving a perfect 100% accuracy rate in imputation, such as the feature "ps_car_10_cat" in Figure 5.37. When analyzing the original values of these features, we found that the data distribution within these features is highly uneven. Either all values appear in one category, or the majority (over 95%) of the values are distributed in one category. Such uneven classification provides good prediction results for both data imputation methods.

Chapter 6

Conclusion

6.1 Answers to the Research Questions

In Chapter 1, we presented some of the current problems in the real estate field and the solutions proposed by KATE Innovations. We have formulated our main research questions as well as some sub-research questions. We now summarize the analysis of the questions we posed earlier.

Main Research Question (MRQ): Given a high-dimensional dataset with missing values (training data) and a set of target features (test data), what techniques can we use to predict the values of the target features accurately?

As analyzed in Chapter 4, our initial step involves the treatment of missing data. Generally, missing data can be addressed through machine learning-based methods and statistical approaches. In our previous experiments, we substantiated the superior performance of machine learning-based data imputation over statistical imputation methods, particularly when dealing with a significant number of class labels.

Additionally, we addressed feature selection for high-dimensional data. However, the decision to employ feature selection significantly depends on the inherent characteristics of the feature data itself. As such, we view this pre-processing step as an optional component within the modeling process, contingent upon the specific characteristics of the data. In our baseline approach, we employed the FCBF feature selection method proposed by Yu and Liu [112]. However, due to the nature of the dataset's features, utilizing only the minimal-optimal feature subset might not suffice to provide enough dimensionality of feature information. Consequently, within our study, we introduced enhancements to the FCBF algorithm. Employing this refined algorithm, we attained enhanced predictive performance outcomes.

In summary, when confronted with high-dimensional datasets with missing values, our methodology entails initial imputation of non-target features, followed by feature selection focusing on highly correlated attributes pertaining to the target feature, thereby reducing the data dimensionality. Through the implementation of these two preprocessing techniques, as validated by empirical charts, we affirm the attainment of remarkably accurate predictive results.

For the sub-questions:

1. How to deal with missing values in the training data.

   To address the presence of missing values within our training data, as outlined in the primary research question, we compared two methods: machine learning-based and statistical-based approaches. More specifically, we use the machine learning-based approach. This is because when dealing with categorical data types, we leverage the fact that the human decision-making process during data entry often aligns with the decision process of tree-based machine learning models. Consequently, for handling missing values, we utilize a tree-based model to emulate the human decision pathway and perform imputation.

However, in cases where a particular feature exhibits substantial missing data, we exercise caution and opt to omit that feature. The rationale behind this decision lies in the potential risk associated with attempting to recover missing data, as such endeavors could introduce significantly biased results into the dataset.

2. How to handle the high dimensional dataset in the training data.

   There are several approaches for dealing with high-dimensional data. Here we utilize the feature selection method. Normally, there are three main approaches: wrapper, filter, and embedded. We first use a filter method called Fast Correlation Based Filter (FCBF) as our baseline. Then we embedded a wrapper forward selection method with FCBF, and this new feature selection method showed a significant improvement compared to the baseline.

   However, whether to use feature selection for high-dimensional data also requires an analysis of the feature data itself. If the correlation of all features within the dataset itself is tight and all the features are important, then the results of our experiments consider the use of feature selection to be unnecessary, considering the additional time and space spent on selection and training. Therefore, we consider feature selection as an optional part of preprocessing.

3. What models can be used to predict the target features accurately?

   We incorporated the Random Forest, Decision Tree, and K-Nearest Neighbor algorithms into our bucket-like ensemble model to predict the target features. We chose these three models because they are versatile in handling both classification and regression tasks. Moreover, as discussed in our initial sub-research question, machine learning models based on training can effectively mimic the human decision-making process.

   Furthermore, in general, employing the Random Forest model tends to yield superior predictive outcomes. However, in scenarios where feature selection is applied within the model, the Decision Tree and K-Nearest Neighbor algorithms occasionally outperform Random Forest due to the reduction in feature dimensions and the properties of the Random Forest algorithm itself.

6.2 Summary

In the field of real estate assessment, there's a significant problem arising from the possibility that experts evaluating properties might accidentally introduce mistakes or biases while making judgments. These errors, whether they come from quickly entering data or personal influences, can later lead to unexpected financial and legal consequences. To address these problems, we establish a solution framework. This solution works by using the ability to predict important property characteristics. These predictions then act as cues to assist evaluators in their assessments. The resulting framework has several key components. First, there's a process that fills in any missing data in important features. Next, it uses the best available model to make predictions about the target. Additionally, there's an optional step for selecting features that are highly correlated to the target features, which are then used to make the prediction. Moreover, the solution employs a bucket-like ensemble model. This ensemble model is constructed using Decision Tree, Random Forests, and K-Nearest Neighbor models. This combination allows it to address both regression and classification problems, making it suitable for a wide range of general problem-solving tasks. In other words, this model can accurately fill in missing data and utilizes feature subsets to make more precise predictions when necessary. This is applicable to both regression and classification tasks, always employing the most suitable predictive model for each specific feature's situation. This combination of predictive capability and data reliability protects against potential mistakes, improving the accuracy and dependability of real estate assessments.

Through experiments, we have demonstrated the effectiveness of our data imputation model - Bucket4Imp - in accurately recovering missing data on a general scale. We achieved higher accuracy results in a multi-class scenario compared to the statistical-based imputation method. Additionally, employing model predictions after data imputation led to higher accuracy compared to making predictions directly using a dataset filled with missing values. Moreover, we believe that the practical approach to feature selection requires a careful examination of the feature set itself. In our experiments, we chose the FCBF algorithm as a baseline and improved it using the forward selection method. This involved progressively expanding the feature subset to identify the most suitable features. The improved feature selection algorithm notably enhanced predictive performance.

Our analysis emphasized the challenges in predicting multi-class features, particularly when these features were evenly distributed across different classes. We illustrated that performance was influenced by the number of class labels and the data distribution among these labels. We found that as the quantity of feature class labels increased and the data became more balanced, the overall data imputation and prediction performance decreased. In contrast, if the data distribution was imbalanced, prediction performance increased.

Additionally, we explored the relationship between the characteristics of the selected model and the dimension of the data features each time a computational model was chosen. We found that when feature selection was not applied, the Random Forest model generally outperformed the others due to its noise robustness and ability to handle complex feature relationships. Conversely, when feature selection was applied, simpler models like Decision Tree and KNN might outperform the Random Forest model as they were more capable of handling the reduced and refined feature set.

In conclusion, our study suggests that the choice of data pre-processing techniques should depend on the nature and complexity of the dataset, especially the usage of feature selection. While machine learning models like Random Forests are generally robust, simpler models like Decision Trees and KNNs can outperform them when the feature set is sufficiently refined through feature selection. Also, machine learning-based data imputation methods often provide superior performance over traditional statistical methods, especially for features with many category labels. Nonetheless, it's crucial to be mindful of the data distribution, as it can significantly influence prediction performance.

6.3 Limitation and Future Work

The constraints on the outcome quality of the prediction framework rely on various factors, encompassing the extent of missing values within the dataset, the chosen feature selection methodologies, the machine learning models integrated into the ensemble prediction model, and the distribution of multi-class data, in addition to the number of class labels.

Our future work will concentrate on enhancing performance by addressing the limitations mentioned above. We plan to achieve this by further exploring alternative feature selection methods and models. Given the multifaceted nature of real estate evaluations, we believe there are other suitable approaches that can make better use of feature selection methods. These approaches would involve narrowing down the target feature by leveraging all available features, while also improving computational complexity. Furthermore, substituting the current predictive model with more accurate models designed specifically for the prediction task could enhance overall predictive capabilities. We also aim to investigate the relationship between the percentage of missing data in the imputed dataset and the performance of feature selection using imputed data. We intend to conduct further research to explore these aspects and create better, more efficient solutions for preparing data in machine learning.

Bibliography

[1] *About the redfin estimate — home value estimator*. URL: https://www.redfin.com/redfin-estimate.

[2] Michy Alice. "Imputing missing data with R; MICE package". In: *Data Science Plus* (2015).

[3] Hussein Almuallim and Thomas G Dietterich. "Learning boolean concepts in the presence of many irrelevant features". In: *Artificial intelligence* 69.1-2 (1994), pages 279–305.

[4] Emre Arslan. *House Price Prediction*. https://www.kaggle.com/code/emrearslan123/house-price-prediction. 2021.

[5] Gabriel Morgan Asaftei, Sudeep Doshi, John Means, and Aditya Sanghvi. *Getting ahead of the market: How big data is transforming real estate*. Mar. 2021. URL: https://www.mckinsey.com/industries/real-estate/our-insights/getting-ahead-of-the-market-how-big-data-is-transforming-real-estate.

[6] Vincent Audigier, François Husson, and Julie Josse. "Multiple imputation for continuous variables using a Bayesian principal component analysis". In: *Journal of statistical computation and simulation* 86.11 (2016), pages 2140–2156.

[7] Gustavo EAPA Batista and Maria Carolina Monard. "An analysis of four missing data treatment methods for supervised learning". In: *Applied artificial intelligence* 17.5-6 (2003), pages 519–533.

[8] Yoshua Bengio and Francois Gingras. "Recurrent neural networks for missing or asynchronous data". In: *Advances in neural information processing systems* 8 (1995).

[9] Derrick A Bennett. "How can I deal with missing data in my study?" In: *Australian and New Zealand journal of public health* 25.5 (2001), pages 464–469.

[10] Gérard Biau and Erwan Scornet. "A random forest guided tour". In: *Test* 25 (2016), pages 197–227.

[11] Avrim L Blum and Pat Langley. "Selection of relevant features and examples in machine learning". In: *Artificial intelligence* 97.1-2 (1997), pages 245–271.

[12] Leo Breiman. "Bagging predictors". In: *Machine learning* 24.2 (1996), pages 123–140.

[13] Leo Breiman. "Random forests". In: *Machine learning* 45 (2001), pages 5–32.

[14] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[15] Zhipeng Cai, Maysam Heydari, and Guohui Lin. "Iterated local least squares microarray missing value imputation". In: *Journal of bioinformatics and computational biology* 4.05 (2006), pages 935–957.

[16] Anthony Cavin. *6 Ways to Encode Features for Machine Learning Algorithms*. 2021. URL: https://towardsdatascience.com/6-ways-to-encode-features-for-machine-learning-algorithms-21593f6238b0 (visited on 07/04/2023).

[17] Kankawee Chanasit, Ekapol Chuangsuwanich, Atiwong Suchato, and Proadpran Punyabukkana. "A real estate valuation model using boosted feature selection". In: *IEEE Access* 9 (2021), pages 86938–86953.

[18] Girish Chandrashekar and Ferat Sahin. “A survey on feature selection methods”. In: *Computers & Electrical Engineering* 40.1 (2014), pages 16–28.

[19] Rung-Ching Chen, Christine Dewi, Su-Wen Huang, and Rezzy Eko Caraka. “Selecting critical features for data classification based on machine learning methods”. In: *Journal of Big Data* 7.1 (2020), pages 1–26.

[20] An-Lin Cheng, Haiqun Lin, Wesley Kasprow, and Robert A Rosenheck. “Impact of supported housing on clinical outcomes analysis of a randomized trial using multiple imputation technique”. In: *The Journal of nervous and mental disease* 195.1 (2007), page 83.

[21] A Christobel and P SivaPrakasam. “The negative impact of missing value imputation in classification of diabetes dataset and solution for improvement”. In: *IOSR J. Comput. Eng.(IOSRJCE)* 7.5 (2012).

[22] Sait Yücebaş, Melike Doğan, and Levent Genç. “A C4.5 – CART DECISION TREE MODEL FOR REAL ESTATE PRICE PREDICTION AND THE ANALYSIS OF THE UNDERLYING FEATURES”. In: *Konya Journal of Engineering Sciences* 10.1 (2022), pages 147–161. DOI: 10.36306/konjes.1013833.

[23] Julien Clavel, Gildas Merceron, and Gilles Escarguel. “Missing data estimation in morphometrics: how much is too much?” In: *Systematic biology* 63.2 (2014), pages 203–218.

[24] Chris Ding and Hanchuan Peng. “Minimum redundancy feature selection from microarray gene expression data”. In: *Journal of bioinformatics and computational biology* 3.02 (2005), pages 185–205.

[25] Yiran Dong and Chao-Ying Joanne Peng. “Principled missing data methods for researchers”. In: *SpringerPlus* 2 (2013), pages 1–17.

[26] Saso Džeroski and Bernard Ženko. “Is combining classifiers with stacking better than selecting the best one?” In: *Machine learning* 54 (2004), pages 255–273.

[27] Craig K Enders. *Applied missing data analysis*. Guilford Publications, 2022.

[28] David Feldman and Shulamith Gross. “Mortgage default: classification trees analysis”. In: *The Journal of Real Estate Finance and Economics* 30.4 (2005), pages 369–396.

[29] Françoise Fessant and Sophie Midenet. “Self-organising map for data imputation and correction in surveys”. In: *Neural Computing & Applications* 10.4 (2002), pages 300–310.

[30] Evelyn Fix and Joseph Lawson Hodges. “Discriminatory analysis. Nonparametric discrimination: Consistency properties”. In: *International Statistical Review/Revue Internationale de Statistique* 57.3 (1989), pages 238–247.

[31] George Forman et al. “An extensive empirical study of feature selection metrics for text classification.” In: *J. Mach. Learn. Res.* 3.Mar (2003), pages 1289–1305.

[32] Pedro J Garcıéa-Laencina, José-Luis Sancho-Gómez, and Anıébal R Figueiras-Vidal. “Pattern classification with missing data: a review”. In: *Neural Computing and Applications* 19.2 (2010), pages 263–282.

[33] Sebastian Gnat. “Impact of categorical variables encoding on property mass valuation”. In: *Procedia Computer Science* 192 (2021), pages 3542–3550.

[34] Magdalena Graczyk, Tadeusz Lasota, Bogdan Trawiński, and Krzysztof Trawiński. “Comparison of Bagging, Boosting and Stacking Ensembles Applied to Real Estate Appraisal”. In: *Intelligent Information and Database Systems*. Edited by Ngoc Thanh Nguyen, Manh Thanh Le, and Jerzy Świątek. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pages 340–350. ISBN: 978-3-642-12101-2.

[35] Mark A Hall. “Correlation-based feature selection of discrete and numeric class machine learning”. In: (2000).

[36] Md Kamrul Hasan, Md Ashraful Alam, Dola Das, Eklas Hossain, and Mahmudul Hasan. “Diabetes prediction using ensembling of different machine learning classifiers”. In: *IEEE Access* 8 (2020), pages 76516–76531.

[37] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Volume 2. Springer, 2009.

[38] Hiroki Horino, Hirofumi Nonaka, Elisa Claire Alemán Carreón, and Toru Hiraoka. "Development of an entropy-based feature selection method and analysis of online reviews on real estate". In: *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*. IEEE. 2017, pages 2351–2355.

[39] Sebastian Jäger, Arndt Allhorn, and Felix Bießmann. "A benchmark for data imputation methods". In: *Frontiers in big Data* 4 (2021), page 693674.

[40] Mortaza Jamshidian and Peter M Bentler. "ML estimation of mean and covariance structures with missing data using complete data routines". In: *Journal of Educational and behavioral Statistics* 24.1 (1999), pages 21–24.

[41] Mortaza Jamshidian and Matthew Mata. "Advances in analysis of mean and covariance structure when data are incomplete". In: *Handbook of latent variable and related models*. Elsevier, 2007, pages 21–44.

[42] Ilyes Jenhani, Nahla Ben Amor, and Zied Elouedi. "Decision trees as possibilistic classifiers". In: *International journal of approximate reasoning* 48.3 (2008), pages 784–807.

[43] José M Jerez, Ignacio Molina, Pedro J Garcıéa-Laencina, Emilio Alba, Nuria Ribelles, Miguel Martién, and Leonardo Franco. "Missing data imputation using statistical and machine learning methods in a real breast cancer problem". In: *Artificial intelligence in medicine* 50.2 (2010), pages 105–115.

[44] José M Jerez, Ignacio Molina, José L Subirats, and Leonardo Franco. "Missing data imputation in breast cancer prognosis". In: *Survival* 8.9 (2006), pages 10–11.

[45] George H John, Ron Kohavi, and Karl Pfleger. "Irrelevant features and the subset selection problem". In: *Machine learning proceedings 1994*. Elsevier, 1994, pages 121–129.

[46] *Kaggle*. https://www.kaggle.com. Accessed on July 10, 2023.

[47] Kaggle. *Porto Seguro's Safe Driver Prediction Dataset*. https://www.kaggle.com/competitions/porto-seguro-safe-driver-prediction/data?select=test.csv. Accessed: July 10, 2023. 2023.

[48] *Kaggle dataset - Porto seguro safe driver prediction*. URL: https://www.kaggle.com/competitions/porto-seguro-safe-driver-prediction/data?select=test.csv.

[49] J Pradeep Kandhasamy and SJPCS Balamurali. "Performance analysis of classifier models to predict diabetes mellitus". In: *Procedia Computer Science* 47 (2015), pages 45–51.

[50] Hyunsoo Kim, Gene H Golub, and Haesun Park. "Missing value estimation for DNA microarray gene expression data: local least squares imputation". In: *Bioinformatics* 21.2 (2005), pages 187–198.

[51] Kenji Kira and Larry A Rendell. "A practical approach to feature selection". In: *Machine learning proceedings 1992*. Elsevier, 1992, pages 249–256.

[52] Kenji Kira and Larry A Rendell. "The feature selection problem: Traditional methods and a new algorithm". In: *Proceedings of the tenth national conference on Artificial intelligence*. 1992, pages 129–134.

[53] Amit Kishor and Chinmay Chakraborty. "Early and accurate prediction of diabetics based on FCBF feature selection and SMOTE". In: *International Journal of System Assurance Engineering and Management* (2021), pages 1–9.

[54] Ron Kohavi and George H John. "Wrappers for feature subset selection". In: *Artificial intelligence* 97.1-2 (1997), pages 273–324.

[55] Daphne Koller, Mehran Sahami, et al. "Toward optimal feature selection". In: *ICML*. Volume 96. 28. 1996, page 292.

[56] Igor Kononenko, Edvard Šimec, and Marko Robnik-Šikonja. "Overcoming the myopia of inductive learning algorithms with RELIEFF". In: *Applied Intelligence* 7.1 (1997), pages 39–55.

[57] Nishoak Kosaraju, Sainath Reddy Sankepally, and K Mallikharjuna Rao. "Categorical Data: Need, Encoding, Selection of Encoding Method and Its Emergence in Machine Learning Models—A Practical Review Study on Heart Disease Prediction Dataset Using Pearson Correlation". In: *Proceedings of International Conference on Data Science and Applications: ICDSA 2022, Volume 1*. Springer. 2023, pages 369–382.

[58] Susanna Levantesi and Gabriella Piscopo. "The importance of economic variables on London real estate market: A random forest approach". In: *Risks* 8.4 (2020), page 112.

[59] Peng Li, Elizabeth A Stuart, and David B Allison. "Multiple imputation: a flexible tool for handling missing data". In: *Jama* 314.18 (2015), pages 1966–1967.

[60] Y Lin and Y Jeon. *Random forests and adaptive nearest neighbors (Technical report)*. Technical report. Technical Report, 2002.

[61] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*. Volume 793. John Wiley & Sons, 2019.

[62] Huawen Liu, Jigui Sun, Lei Liu, and Huijie Zhang. "Feature selection with dynamic mutual information". In: *Pattern Recognition* 42.7 (2009), pages 1330–1339.

[63] Marco Locurcio, Francesco Tajani, and Pierluigi Morano. *Computational Methods Applied to Data Analysis for Modeling Complex Real Estate Systems*. 2020.

[64] Sifei Lu, Zengxiang Li, Zheng Qin, Xulei Yang, and Rick Siow Mong Goh. "A hybrid regression technique for house prices prediction". In: *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*. 2017, pages 319–323. DOI: 10.1109/IEEM.2017.8289904.

[65] Christina Mack, Zhaohui Su, and Daniel Westreich. "Managing missing data in patient registries: addendum to registries for evaluating patient outcomes: a user's guide". In: (2018).

[66] Paul Madley-Dowd, Rachael Hughes, Kate Tilling, and Jon Heron. "The proportion of missing data should not be used to guide decisions on multiple imputation". In: *Journal of Clinical Epidemiology* 110 (2019), pages 63–73. ISSN: 0895-4356. DOI: https://doi.org/10.1016/j.jclinepi.2019.02.016. URL: https://www.sciencedirect.com/science/article/pii/S0895435618308710.

[67] Anastasios G Malliaris and Mary Malliaris. "What drives gold returns? A decision tree analysis". In: *Finance Research Letters* 13 (2015), pages 45–53.

[68] Md Maniruzzaman, Md Jahanur Rahman, Md Al-MehediHasan, Harman S Suri, Md Menhazul Abedin, Ayman El-Baz, and Jasjit S Suri. "Accurate diabetes risk stratification using machine learning: role of missing value and outliers". In: *Journal of medical systems* 42 (2018), pages 1–17.

[69] Tshilidzi Marwala and S Chakraverty. "Fault classification in structures with incomplete measured data using autoassociative neural networks and genetic algorithm". In: *Current Science* (2006), pages 542–548.

[70] Daniel McNeish. "Missing data methods for arbitrary missingness with small samples". In: *Journal of Applied Statistics* 44.1 (2017), pages 24–39.

[71] Sukhdev Mishra and Diwakar Khare. "On comparative performance of multiple imputation methods for moderate to large proportions of missing data in clinical trials: a simulation study". In: *J Med Stat Inform* 2.1 (2014), page 9.

[72] Thuraiya Mohd, Nur Syafiqah Jamil, Noraini Johari, Lizawati Abdullah, and Suraya Masrom. "An overview of real estate modelling techniques for house price prediction". In: *Charting a Sustainable Future of ASEAN in Business and Social Sciences: Proceedings of the 3 International Conference on the Future of ASEAN (ICoFA) 2019—Volume 1*. Springer. 2020, pages 321–338.

[73] Yar Muhammad, Muhammad Tahir, Maqsood Hayat, and Kil To Chong. "Early and accurate detection and diagnosis of heart disease using intelligent computational model". In: *Scientific reports* 10.1 (2020), page 19747.

[74] Muhammad Fahmi Mukhlishin, Ragil Saputra, and Adi Wibowo. "Predicting house sale price using fuzzy logic, Artificial Neural Network and K-Nearest Neighbor". In: *2017 1st International Conference on Informatics and Computational Sciences (ICICoS)*. IEEE. 2017, pages 171–176.

[75] Arundhati Navada, Aamir Nizam Ansari, Siddharth Patil, and Balwant A Sonkamble. "Overview of use of decision tree algorithms in machine learning". In: *2011 IEEE control and system graduate research colloquium*. IEEE. 2011, pages 37–42.

[76] Sanaz Nikfalazar, Chung-Hsing Yeh, Susan Bedingfield, and Hadi A Khorshidi. "Missing data imputation using decision trees and fuzzy clustering with iterative learning". In: *Knowledge and Information Systems* 62 (2020), pages 2419–2437.

[77] Shigeyuki Oba, Masa-aki Sato, Ichiro Takemasa, Morito Monden, Ken-ichi Matsubara, and Shin Ishii. "A Bayesian missing value estimation method for gene expression profile data". In: *Bioinformatics* 19.16 (2003), pages 2088–2096.

[78] Worapat Paireekreng and Worawat Choensawat. "An Ensemble Learning Based Model for Real Estate Project Classification". In: *Procedia Manufacturing* 3 (2015). 6th International Conference on Applied Human Factors and Ergonomics (AHFE 2015) and the Affiliated Conferences, AHFE 2015, pages 3852–3859. ISSN: 2351-9789. DOI: https://doi.org/10.1016/j.promfg.2015.07.892. URL: https://www.sciencedirect.com/science/article/pii/S2351978915008938.

[79] Adam Pantanowitz and Tshilidzi Marwala. "Missing data imputation through the use of the random forest algorithm". In: *Advances in computational intelligence*. Springer. 2009, pages 53–62.

[80] Marie-Therese Puth, Markus Neuhäuser, and Graeme D Ruxton. "Effective use of Pearson's product–moment correlation coefficient". In: *Animal behaviour* 93 (2014), pages 183–189.

[81] Md Geaur Rahman and Md Zahidul Islam. "Missing value imputation using decision trees and decision forests by splitting and merging records: Two novel techniques". In: *Knowledge-Based Systems* 53 (2013), pages 51–65.

[82] Soumya Raychaudhuri, Joshua M Stuart, and Russ B Altman. "Principal components analysis to summarize microarray experiments: application to sporulation time series". In: *Biocomputing 2000*. World Scientific, 1999, pages 455–466.

[83] *RENTestimate Discover the rental value of any residential property*. URL: https://www.homeunion.com/rentestimate/.

[84] Achmad Ridok, Wayan Firdaus Mahmudy, and Muhaimin Rifai. "An improved artificial immune recognition system with fast correlation based filter (FCBF) for feature selection". In: *2017 Fourth International Conference on Image Information Processing (ICIIP)*. IEEE. 2017, pages 1–6.

[85] Marko Robnik-Šikonja and Igor Kononenko. "An adaptation of Relief for attribute estimation in regression". In: *Machine learning: Proceedings of the fourteenth international conference (ICML'97)*. Volume 5. 1997, pages 296–304.

[86] Donald B Rubin. "Inference and missing data". In: *Biometrika* 63.3 (1976), pages 581–592.

[87] Karshiev Sanjar, Olimov Bekhzod, Jaesoo Kim, Anand Paul, and Jeonghong Kim. "Missing data imputation for geolocation-based price prediction using KNN–MCF method". In: *ISPRS International Journal of Geo-Information* 9.4 (2020), page 227.

[88] Rushab Sawant, Yashwant Jangid, Tushar Tiwari, Saurabh Jain, and Ankita Gupta. "Comprehensive analysis of housing price prediction in pune using multi-featured random forest approach". In: *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*. IEEE. 2018, pages 1–5.

[89] Joseph L Schafer. "Multiple imputation: a primer". In: *Statistical methods in medical research* 8.1 (1999), pages 3–15.

[90] Joseph L Schafer and John W Graham. "Missing data: our view of the state of the art." In: *Psychological methods* 7.2 (2002), page 147.

[91] Joseph L Schafer and Maren K Olsen. "Multiple imputation for multivariate missing-data problems: A data analyst's perspective". In: *Multivariate behavioral research* 33.4 (1998), pages 545–571.

[92] Baris Senliol, Gokhan Gulgezen, Lei Yu, and Zehra Cataltepe. "Fast Correlation Based Filter (FCBF) with a different search strategy". In: *2008 23rd international symposium on computer and information sciences*. IEEE. 2008, pages 1–4.

[93] *Set Cover Problem*. URL: https://en.wikipedia.org/wiki/Set_cover_problem.

[94] Blake Shaw and Tony Jebara. "Structure preserving embedding". In: *Proceedings of the 26th Annual International Conference on Machine Learning*. 2009, pages 937–944.

[95] Ali Soltani, Mohammad Heydari, Fatemeh Aghaei, and Christopher James Pettit. "Housing price prediction incorporating spatio-temporal dependency into machine learning algorithms". In: *Cities* 131 (2022), page 103941. ISSN: 0264-2751. DOI: https://doi.org/10.1016/j.cities.2022.103941. URL: https://www.sciencedirect.com/science/article/pii/S0264275122003808.

[96] Matthias Studer, Gilbert Ritschard, Alexis Gabadinho, and Nicolas S Müller. "Discrepancy analysis of state sequences". In: *Sociological methods & research* 40.3 (2011), pages 471–510.

[97] Mingxu Sun, Xiaodong Liu, and Scholas Mbonihankuye. "Analysis of the efficiency-energy with regression and classification in household using K-NN". In: *Artificial Intelligence and Security: 5th International Conference, ICAIS 2019, New York, NY, USA, July 26-28, 2019, Proceedings, Part II 5*. Springer. 2019, pages 358–368.

[98] Fei Tang and Hemant Ishwaran. "Random forest missing data algorithms". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 10.6 (2017), pages 363–377.

[99] Tressy Thomas and Enayat Rajabi. "A systematic review of machine learning-based missing value imputation techniques". In: *Data Technologies and Applications* 55.4 (2021), pages 558–585.

[100] Nicholas J Tierney, Fiona A Harden, Maurice J Harden, and Kerrie L Mengersen. "Using decision trees to understand structure in missing data". In: *BMJ open* 5.6 (2015), e007450.

[101] *UCI dataset - The Insurance Company (TIC) Benchmark*. URL: https://www.kaggle.com/datasets/kushshah95/the-insurance-company-tic-benchmark?resource=download.

[102] H Cevallos Valdiviezo and Stefan Van Aelst. "Tree-based prediction on incomplete data using imputation or surrogate decisions". In: *Information Sciences* 311 (2015), pages 163–181.

[103] Stef Van Buuren and Karin Groothuis-Oudshoorn. "mice: Multivariate imputation by chained equations in R". In: *Journal of statistical software* 45 (2011), pages 1–67.

[104] Tim Verdonck, Bart Baesens, Maríéa Óskarsdóttir, et al. "Special issue on feature engineering editorial". In: *Machine Learning* (2021), pages 1–12.

[105] Marc K Walton. "Addressing and advancing the problem of missing data". In: *Journal of Biopharmaceutical Statistics* 19.6 (2009), pages 945–956.

[106] Huimin Wang, Jianxiang Tang, Mengyao Wu, Xiaoyu Wang, and Tao Zhang. "Application of machine learning missing data imputation techniques in clinical decision making: taking the discharge assessment of patients with spontaneous supratentorial intracerebral hemorrhage as an example". In: *BMC Medical Informatics and Decision Making* 22.1 (2022), pages 1–14.

[107] Qian Wang, Weijia Cao, Jiawei Guo, Jiadong Ren, Yongqiang Cheng, and Darryl N Davis. "DMP_MI: an effective diabetes mellitus classification algorithm on imbalanced data with missing values". In: *IEEE access* 7 (2019), pages 102232–102238.

[108] Florian Wetschoreck. *RIP correlation. introducing the predictive power score*. May 2020. URL: https://towardsdatascience.com/rip-correlation-introducing-the-predictive-power-score-3d90808b9598.

[109] *What is a Zestimate? Zillow's Zestimate Accuracy*. Dec. 2022. URL: https://www.zillow.com/z/zestimate/.

[110] Sukran Yalpir, Suleyman Sisman, Ali Utku Akar, and Fatma Bunyan Unel. "Feature selection applications and model validation for mass real estate valuation systems". In: *Land use policy* 108 (2021), page 105539.

[111] Lei Yu and Huan Liu. "Efficient feature selection via analysis of relevance and redundancy". In: *The Journal of Machine Learning Research* 5 (2004), pages 1205–1224.

[112] Lei Yu and Huan Liu. "Feature selection for high-dimensional data: A fast correlation-based filter solution". In: *Proceedings of the 20th international conference on machine learning (ICML-03)*. 2003, pages 856–863.

[113] Xinmeng Zhang, Chao Yan, Cheng Gao, Bradley A Malin, and You Chen. "Predicting missing values in medical data via XGBoost regression". In: *Journal of healthcare informatics research* 4.4 (2020), pages 383–394.

[114] Zhishuo Zhang. "Decision Trees for Objective House Price Prediction". In: *2021 3rd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*. IEEE. 2021, pages 280–283.

[115] Zhenyu Zhao, Radhika Anand, and Mallory Wang. "Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform". In: *2019 IEEE international conference on data science and advanced analytics (DSAA)*. IEEE. 2019, pages 442–452.

Appendix A

Appendix

A.1 House Dataset and Insurance Dataset

Below are the results of the impact on data prediction with and without feature selection after data imputation, as well as the results of direct data prediction. Similar to the KATE dataset, we employed the same model and performance calculation metrics to observe the outcomes.

Table A.1: House Dataset: Predictive Performance Using 65% Data for Training (Data Imputation and Data Prediction)

Feature Name	Performance on Validation Set (Balanced Accuracy)				Performance on Test Set (Using Best Model)	
	DT	RF	KNN	Best Model	Balanced Accuracy	F1 Score
LotShape	0.6439	0.6976	0.6049	RF	0.3516	0.6681
LotConfig	0.6049	0.6829	0.6293	RF	0.2025	0.5996
Neighborhood	0.4976	0.6000	0.2146	RF	0.5380	0.6135
BldgType	0.9610	0.9512	0.8585	DT	0.9350	0.9901
HouseStyle	0.9415	0.9366	0.5220	DT	0.7713	0.9364
MasVnrType	0.8634	0.9171	0.4488	RF	0.6165	0.8816
BsmtExposure	0.6585	0.7366	0.6390	RF	0.4299	0.6687
GarageFinish	0.6244	0.7073	0.5366	RF	0.6752	0.6976
SaleType	0.9220	0.9659	0.9220	RF	0.2775	0.9114

A.2 UCI Dataset

Below is the target feature's data distribution from the UCI dataset and the presence of missing data for each feature in the newly generated dataset with missing values.

Table A.2: Insurance Dataset: Predictive Performance Using 65% Data for Training (Data Imputation and Data Prediction)

Feature Name	Performance on Validation Set (Balanced Accuracy)				Performance on Test Set (Using Best Model)	
	DT	RF	KNN	Best Model	Balanced Accuracy	F1 Score
ps_ind_16_bin	0.8724	0.9235	0.5663	RF	0.8686	0.9173
ps_calc_16_bin	0.5	0.6276	0.5459	RF	0.5083	0.5284
ps_calc_10	0.1276	0.1786	0.0765	RF	0.0672	0.1071
ps_ind_03	0.1633	0.1786	0.1276	RF	0.1113	0.1370
ps_calc_06	0.2653	0.2398	0.1735	DT	0.1593	0.2268
ps_car_01_cat	0.3979	0.4694	0.2041	RF	0.1720	0.4323
ps_ind_09_bin	0.9745	0.9235	0.8112	DT	0.9742	0.9872
ps_car_07_cat	0.8929	0.9490	0.9439	RF	0.5	0.9356
ps_ind_02_cat	0.6786	0.7704	0.6837	RF	0.2812	0.6904

Table A.3: House Dataset: Predictive Performance Using 65% Data for Training (Data Imputation, Feature Selection, and Data Prediction)

Feature Name	Performance on Validation Set (Balanced Accuracy)				Performance on Test Set (Using Best Model)	
	DT	RF	KNN	Best Model	Balanced Accuracy	F1 Score
LotShape	0.5561	0.5561	0.5707	KNN	0.2430	0.5756
LotConfig	0.5659	0.5756	0.7024	KNN	0.2545	0.5949
Neighborhood	0.0878	0.1073	0.0878	RF	0.0672	0.1085
BldgType	0.9805	0.9707	0.9707	DT	0.9043	0.9709
HouseStyle	0.9659	0.9659	0.9659	DT	0.7221	0.9353
MasVnrType	0.8488	0.8585	0.8976	KNN	0.4931	0.8377
BsmtExposure	0.4878	0.4829	0.5659	KNN	0.2549	0.5041
GarageFinish	0.3415	0.4439	0.3366	RF	0.3977	0.4276
SaleType	0.9512	0.9512	0.9463	DT	0.3564	0.9416

Table A.4: Insurance Dataset: Predictive Performance Using 65% Data for Training (Data Imputation, Feature Selection, and Data Prediction)

Feature Name	Performance on Validation Set (Balanced Accuracy)				Performance on Test Set (Using Best Model)	
	DT	RF	KNN	Best Model	Balanced Accuracy	F1 Score
ps_ind_16_bin	0.7857	0.7959	0.7602	RF	0.7372	0.8081
ps_calc_16_bin	0.4847	0.5	0.5510	KNN	0.5231	0.5705
ps_calc_10	0.0918	0.1173	0.1020	RF	0.0545	0.0971
ps_ind_03	0.1122	0.0765	0.1020	DT	0.0629	0.0833
ps_calc_06	0.1990	0.25	0.2143	RF	0.1153	0.1753
ps_car_01_cat	0.2296	0.2806	0.2806	RF	0.0719	0.2293
ps_ind_09_bin	0.8112	0.8265	0.8163	RF	0.5691	0.7571
ps_car_07_cat	0.9031	0.9337	0.9439	KNN	0.5000	0.9356
ps_ind_02_cat	0.6327	0.6071	0.6939	KNN	0.2437	0.6170

Table A.5: House Dataset: Predictive Performance Using 65% Data for Training (Direct Data Prediction)

Feature Name	Performance on Validation Set (Balanced Accuracy)				Performance on Test Set (Using Best Model)	
	DT	RF	KNN	Best Model	Balanced Accuracy	F1 Score
LotShape	0.7073	0.7512	0.6976	RF	0.6481	0.7756
LotConfig	0.7073	0.7366	0.7122	RF	0.3270	0.6242
Neighborhood	0.5219	0.6439	0.2341	RF	0.5385	0.5849
BldgType	0.9561	0.9463	0.8244	DT	0.9127	0.9722
HouseStyle	0.9366	0.9024	0.5512	DT	0.7600	0.9394
MasVnrType	0.9073	0.9268	0.6439	RF	0.9009	0.9575
BsmtExposure	0.6195	0.7317	0.6634	RF	0.3768	0.6944
GarageFinish	0.6390	0.7415	0.5707	RF	0.7642	0.8465
SaleType	0.9366	0.9610	0.8537	RF	0.4959	0.9175

Table A.6: Insurance Dataset: Predictive Performance Using 65% Data for Training (Direct Data Prediction)

Feature Name	Performance on Validation Set (Balanced Accuracy)				Performance on Test Set (Using Best Model)	
	DT	RF	KNN	Best Model	Balanced Accuracy	F1 Score
ps_ind_16_bin	0.8791	0.9341	0.5824	RF	0.9038	0.9340
ps_calc_16_bin	0.5824	0.5549	0.5824	KNN	0.5286	0.5711
ps_calc_10	0.0989	0.0934	0.0989	DT	0.1196	0.1249
ps_ind_03	0.2308	0.1868	0.1374	DT	0.1364	0.1335
ps_calc_06	0.1099	0.2747	0.2143	RF	0.1433	0.1388
ps_car_01_cat	0.3242	0.4780	0.2692	RF	0.2657	0.4992
ps_ind_09_bin	1.0000	0.9615	0.7802	DT	1.0000	1.0000
ps_car_07_cat	0.9011	0.9670	0.9670	RF	0.5000	0.9049
ps_ind_02_cat	0.6374	0.7198	0.7033	RF	0.2771	0.6710

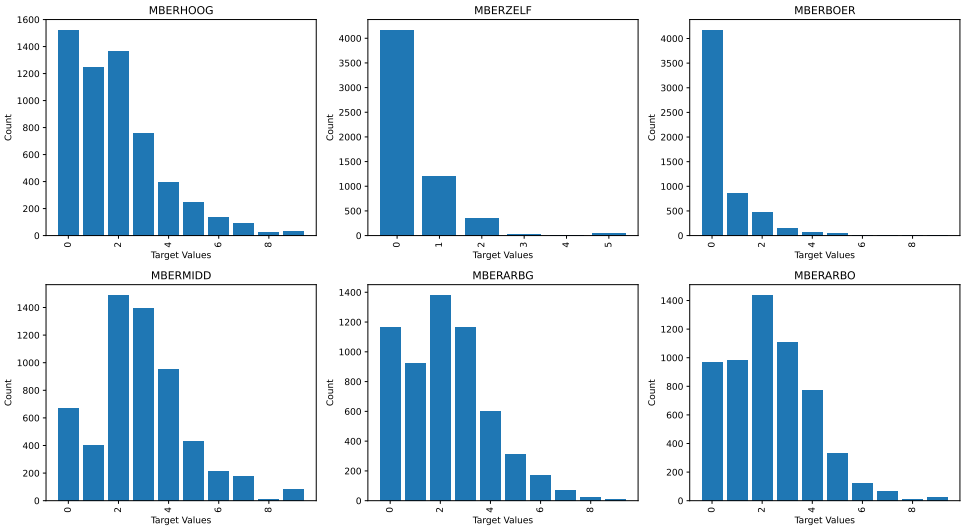

Figure A.1: UCI dataset: Target Feature's Data Distribution

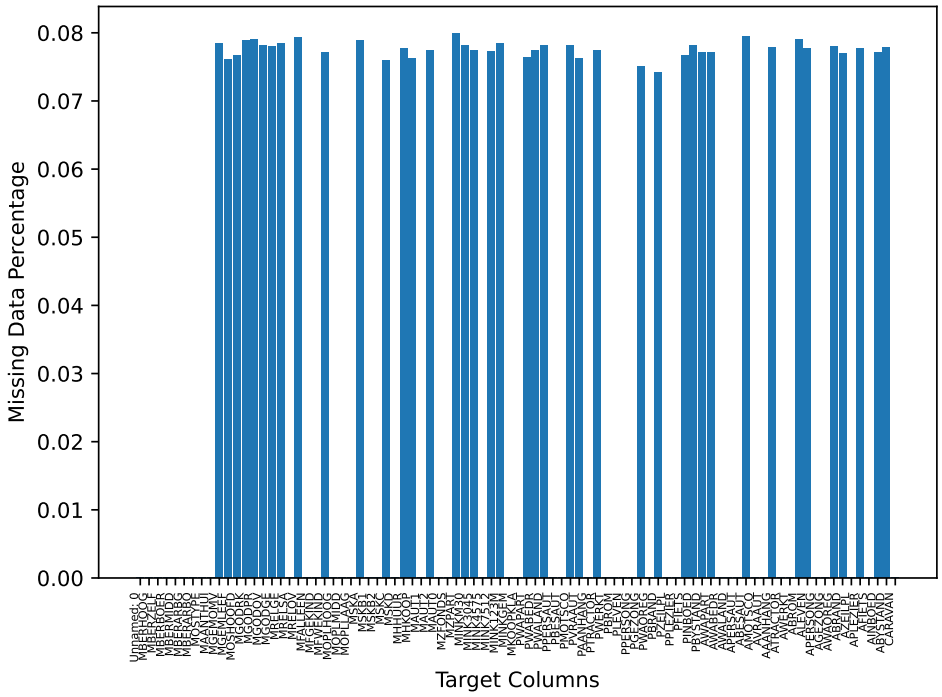

Figure A.2: UCI dataset: New Dataset with Generated Missing Situation