Home Credit Default Prediction

Hampus Fink Gärdström*

December 31, 2024

Abstract

The prediction of credit defaults is a significant challenge in the financial sector, where accurate forecasting can reduce risks for lenders. This study investigates the performance of several machine learning models—Logistic Regression, Random Forest, K-Nearest Neighbors, and XGBoost—on the Home Credit Default Risk dataset, a large-scale dataset containing 300,000+ loan records and 121 features. The study also explores the effects of dimensionality reduction techniques, specifically Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA), on model performance. The evaluation focuses on classification accuracy, precision, recall, and the ability to identify defaulted loans, with the aim of determining the best model and assessing the impact of feature selection and reduction on model outcomes.

Keywords: *Credit default prediction*, *machine learning models*, *dimensionality reduction*, *RFE*, *PCA*, *class imbalance*

*Department of Computer Science and Engineering, University of Southern Denmark. Email: hampus.cs13@nycu.edu.tw

1 Introduction

Predicting loan defaults is a critical task for financial institutions to mitigate risks associated with credit lending. With the increasing availability of large datasets containing various financial and demographic attributes, machine learning techniques have become a popular approach to address this problem. However, challenges such as class imbalance, high-dimensional data, and missing values complicate model training and evaluation. This study examines the impact of dimensionality reduction methods and model selection on the performance of machine learning models for loan default prediction, using the Home Credit Default Risk dataset.

The dataset consists of over 300,000 entries and 121 features that describe various aspects of loan applicants. One of the prominent issues within this dataset is the class imbalance, where a significant proportion of loans do not result in defaults. This imbalance can lead to biased model predictions, as machine learning algorithms may prioritize the majority class, resulting in lower sensitivity to the minority class (defaulted loans). To address this, the study compares several machine learning models, including Logistic Regression, Random Forest, K-Nearest Neighbors, and XGBoost, to determine the model that performs best for this prediction task.

In addition, the study explores the effect of dimensionality reduction techniques, specifically Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA), on model performance. RFE is used to iteratively remove less important features, while PCA reduces the dimensionality of the feature set by transforming it into a smaller number of uncorrelated components. These techniques aim to address issues such as multicollinearity, overfitting, and computational efficiency. The models are evaluated based on metrics such as classification accuracy, precision, recall, and their ability to identify the minority class.

2 Data

The data used in this study is from the Home Credit Default dataset, accessible on the Kaggle platform[1]. This dataset comprises extensive records related to credit loans, with a single dependent variable, representing a binary classification to indicate whether a loan was defaulted, and 121 independent variables. The dataset contains approximately 300,000 entries.

Upon examining the distribution of the target variable, significant class imbalance is evident in fig. 1. Specifically, only around 8% of the total loans in the dataset were ultimately defaulted. This imbalance introduces several methodological issues that need to be considered. Firstly, class imbalance can bias model performance, as standard machine learning algorithms may inherently favor the majority class, resulting in higher accuracy but reduced sensitivity to the minority class, which, in this case, is the defaulted loans. Addressing this issue may require methods such as resampling, class weighting, or specialized algorithms designed to manage imbalanced data distributions.

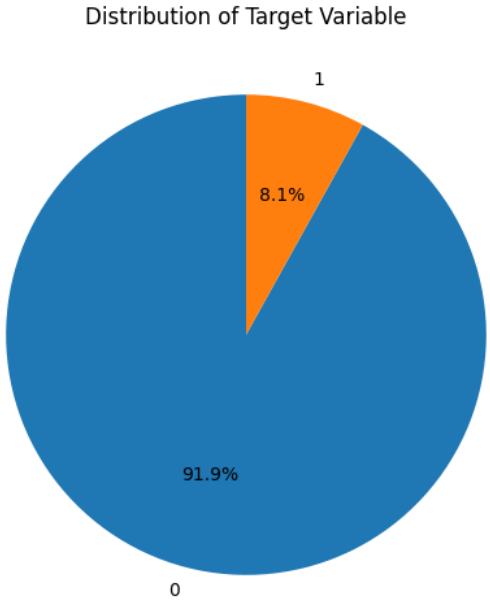

Figure 1: Distribution of the target variable showing class imbalance

[1] https://www.kaggle.com/competitions/home-credit-default-risk

With the high number of features, a substantial portion of the dataset exhibits missing values. In particular, less than 3% of the entries include values for all features, indicating significant sparsity. Although most of the features appear to have no missing values, certain features have missing data in a considerable portion of the records, as shown in Fig. 2. This uneven distribution of missing values poses challenges for feature engineering and modeling, as imputation methods may be required to address these gaps in the data, potentially introducing bias or noise into the model. Additionally, the high rate of missing data in specific features could indicate either data collection limitations or inherent dependencies within certain variables that impact their availability.

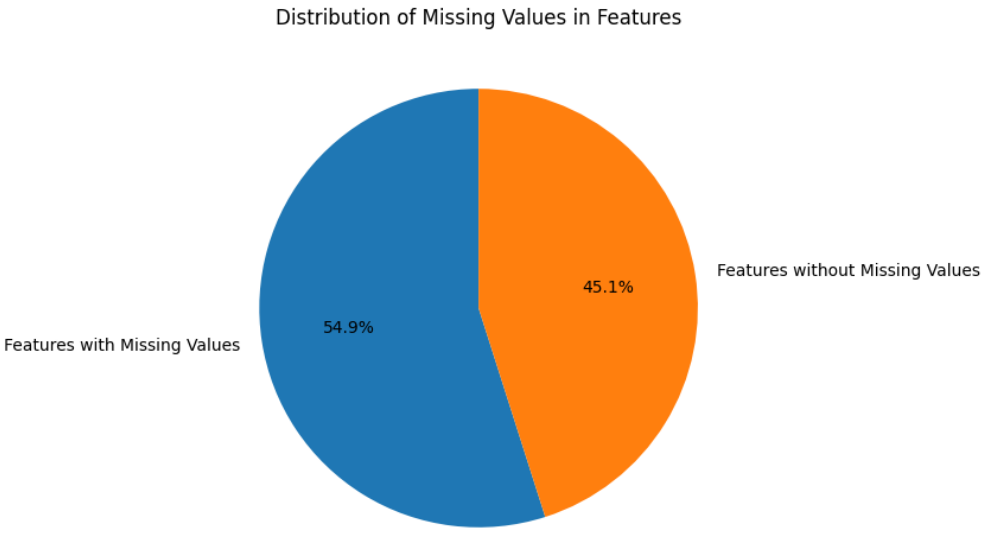

Figure 2: Features of missing values

Further examination of the missing values reveals that several features are missing in the majority of records, potentially affecting their reliability in predictive modeling. These sparsely populated features may contribute little predictive power because of their limited representation across the dataset. Decisions on whether to retain or exclude these features must consider the potential trade-off between preserving the integrity of the data set and mitigating the noise introduced by imputation or other methods.

Analyzing and presenting the distribution of each feature is challenging, given the rel-

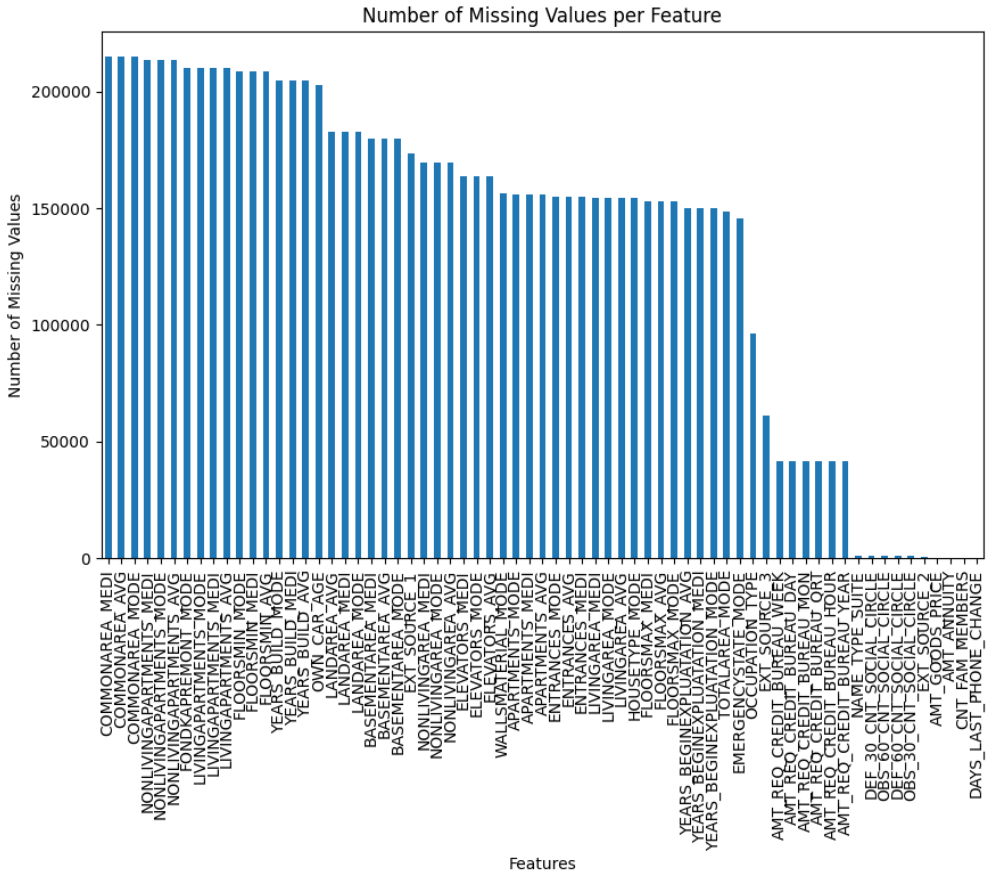

Figure 3: Features of missing values

atively high feature count. Box plots were used to visualize the range and spread of each feature; they can be found in the appendix as fig. 10 & **??**. Observing these visualizations reveals that the dataset comprises both categorical and continuous features, each contributing differently to the loan default prediction model. The boxplots highlight the presence of potential outliers, especially within 'socialcircle'-related features, where certain values deviate significantly from the bulk of the data. This presence of outliers suggests that some records may exhibit unusual patterns in specific features, which might warrant further inspection to ensure that these do not unduly influence model training and evaluation.

A comprehensive pairwise plot of all features is impractical due to the large feature count; however, a selection of key features was plotted to investigate their relationships. These scatterplots in Fig. 4 suggest varying degrees of correlation among selected features. Certain

features demonstrate observable correlation, while others exhibit no discernible relationship. The presence of prominent outliers in some scatterplots is consistent with previous observations from the boxplots, underscoring the need for potential outlier mitigation techniques.

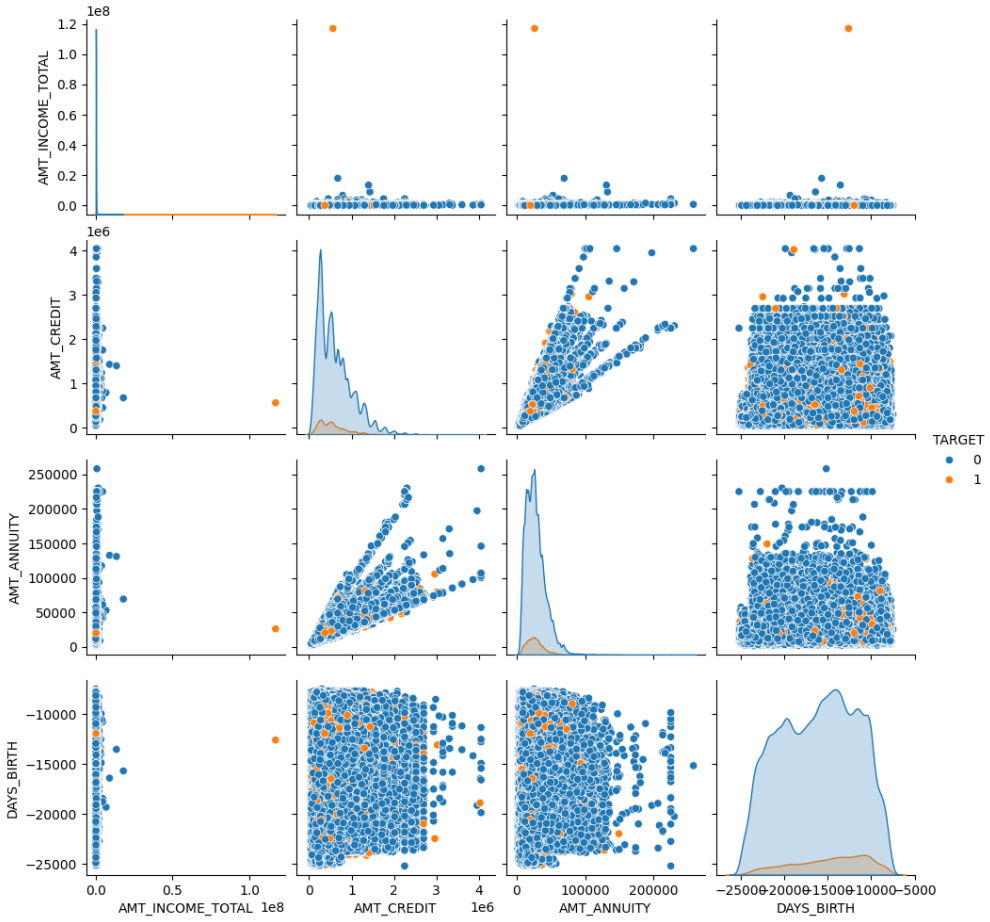

Figure 4: Scatterplot of subset of features

To systematically explore correlations among the features, a correlation matrix was generated; refer to Fig. 5. The correlation matrix shows that most features are weakly or non-correlated with one another, indicating low multicollinearity within the dataset. However, several features exhibit high correlation coefficients with each other, particularly those associated with the debtor's housing situation. This finding is intuitive, as related features are likely to display overlapping information content due to shared dependencies. Such high-correlation clusters could lead to multicollinearity issues, thus affecting the stability and in-

terpretability of the model. Dimensionality reduction methods, such as principal component analysis or feature selection techniques, may be considered to address this multicollinearity while retaining the most informative aspects of the data set.

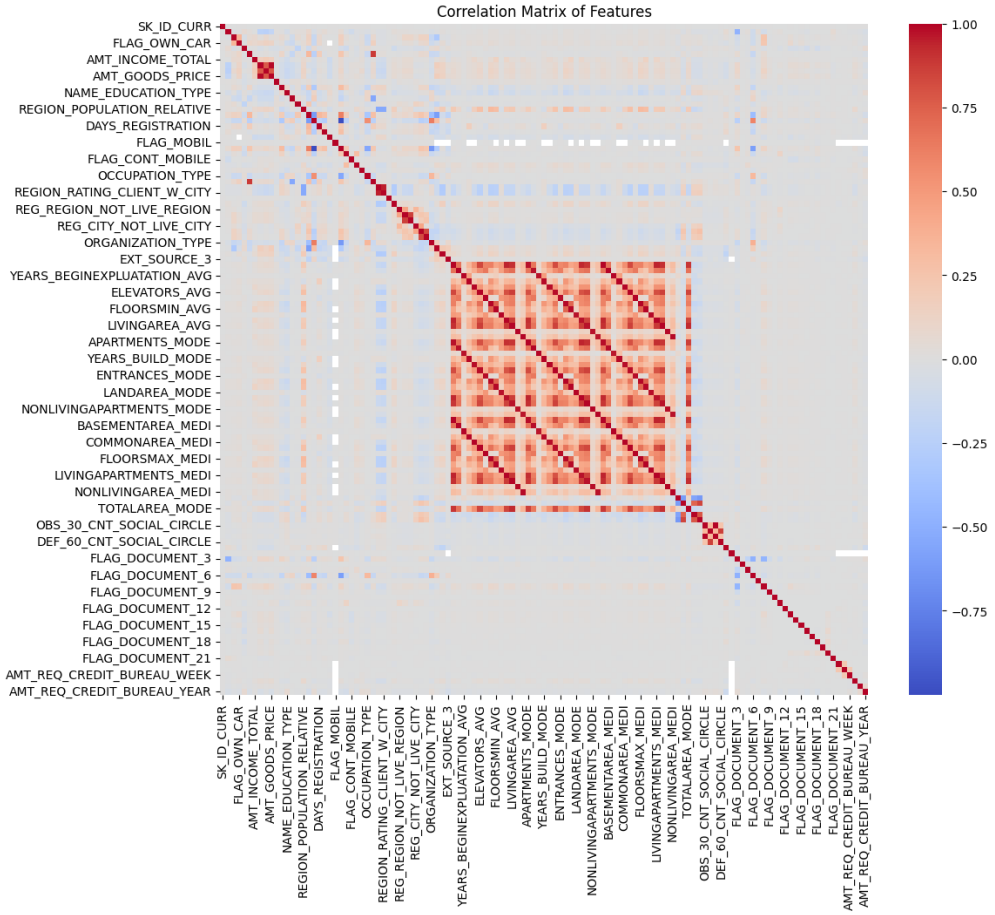

Figure 5: Correlation matrix of numerical features

3 Experimental Setup

This section provides a detailed account of the experimental design, including the data preprocessing steps, methods for handling missing values and categorical variables, feature scaling, and the implementation of feature reduction techniques. The aim of this experimental setup is to prepare the dataset for machine learning models while systematically addressing data quality issues and dimensionality challenges. Two separate pipelines were

constructed: one utilizing Recursive Feature Elimination (RFE) for feature selection and another employing Principal Component Analysis (PCA) for dimensionality reduction. These pipelines allow for a comparative analysis of the effectiveness of each approach in improving model performance and generalizability.

3.1 Data Splitting and Preprocessing

To ensure a robust and unbiased evaluation of model performance, the dataset was divided into training and testing subsets. An 80/20 split was used, allocating 80% of the data to the training subset for model development and retaining the remaining 20% as the testing subset for independent evaluation. This partitioning strategy balances the need for sufficient training data with the requirement for a reliable estimate of out-of-sample performance.

The dataset includes categorical features that cannot be directly processed by machine learning models that rely on numerical inputs. To address this, one-hot encoding was applied. One-hot encoding transforms each categorical feature into a series of binary columns, where each column represents a unique category. This method ensures that no ordinal relationships are introduced among categories, which could otherwise mislead the model into interpreting certain categories as having higher or lower significance.

The dataset exhibits a substantial amount of missing data, both in numerical and categorical features. To mitigate potential biases and maintain the usability of the dataset, a simple imputation strategy was employed. For numerical features, missing values were replaced with the mean of the corresponding feature, calculated from the training subset. This approach ensures that the imputed values are representative of the observed data while avoiding the introduction of extreme or arbitrary values.

For categorical features, missing values were addressed after one-hot encoding by assigning a dedicated binary label to explicitly denote missingness. This approach preserves the information that a value was missing while avoiding any assumptions about the likely category of the missing entry. The inclusion of this explicit label facilitates downstream analyses

by allowing models to treat missingness as an informative feature.

3.1.1 Feature Scaling

Features were standardized using a standard scaler, which adjusts the data to have a mean of zero and a standard deviation of one. Standardization is particularly critical for machine learning algorithms that are sensitive to feature scaling, such as PCA and gradient-based optimization methods. By standardizing the features, the analysis ensures that all variables contribute equally to model training, preventing features with larger numerical ranges from disproportionately influencing the results. This preprocessing step is also a prerequisite for PCA.

3.2 Feature Engineering

To address the challenges posed by the high dimensionality of the dataset, two feature engineering approaches were employed: Recursive Feature Elimination (RFE) and Principal Component Analysis (PCA). These techniques aim to reduce the dimensionality of the dataset while retaining the most informative aspects of the data.

3.2.1 Recursive Feature Elimination (RFE)

Recursive Feature Elimination (RFE) is a systematic feature selection methodology that aims to identify the most relevant predictors by iteratively removing features that contribute the least to the predictive performance of a model. The algorithm proceeds by training a base model and ranking features based on their importance. At each iteration, the feature deemed least important is removed, and the process is repeated until the desired number of features remains. For this study, RFE was configured to retain 10 features, a choice derived through empirical evaluation to ensure an optimal balance between computational efficiency and model performance.

The decision to select 10 features was informed by a series of preliminary experiments

that explored the trade-off between model complexity and predictive accuracy. Various subset sizes were systematically tested, with performance metrics such as accuracy and generalizability serving as evaluation criteria. It was observed that subsets containing fewer than 10 features led to a noticeable decline in predictive accuracy, likely due to the exclusion of important information. Conversely, subsets larger than 10 features exhibited diminishing returns, with marginal improvements in accuracy failing to justify the added computational overhead. Consequently, 10 features were determined to be an appropriate subset size, striking a balance between information retention and model simplicity.

RFE was implemented using a base model trained on the training dataset to ensure the selection of features that maximize the model's predictive power while minimizing redundancy. This iterative process allows for a data-driven identification of the most influential features, ensuring that the resulting model is both parsimonious and effective. Importantly, the use of RFE aligns with the broader objective of improving model interpretability by reducing the feature space to a manageable subset.

The impact of the number of features selected by RFE on model performance is illustrated in Fig. 6. The figure depicts the accuracy of a logistic regression model as a function of the number of features retained during the RFE process. The results indicate that performance initially improves as the number of features increases, peaks at a smaller subset, and subsequently declines before stabilizing at larger feature counts. This behavior underscores the importance of careful selection of the subset size. The choice of 10 features was guided by this analysis, as it represents a point where the model achieves strong performance without excessive complexity. It should be noted, however, that the use of logistic regression as the scoring mechanism in RFE may introduce a bias favoring this specific model, potentially influencing the feature selection process.

Further examination of the selected features highlights their relative contributions to the model, as shown in Fig. 7. The coefficients associated with each feature reveal that certain predictors, such as AMT_CREDIT and AMT_GOODS_PRICE, exhibit significant influence

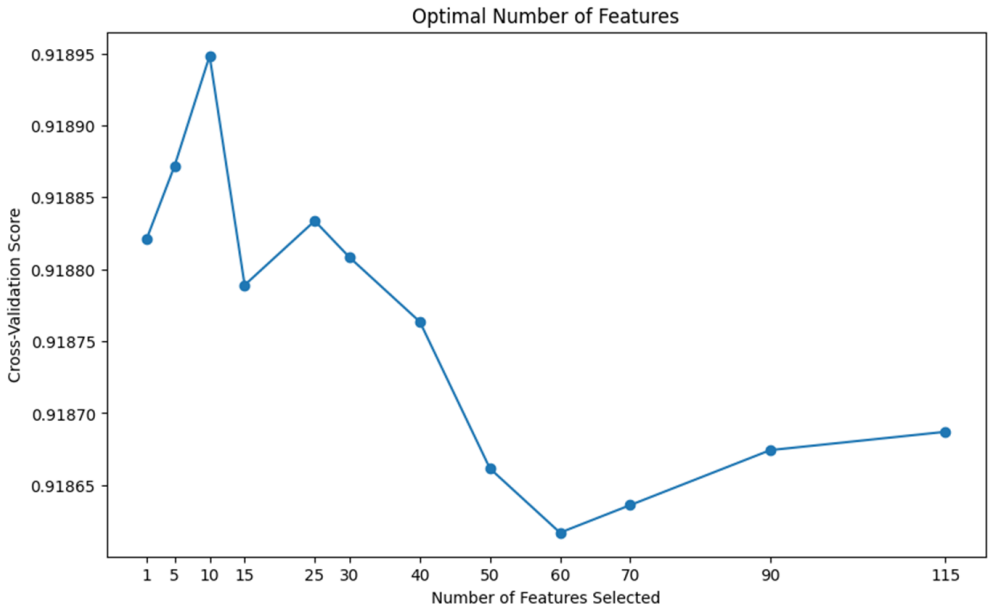

Figure 6: Performance of RFE as a Function of Selected Features

on the model's predictions. In contrast, the remaining features demonstrate comparatively lower but similar contributions. These results suggest that while all selected features contribute to the overall predictive capability of the model, some are more critical than others. This analysis reinforces the importance of both the feature selection process and subsequent evaluation of feature importance to ensure robust and interpretable model development.

3.2.2 Principal Component Analysis (PCA)

PCA is a mathematical technique used for dimensionality reduction. It works by transforming the original feature space into a set of orthogonal components, known as principal components. Each principal component is a linear combination of the original features. The components are ordered based on the amount of variance they explain, with the first component explaining the most variance, the second component explaining the next largest portion, and so on.

In this analysis, PCA was applied to the dataset with the objective of retaining 95% of the total variance. This threshold was selected because it represents a balance between retaining the majority of the data's information and reducing the dimensionality of the

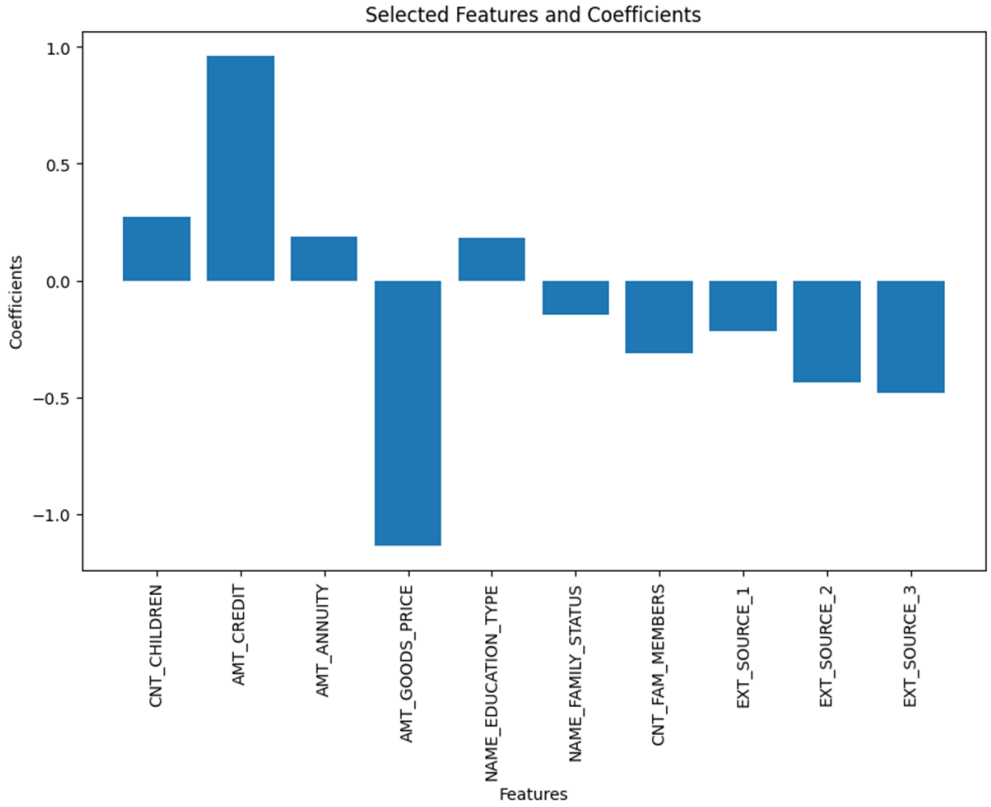

Figure 7: Coefficients of Features Selected by RFE

dataset. By retaining 95% of the variance, the dataset was reduced to a manageable number of dimensions, which helps simplify the subsequent analysis and modeling process.

The analysis of the dataset showed that retaining 95% of the variance required 71 principal components. This means that, to capture most of the variability in the data, only 71 out of the original features were necessary, indicating that much of the data's information could be represented by a smaller number of components. This reduction in dimensionality is a typical outcome of PCA, which simplifies the data structure while preserving its essential characteristics. The number of components required to reach the 95% variance retention level was a key result, indicating how much of the data's information could be condensed into fewer dimensions.

The first figure (Fig. 8) shows the cumulative explained variance as the number of principal components increases. The plot indicates that as more components are added,

the amount of variance explained increases, with a noticeable leveling off after the 71st component. This illustrates how the additional components contribute progressively less to explaining the overall variance. After 71 components, the cumulative variance approaches 95%, and adding more components does not significantly improve the explanation of the data's variability. This suggests that the first 71 components capture the majority of the relevant information in the dataset, making additional components redundant.

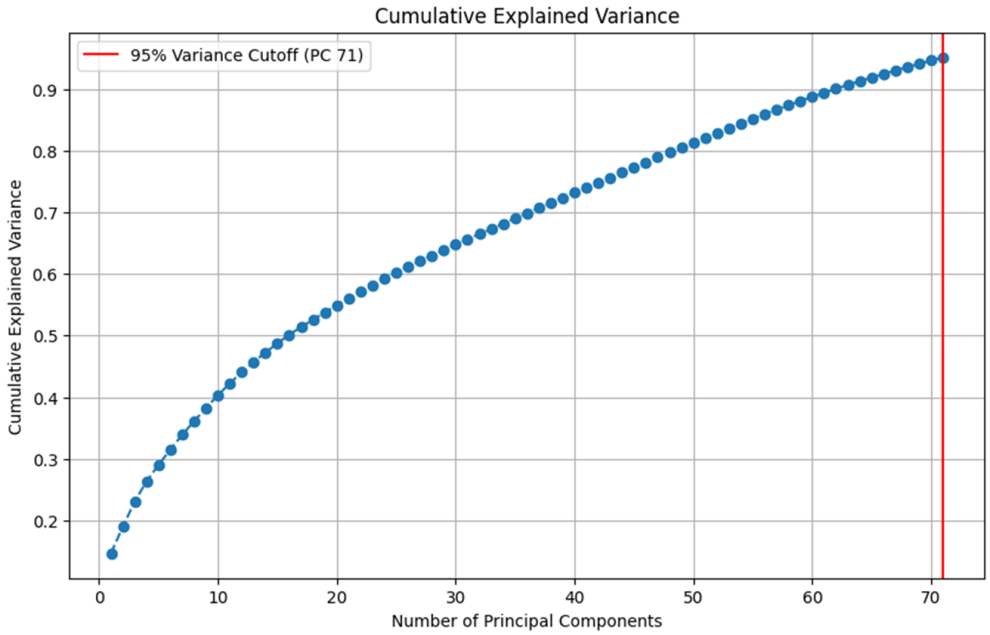

Figure 8: PCA Variance Cutoff

The second figure (Fig. 9) displays the variance ratio of each individual principal component. It shows that the first component explains approximately 0.14 of the total variance, which is significantly higher than the subsequent components. The second and third components each explain around 0.04 of the variance, and the contribution of each subsequent component decreases further, with the later components explaining around 0.02 of the variance. This distribution of variance across the components indicates that a small number of components dominate the variance explanation, while the remaining components explain minimal amounts of variance. This characteristic is typical of PCA, where the first few components capture the majority of the variability in the data, and the remaining components

contribute only marginally.

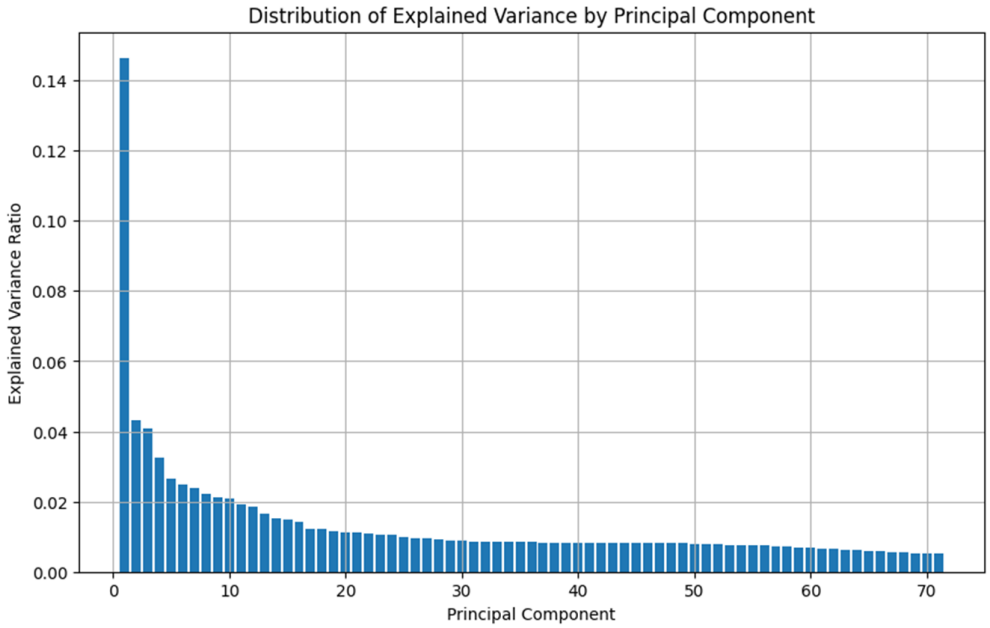

Figure 9: PCA Variance Ratio

The results from the PCA indicate that a substantial amount of the dataset's information can be retained with a relatively small number of principal components. This is particularly useful in reducing the complexity of the data, making it more manageable for modeling and analysis. The large variance explained by the first component, followed by diminishing returns from subsequent components, reflects a common pattern in high-dimensional datasets. It shows that a small subset of components can effectively represent the structure of the original data, which is a key advantage of using PCA for dimensionality reduction.

These results suggest that the dimensionality reduction performed through PCA could be beneficial in simplifying the dataset while retaining most of the important information. The reduction in dimensionality also helps to address potential issues related to overfitting and multicollinearity. By transforming the data into orthogonal components, PCA removes correlations between features, which can improve the stability and interpretability of subsequent models. Furthermore, by reducing the number of features while maintaining high variance retention, PCA helps to streamline the analysis and reduce computational complex-

ity without losing significant information.

3.3 Model Selection

For the initial model selection, logistic regression was chosen as the benchmark model. Logistic regression is a well-established method for binary classification tasks, known for its simplicity and interpretability. It serves as a basic starting point to compare the performance of more complex models. After evaluating logistic regression, three additional models—Random Forest, K-Nearest Neighbors (KNN), and XGBoost—were considered for comparison. These models were selected due to their widespread use and varying characteristics in terms of complexity and interpretability. Random Forest is an ensemble method that combines multiple decision trees to improve predictive performance, while KNN is a simple non-parametric method based on the distance between data points. XGBoost, an advanced gradient boosting model, is known for its high performance in many machine learning tasks.

3.4 Hyperparameter Tuning

Hyperparameter tuning was performed using RandomizedSearchCV. This method was selected as it provides an efficient way to search for optimal hyperparameters by randomly sampling combinations within a specified search space, rather than performing an exhaustive grid search. RandomizedSearchCV was run for 25 iterations with 5-fold cross-validation (CV) to evaluate the model's performance on different subsets of the training data. The use of cross-validation ensures that the model's performance is evaluated in a robust manner, reducing the potential for overfitting. The 25 iterations provide a reasonable balance between exploration of the hyperparameter space and computational efficiency.

The search space for hyperparameter tuning was defined for each of the models considered. Table 1 presents the specific hyperparameter ranges for each model. The search space was carefully selected to cover a broad range of potential values that might optimize each model's performance. The goal of hyperparameter tuning is to identify the best combina-

tion of hyperparameters that maximizes model accuracy, while also ensuring that the model generalizes well to unseen data.

Model	Parameter	Values
Logistic Regression	C	0.001, 0.01, 0.1, 1, 10, 100
	solver	lbfgs, liblinear
Random Forest	n_estimators	25, 50, 100
	max_depth	None, 10, 20
	min_samples_split	2, 5, 10
	min_samples_leaf	1, 2
K-Nearest Neighbors	n_neighbors	3, 5, 7, 10, 15, 20
	weights	uniform, distance
	leaf_size	20, 30, 40
XGBoost	max_depth	3, 5, 7, 9
	learning_rate	0.01, 0.1, 0.2
	n_estimators	25, 50, 100
	subsample	0.8, 1.0
	colsample_bytree	0.8, 1.0
	gamma	0, 0.1, 0.2

Table 1: Hyperparameter search space

3.4.1 PCA

Hyperparameter tuning was performed on models using the dataset transformed by Principal Component Analysis (PCA). As PCA reduces the dimensionality of the data while retaining a significant amount of variance, it can improve the efficiency of models by reducing computational complexity and mitigating issues like multicollinearity. The same hyperparameter search space, as described previously, was applied to the models trained on the PCA-reduced dataset. Table 2 presents the results of the hyperparameter tuning for each model when applied to the PCA-handled dataset. The performance metrics indicate how well the tuned models perform when using the reduced feature set, providing insight into the effect of dimensionality reduction on model accuracy and efficiency.

Model	Parameter	Values
Logistic Regression	C	0.001
	solver	lbfgs
Random Forest	n_estimators	50
	max_depth	20
	min_samples_split	5
	min_samples_leaf	2
K-Nearest Neighbors	n_neighbors	10
	weights	uniform
	leaf_size	40
XGBoost	max_depth	3
	learning_rate	0.1
	n_estimators	100
	subsample	1.0
	colsample_bytree	0.8
	gamma	0.2

Table 2: PCA Hyperparameters

3.4.2 RFE

Similarly, hyperparameter tuning was also conducted for models trained on the dataset processed through Recursive Feature Elimination (RFE). RFE is a feature selection technique that iteratively removes the least important features based on model performance, and it can help improve model accuracy by eliminating irrelevant or redundant features. The same hyperparameter search space was used for RFE-processed data. Table 3 presents the results of the hyperparameter tuning on the RFE-handled dataset. The results show how the models performed after feature elimination, indicating whether the reduced feature set had a positive or negative effect on model performance.

4 Results

The performance of the models on the PCA and RFE datasets is presented in Table 4 and Table 5 respectively. The models were evaluated based on accuracy, precision, recall, and F1-score for both class 0 and class 1. In both datasets, the models showed a strong bias towards

Model	Parameter	Values
Logistic Regression	C	100
	solver	lbfgs
Random Forest	n_estimators	50
	max_depth	10
	min_samples_split	10
	min_samples_leaf	1
K-Nearest Neighbors	n_neighbors	15
	weights	uniform
	leaf_size	30
XGBoost	max_depth	3
	learning_rate	0.01
	n_estimators	50
	subsample	1.0
	colsample_bytree	0.8
	gamma	0.1

Table 3: RFE Hyperparameters

class 0, which is the majority class, and their performance in detecting class 1 instances was poor.

The distribution of class 0 in the dataset was approximately 92%, leading to a situation where the models achieved high accuracy primarily by predicting the majority class. Given that class 1 was the minority class, all models struggled to correctly identify instances of class 1, resulting in very low precision, recall, and F1-scores for this class across all models.

4.1 Model Performance on PCA Dataset

The models trained on the PCA dataset exhibited similar trends. Logistic regression achieved an accuracy of 92.01%, with a cross-validation mean accuracy of 92.19% (+/- 0.0003). The precision, recall, and F1-score for class 0 were 0.92, 1.00, and 0.96, respectively. However, for class 1, the model performed poorly with precision, recall, and F1-scores all equal to zero. This suggests that logistic regression was unable to correctly identify any instances of class 1 in the PCA-handled data.

Similarly, the Random Forest model achieved an accuracy of 92.01%, with a cross-

Model	Accuracy	CV Mean Accuracy	Precision (0)	Recall (0)	F1-Score (0)	Precision (1)	Recall (1)	F1-Score (1)
Logistic Regression	0.9201	0.9219 (+/- 0.0003)	0.92	1.00	0.96	0.00	0.00	0.00
Random Forest	0.9201	0.9219 (+/- 0.0002)	0.92	1.00	0.96	0.00	0.00	0.00
K-Nearest Neighbors	0.9195	0.9220 (+/- 0.0004)	0.92	1.00	0.96	0.00	0.00	0.00
XGBoost	0.9198	0.9222 (+/- 0.0005)	0.92	1.00	0.96	0.33	0.00	0.01

Table 4: Performance results PCA

validation mean accuracy of 92.19% (+/- 0.0002). The precision, recall, and F1-score for class 0 were 0.92, 1.00, and 0.96, respectively. Like logistic regression, Random Forest failed to identify any instances of class 1, resulting in zero precision, recall, and F1-scores for class 1.

K-Nearest Neighbors (KNN) performed similarly, achieving an accuracy of 91.95%, with a cross-validation mean accuracy of 92.20% (+/- 0.0004). The precision, recall, and F1-score for class 0 were 0.92, 1.00, and 0.96. However, for class 1, KNN also showed no ability to identify instances of class 1, with all metrics for class 1 equal to zero.

XGBoost achieved an accuracy of 91.98%, with a cross-validation mean accuracy of 92.22% (+/- 0.0005). The precision, recall, and F1-score for class 0 were 0.92, 1.00, and 0.96. However, while XGBoost was able to predict class 1 with a precision of 0.33, its recall and F1-score for class 1 were still very low (0.00 and 0.01, respectively). This indicates that while XGBoost showed some potential for detecting class 1, it was still largely ineffective at correctly identifying class 1 instances.

4.2 Model Performance on RFE Dataset

The results on the RFE dataset were similar to those on the PCA dataset. Logistic regression achieved an accuracy of 92.05%, with a cross-validation mean accuracy of 92.20% (+/- 0.0003). The precision, recall, and F1-score for class 0 were 0.92, 1.00, and 0.96, but for class 1, all performance metrics were zero, indicating that the model failed to identify class 1 instances entirely.

Random Forest showed an accuracy of 91.88%, with a cross-validation mean accuracy of 92.20% (+/- 0.0007). As with logistic regression, the model's precision, recall, and F1-score for class 0 were 0.92, 1.00, and 0.96. However, the precision, recall, and F1-scores for class 1 were zero, indicating poor detection of class 1 instances.

K-Nearest Neighbors (KNN) achieved an accuracy of 91.79%, with a cross-validation mean accuracy of 92.21% (+/- 0.0009). Again, the model showed high performance for class 0 with precision, recall, and F1-scores of 0.92, 1.00, and 0.96, respectively, but performed poorly for class 1, with all metrics for class 1 equal to zero.

XGBoost achieved an accuracy of 92.01%, with a cross-validation mean accuracy of 92.20% (+/- 0.0002). While the precision, recall, and F1-scores for class 0 were 0.92, 1.00, and 0.96, the performance for class 1 remained low. The precision for class 1 was 0.33, but recall and F1-score were still close to zero (0.00 and 0.01, respectively).

4.3 Summary of Results

In both the PCA and RFE datasets, all models exhibited a similar pattern of performance. The high accuracy scores were driven by the models' ability to correctly classify class 0 instances, which constitute the majority of the dataset. However, all models showed significant difficulty in detecting class 1, the minority class. This is reflected in the zero or near-zero precision, recall, and F1-scores for class 1 across all models, with the exception of XGBoost, which showed a slight improvement in precision but still failed to effectively detect class 1 instances.

Model	Accuracy	CV Mean Accuracy	Precision (0)	Recall (0)	F1-Score (0)	Precision (1)	Recall (1)	F1-Score (1)
Logistic Regression	0.9205	0.9220 (+/- 0.0003)	0.92	1.00	0.96	1.00	0.00	0.01
Random Forest	0.9188	0.9220 (+/- 0.0007)	0.92	1.00	0.96	0.17	0.00	0.01
K-Nearest Neighbors	0.9179	0.9221 (+/- 0.0009)	0.92	1.00	0.96	0.23	0.01	0.02
XGBoost	0.9201	0.9220 (+/- 0.0002)	0.92	1.00	0.96	0.00	0.00	0.00

Table 5: Performance of models RFE

These results highlight the challenges posed by class imbalance, where the models overwhelmingly predict the majority class (class 0) and fail to identify the minority class (class 1). Despite achieving high accuracy scores, the models were largely ineffective in correctly predicting class 1 instances, which is the primary area of interest for this analysis.

In conclusion, while all models performed well in terms of overall accuracy due to the class imbalance, their ability to correctly identify class 1 instances was inadequate. Future work should focus on addressing the class imbalance, possibly through techniques like resampling, adjusting class weights, or utilizing different evaluation metrics such as the area under the precision-recall curve to better capture the performance with respect to the minority class.

5 Conclusion

The performance of multiple machine learning models was evaluated on the same dataset twice, one where PCA was applied and one where RFE was, with the objective of assessing their ability to handle a highly imbalanced classification problem and the impact the dimen-

sionality reduction methods would have. The results indicated that, despite achieving high accuracy scores, all models exhibited a strong bias toward the majority class (class 0), while their performance in identifying instances of the minority class (class 1) was inadequate.

For all models—Logistic Regression, Random Forest, K-Nearest Neighbors, and XGBoost—class 0 was classified with high precision, recall, and F1-score, leading to high overall accuracy. However, the models demonstrated substantial difficulty in detecting class 1, as evidenced by the low precision, recall, and F1-scores for class 1, which were close to zero for most models. XGBoost showed marginal improvement in precision for class 1, but it remained ineffective in correctly identifying instances of class 1.

These results highlight the limitations of conventional classification algorithms in the context of class imbalance, where the majority class dominates the model's predictions, and the minority class is insufficiently detected. Despite the high accuracy scores, the models demonstrated a lack of practical utility for tasks that require effective identification of the minority class.

Future research should focus on addressing the issue of class imbalance by exploring techniques such as resampling (e.g., SMOTE), adjusting class weights, or utilizing alternative evaluation metrics, such as the area under the precision-recall curve, to better reflect model performance on the minority class. Additionally, the exploration of algorithms specifically designed to handle imbalanced datasets could offer potential improvements in the detection of class 1.

In summary, while the models performed well in terms of classifying the majority class, their ability to correctly identify instances of the minority class was severely limited. This emphasizes the need for further research and refinement in handling class imbalance, particularly in contexts where the minority class is of greater significance.

References

Appendix

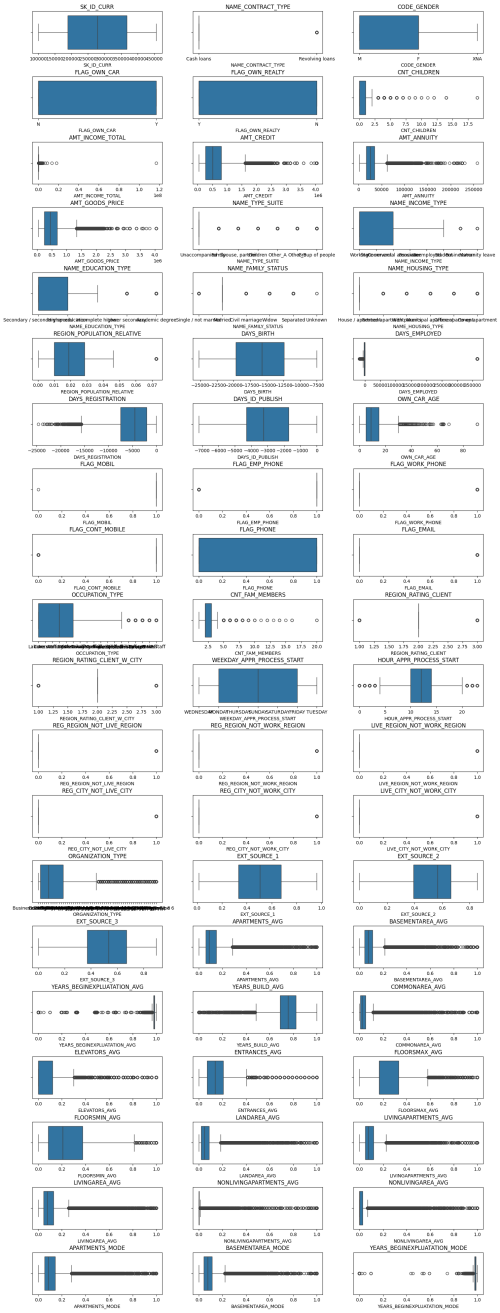

Figure 10: Features of missing values

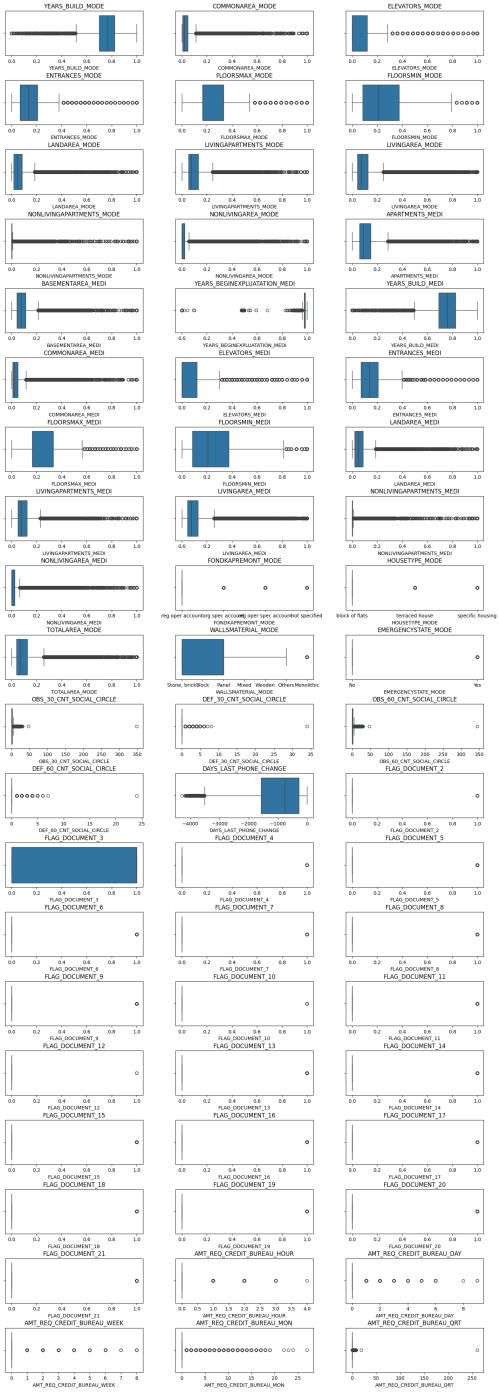

Figure 11: Features of missing values