Imbalanced Classification of Fraudulent Credit Card Transactions Using Machine Learning

1st Gamze Tuncay
Department of Computer Engineering
Istanbul Technical University
Istanbul, Turkey
tuncay15@itu.edu.tr

***Abstract*—Credit card fraud is a significant and growing problem, with billions of transactions occurring every day around the world. Fraudsters are constantly adapting to new technologies, making it difficult to detect and prevent fraud. This study investigates the effectiveness of feature selection and sampling techniques with ML algorithms for imbalanced classification of fraudulent transactions. The model will be trained on a data set of over 284,000 credit card transactions, of which 0.172% are fraudulent. A variety of ML techniques will be used, including Random Forests, Logistic Regression, XGBoost and CatBoost. In addition to ML techniques, resampling techniques to handle imbalance data and feature selection is used to improve the model's performance. The model's performance will be evaluated using a variety of metrics, including the confusion matrix, precision, recall, accuracy, F1-score, area under the curve (AUC) and Matthew's correlation coefficient (MCC).**

***Index Terms*—fraud detection, feature selection, machine learning, resampling, data mining**

I. Introduction

In recent years, credit card transactions are increasing dramatically driven by the growth of digital payment systems, the popularity of e-commerce, mobile banking apps, as well as security enhancements. Digital payment systems make it easier and more convenient for people to use credit cards, the popularity of e-commerce has led to an increase in online purchases which are often made with credit cards, and mobile banking apps have made it easier for people to use credit cards and manage their accounts, and security enhancements have made credit cards more secure and reliable [1].

The fact that billions of transactions occur every day around the world and that people are using their credit cards more than ever through cellphone apps, online shops, and digital wallets means that there are fraudsters and that they too are adapting to these new technologies. Any unauthorized use of a credit card or credit card information to make purchases or obtain cash advances is identified as credit card fraud. This covers credit card information theft techniques such as skimming credit card information, stealing actual credit cards, and using phishing scams. Both cardholders and shops may suffer significant financial damages as a result of credit card fraud.

A. Problem Statement

Fraud detection and prevention are the two methods that can be applied to stop losses caused by fraud. A proactive strategy that prevents fraud from occurring in the first place is fraud prevention [2]. Some of the examples of technologies used to prevent fraud are Address Verification Service (AVS), Card Verification Value (CVV), Personal Identification Number (PIN), and 3-D Secure (3DS). The purpose of AVS is to use the card holder's registered address to verify their identity. PIN entails verifying the customer's entered numerical code. 3DS is a security feature that requires users to enter a code sent to their phone or email address to complete a purchase. CVV is a three-digit number located on the card that is used to confirm that the card is in the customer's hands at the time of purchase. On the other hand, credit card fraud detection refers to the procedures, tools, techniques, and strategies applied by credit card companies and financial institutions to prevent identity theft and stop fraudulent transactions. The majority of current fraud detection solutions utilize data mining techniques to handle data analysis, finding relevant information, identifying anomalies, forecasting, decision-making, and remediation operations.

Detecting credit card fraud has different challenges. The first challenge is that enormous data sets are processed constantly and model building needs to be fast enough to respond to fraud promptly. Such problems have become manageable by using different machine learning (ML) algorithms. They can identify the anomaly and classify it as a fraudulent transaction as they are quick and easy to use. The second problem is the imbalanced data set. The majority of transactions are not fraudulent, therefore it's highly challenging to detect the fraudulent ones. Undersampling the majority class, oversampling the minority class, and generating synthetic samples are the resampling techniques to handle imbalanced data sets [3]. The data sets may contain some features that are redundant or have little impact on the result. Overfitting can occur from using such features for ML, which could make the model more complex. In order to get useful information and generate accurate estimations, dimensionality reduction techniques like feature selection are utilized to deal with the high dimensionality problem [4].

This study aims to implement machine learning pipelines with resampling and feature selection strategies for an imbalance classification of fraudulent credit card transactions. The sampling strategy on the data set, feature selection, and detection techniques employed all have major effects on the performance of fraud detection in credit card transactions. Credit card fraud occurs on a regular basis, resulting in massive financial losses to financial institutions and individuals. Thus, it becomes critically important for financial institutions to reduce financial losses.

B. Contributions

The main contributions of this study can be summarized as follows:

- Investigated the impact of different resampling techniques (SMOTE, Random Oversampling, Tomek's Links) on the performance of fraud detection models
- Evaluated the effect of uni-variate feature selection method on model performance, shedding light on the interpretability of the selected features
- Conduct detailed experiments on public data set to analyze different aspects of ML methods (Random Forest, Logistic Regression, XGBoost, CatBoost) on credit card fraud detection
- Developed customized ML pipelines tailored for imbalanced data sets, incorporating resampling techniques and feature selection methods seamlessly

The rest of the paper is organized as follows. Section II presents related works about imbalanced classification for credit card fraudulent transactions. Section III gives details of proposed ML-based fraud detection model and background methodology. Section V presents performance evaluation and experimental results of the study. Lastly, Section **??** concludes the paper.

II. Related Works

Credit card fraud is a problem that banks and financial institutions must deal with. It happens when money or property is obtained fraudulently through the use of credit cards by outsiders. It is considered essential to set up efficient methods for identifying fraudulent transactions. In general, fraud detection is seen as a data mining classification problem, with the goal being to accurately categorize credit card transactions as either valid or fraudulent.

A. Resampling

Several sampling approaches have been employed to address imbalanced data in credit card transactions. These methods aim to enhance the performance of fraud detection models by mitigating the impact of the skewed distribution of fraudulent and non-fraudulent transactions.

In [3], the Tomek links algorithm is used as a data cleaning method followed by random subsampling as a data cleaning option. They report that this use improves the performance of the classification algorithm by removing noise from the majority class and helps to reduce the likelihood of information loss with random oversampling. A random sampling approach is used in [5] and they found that random oversampling can improve the performance of classification models by balancing the classes and reducing the impact of the minority class. However, they also noted that random oversampling may lead to over-fitting and reduced generalization performance, especially when the data set is small or the imbalance is extreme [5]. The paper [6] used data from the real world to find fraudulent transactions using SMOTE and ML algorithms. Furthermore, the effectiveness of Decision Trees, Logistics Regression, and Random Forest is compared by using the "With SMOTE" and "Without SMOTE" techniques. They reported that, performance of every metric is improved with SMOTE compared to without SMOTE. According to Sisodia, Reddy and Bhandari [7], when SMOTE is applied along with Random Forests algorithm for credit card fraud detection, the performance of algorithm is improved based on precision, F1 score, Matthew's correlation coefficient (MCC) and recall metrics.

B. Feature Selection

Feature selection enhances the performance and interpretability of credit card fraud detection models. By systematically choosing relevant features and discarding irrelevant ones, feature selection optimizes model's efficiency, enhances model generalization and facilitates model validation by ensuring that the model's performance is evaluated based on meaningful features.

In 2022, Ileberi, Sun and Wang proposed a Generic Aalgorithm-based feature selection method in conjunction with several supervised ML algorithms [8]. The experimental results that were achieved using the Generic Algorithm selected attributes demonstrated that the Generic Algorithm with Random Forest achieved an overall optimal accuracy of 99.98%, which was superior to those achieved by existing methods [8]. In the academic study conducted by Bayhan, Yavuz Güvensan ve Karsligil [9], all features were ranked according to their importance using an iterative Feature Selection method on credit card fraud detection success. Classification processes were tried by selecting the most important features in various numbers and it was reported that the success was at the highest level when 30 features were selected [9]. Lima and Pereira proposed a voting approach to choose the best features [10]. The study achieved a very good performance in fraud detection using the proposed methodology, reducing the number of features and presenting financial gains of up to 61% compared to the actual scenario of the company [10].

C. Fraud Detection

Even with the outstanding developments in credit card fraud detection that have been reported in recent research [11], [12], statistical techniques and AI-based strategies remain crucial for effectively addressing fraud detection. Due to their high prediction accuracy, deep learning (DL) and ML models are widely utilized in anomaly detection applications, regardless of

the industry such as medicine, remote sensing, cyber security [13].

In 2018, Mishra and Ghorpade reported [14] the results and conclusions of several machine learning models used for credit card fraud detection. The models include Logistic Regression, Support Vector Machines, Random Forest, Stacked Classifiers, and Gradient Boosted Trees. The results show that Random Forest performs the best on the actual data set, with a recall of around 96% and a precision of 0.05. The other models also perform well, with precision and recall over 90%. The conclusion is that these models can significantly improve fraud detection and reduce mis-classification of genuine transactions as fraud. Varmedja et al. in 2019 [15]. authored a paper that evaluated the performance of different machine learning algorithms, including Gradient Boosting, Support Vector Machines, Decision Tree, Logistic Regression, Random Forest, and Multi-Layer Perceptron. They used metrics such as accuracy, recall, and precision to compare the performance of these algorithms. The results showed that oversampling the data can improve fraud detection rate, and classical algorithms can be as successful as deep learning algorithms.

III. Experimental Setups and Methods

A variety of techniques is used to train a model to accurately identify fraudulent transactions (Fig 1). The pre-processing step covers removing duplicate rows from the data, converting the time data into hour format that is more suitable for ML algorithms, and normalizing the amount feature so that all values have the same scale.

After pre-processing, to evaluate the performance of different resampling, feature selection, and ML algorithms, a series of experiments are conducted. The considered options are four different resampling options (SMOTE, Random Oversampling, TOMEK Undersampling, and No Resampling), three different feature selection options (Select K Best Feature with k=10, Select K Best Feature with k=15, and No Feature Selection), and four different ML algorithms (Random Forest, Logistic Regression, XGBoost, and CatBoost). This resulted in a total of 4 x 3 x 4 = 48 experiments.

Before the experiments, Grid Search and 5-fold Cross Validation methods were applied to find out best hyper parameter for each ML models. The explored parameters and best parameters for each model is given the Table I.

For each experiment, the data was divided into a training set and a test set. The resampling technique was applied to the training set. Subsequently, the feature selection technique was applied to the training set. Finally, the training set was utilized to train the ML algorithm, and its performance on the test set was assessed. This process was repeated for all 48 experiments.

The impacts of resampling, and feature selection techniques for each ML algorithms on the performance of the model was systematically evaluated through this experimental setup. The expectation is that the best combination of methods for the data set and task will be identified through this experiment. Also, the effectiveness of feature selection and sampling

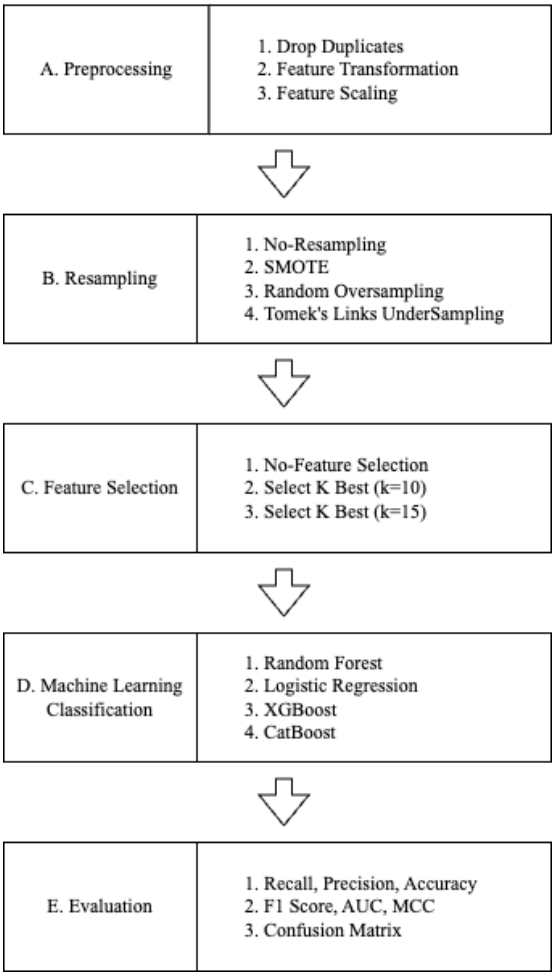


Fig. 1. The applied methodology

TABLE I
Grid Space Parameters for Each Model

Models	Grid Space
Random Forest	criterion : ['gini', **'entropy'**], max_depth': [5, 10, **15**], max_features : [**3**], min_samples_split : [2, **4**, 8], n_estimators : [10, **30**, 50, 100, 150], random_state : [**13**]
Logistic Regression	C : [0.001, 0.01, 0.1, **1**, 10, 100, 1000], max_iter : [**100**, 1000], penalty : [**'l2'**]
XGBoost	learning_rate : [**0.2**, 0.1, 0.01], max_delta_step : [0,1,**2**,4], max_depth : [1,**3**,5], min_child_weight : [1,**4**,7], subsample : [0.8, **0.9**, 1]
CatBoost	depth : [1,**5**,10], iterations: [50,**100**], l2_leaf_reg: [0,**1**], learning_rate: [**0.2**, 0.1, 0.01]

techniques with ML algorithms for imbalanced classification was investigated.

A. Dataset

The goal of this study is to develop a model that can accurately identify fraudulent transactions. This is a challenging task, as the data set is highly unbalanced and there are a variety of factors that can contribute to fraud.

The data set contains over 284,000 credit card transactions, of which 0.172% are fraudulent. The transactions are labeled as either fraudulent or genuine. The features in the data set are numerical and are the result of a principal component analysis (PCA) transformation (Fig 2). The target variable is the class, which is 1 for fraudulent transactions and 0 for genuine transactions [16].

After pre-processing, the correlations between features were calculated. It is apparent from the correlation matrix (Fig 3) that the PCA features do not correlate with each other, however some features do correlate either positively or negatively with Hour and Amount features. This allows us to gain a better understanding of the data at our disposal. Also, we can conclude following statements:

- V2 and V5 are highly negatively correlated with Amount.
- V7 and V20 is correlated with Amount.
- V9 is highly negatively correlated with Hour.

B. Resampling

Resampling is a technique employed to equalize the number of instances per class, involving either under-sampling to decrease majority class instances or over-sampling to increase minority class instances [17]. Under-sampling randomly eliminates majority class instances, offering advantages such as improved run time and storage efficiency but potentially leading to biased remaining data and reduced accuracy for class distribution [18]. On the other hand, over-sampling randomly replicates minority class instances to achieve a balanced class distribution without eliminating data. While over-sampling avoids data loss and generally outperforms under-sampling, it may introduce over-fitting issues due to replicated instances [18].

1) SMOTE (Synthetic Minority Over-sampling Technique): It works by creating synthetic examples of the minority class. These synthetic examples are created by taking a minority class example and interpolating between it and its k nearest neighbors. This creates new examples that are located near the original minority class examples, but are not exactly the same. This prevents the over-fitting issue from occurring during the training phase. Also, one of the most frequently used methods for addressing the problem of class imbalance in datasets is SMOTE [19].

2) Random Oversampling: In order to balance the dataset, random oversampling selects examples from the minority class at random and duplicates them [20]. It is computationally efficient and simple for implementation.

3) Tomek's Links Undersampling: It works by identifying and eliminating samples from the majority class that are noisy or ambiguous. The method identifies pairs of points (known as Tomek links) that are more closely spaced apart than any two points in the same class [17].

C. Feature Selection

As a preprocessing strategy for data, feature selection has demonstrated effectiveness and efficiency, particularly in the preparation of high-dimensional data for diverse data-mining and ML challenges. The primary goals of feature selection include the creation of simpler and more comprehensible models, enhancement of data-mining performance, and the preparation of clean, easily understandable data [21].

1) Select K Best Feature: Select K Best algorithm scores and ranks the features according to how they relate to the output variable using statistical tests such as the mutual information score, ANOVA F-test, and chi-squared test. After scoring, all features are deleted except for the top k features [22]. This study uses the ANOVA F-test as the scoring parameter.

D. Machine Learning Classification

1) Random Forest: It is a powerful ensemble learning method and is a popular choice in ML due to its robust performance across diverse domains. Breiman (2001) introduced the concept of random forests as an ensemble of decision trees, where each tree is constructed using a random subset of features and trained on a bootstrap sample of the data. This ensemble approach mitigates over-fitting and enhances the model's generalization capability [23].

Random Forest operates through a process of bagging (Bootstrap Aggregating) and employs multiple decision trees, each trained on a different subset of the data set. The combination of individual trees' predictions, typically through a majority voting scheme, results in a more robust and accurate overall prediction. This ensemble technique leverages the diversity among the trees to enhance the model's performance [24].

Mathematically, the prediction of a Random Forest model can be represented as:

$$\hat{Y} = \text{mode}\{h_1(X), h_2(X), \ldots, h_n(X)\} \tag{1}$$

where $\hat{Y}$ is the predicted output, $h_i(X)$ represents the prediction of the i-th decision tree, and $mode$ mode denotes the mode function selecting the most frequent prediction.

The random subset of features used in each tree construction ensures de-correlation among the trees, contributing to the model's stability and preventing it from being overly influenced by a single feature [25]. The Random Forest algorithm has been successfully applied in various fields, including classification, regression, and feature selection, making it a versatile and widely adopted tool in ML [25].

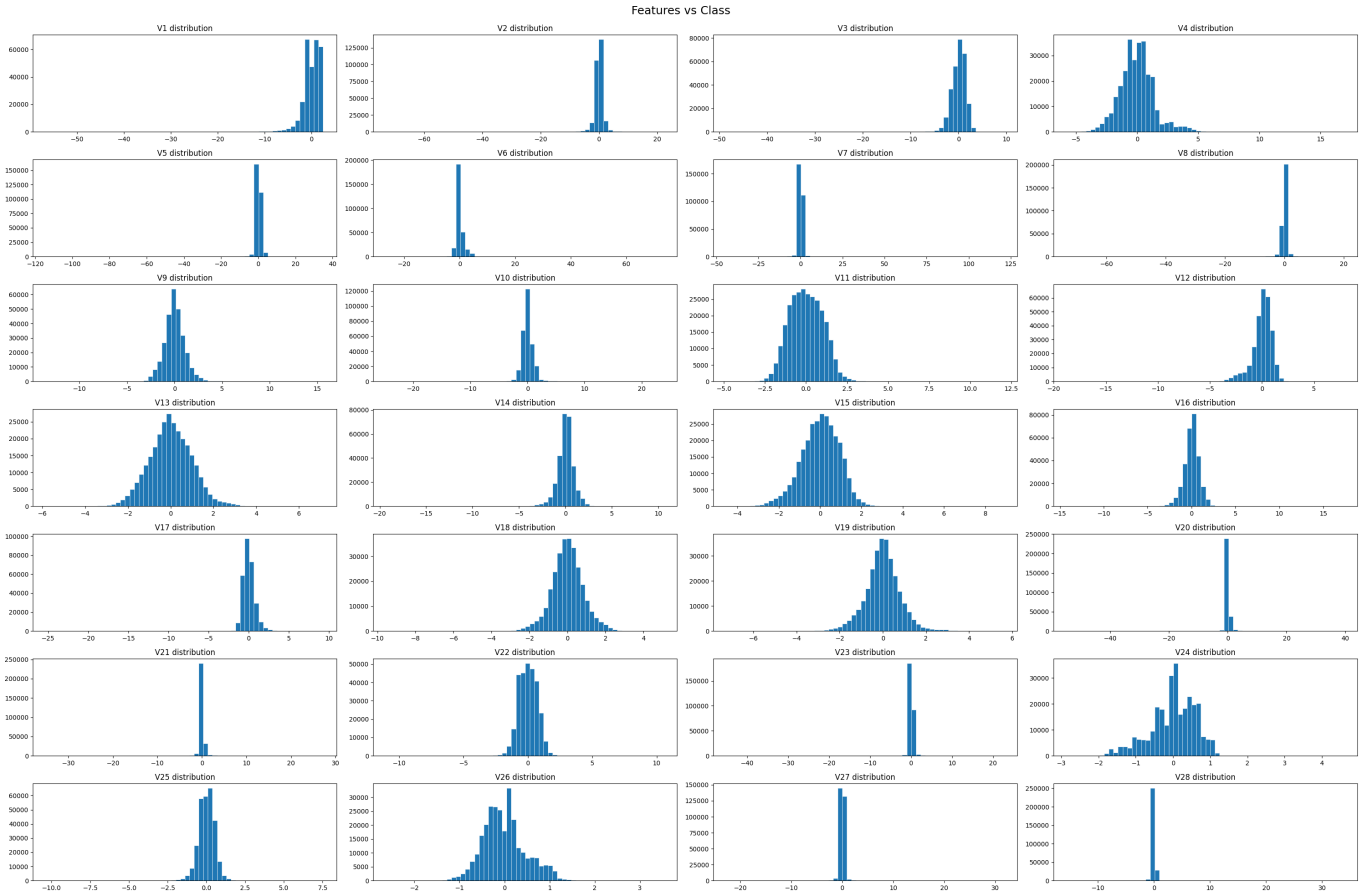


Fig. 2. Distribution of PCA features

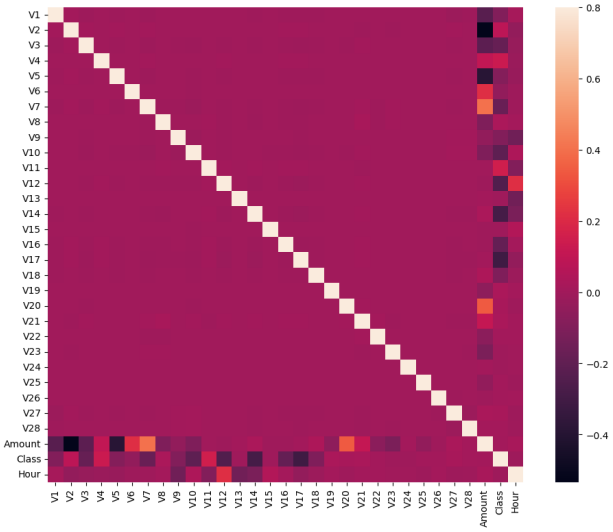


Fig. 3. Correlation matrix of features

2) Logistic Regression: For many binary classification tasks, Logistic Regression is a popular ML algorithm that is simple to use and effective. Because of its ease of interpretation, it is a good option for problems where it is crucial to understand how the input features relate to the output.

It predicts the probability of an event occurring and it is based on the logistic function, which is a sigmoid function that squashes its input between 0 and 1 [15]. Logistic regression learns a linear model that relates the input features to the probability of the positive class given a set of training data with input features and binary labels. A weight vector w, with each weight corresponding to an input feature, is used to represent the linear model. For a given input x, the probability of the positive class is computed as follows [26]:

$$probability : p(y = 1|x) = \sigma(w^T x) \tag{2}$$

$$sigmoidfunction : \sigma(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

3) XGBoost: XGBoost (eXtreme Gradient Boosting) is a decision-tree-based ensemble machine learning algorithm. It has remarkable performance on a wide range of tasks, including classification, regression, and ranking. Unlike traditional gradient boosting algorithms, XGBoost incorporates several innovative techniques to improve its efficiency and accuracy.

XGBoost follows an iterative approach to building an ensemble of decision trees. Each tree in the ensemble aims to minimize the prediction error of the previous trees, gradually improving the overall model's performance [27]. XGBoost

optimizes an objective function that consists of two components: loss function and regularization term. The loss function measures the prediction error, while the regularization term penalizes model complexity [27].

Additionally, XGBoost provides important insights including include cache access patterns, partitioning, and data compression to assist in the development of a quick and scalable tree boosting system [28]. XGBoost exceeds the majority of other machine learning algorithms in terms of speed and accuracy thanks to these mechanisms and insights [28].

4) CatBoost: With the use of binary decision trees as base predictors and gradient boosting, CatBoost [29] is a potent machine learning algorithm that produces state-of-the-art outcomes in a range of useful tasks. In contrast to XGBoost, CatBoost presents a unique ordering principle to address prediction shift brought on by a particular type of target leakage. In fact, this causes a shift in the distribution of FX—X for a training instance X from the distribution of FX—X for a test example X [30]. Assume that a prediction model F obtained after multiple steps of boosting relies on the targets of all training examples [30]. This ultimately causes the learned model's predictions to change. To prevent prediction shift, the CatBoost uses ordered boosting, a variation of the conventional gradient boosting algorithm [30].

IV. Performance Evaluation and Results

A. Metrics

The performance of the proposed model will be evaluated using a variety of metrics, including the confusion matrix, precision, recall, accuracy, F1-score, AUC and MCC. These metrics are commonly used in the literature on fraud detection to evaluate the performance of fraud detection models.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{7}$$

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}} \tag{8}$$

The confusion matrix is a table that shows the number of transactions that are correctly and incorrectly classified by the model. Precision is the fraction of transactions that are predicted to be fraudulent and that are actually fraudulent (4). Recall is the fraction of fraudulent transactions that are correctly predicted by the model (5).Accuracy is the fraction of all transactions that are correctly predicted by the model (6). The F1-score is a harmonic mean of precision and recall. AUC is a measure of how well a model can distinguish between positive and negative examples (7). It is calculated by plotting the receiver operating characteristic (ROC) curve and then calculating the area under the curve.

B. Results

In this study, 48 classifier models based on Random Forest, Logistic Regression, XGBoost and CatBoost are developed. To evaluate these models, 70% of the data set is used for training and validation while 30% is set aside for testing. The experimental results are shown in the Fig VI-B. According to results,

- (Purple) Logistic Regression with SMOTE Sampling and Select K Best (k=10) feature selection model is best performed on recall metric.
- (Blue) CatBoost without resampling and feature selection algorithm is best performed model on precision metric.
- (Green) Random Forest with Select K Best (k=15) feature selection is best performed on Accuracy metric.
- (Red) Random Forest with Random Over Sampling and Select K Best (k=10) feature selection model is best performed on F1 Score metric.
- (Orange) XGBoost with Select K Best (k=15) feature selection model is best performed on AUC metric.
- (Yellow) XGBoost with Tomek's Links model is best performed on MCC metric.

Mean of metrics are calculated according to resampling techniques, feature selection techniques and classifiers (Fig 4). An observation of the results shows that there is significant improvement from the re-sampled data set for recall. However, considering other metrics, sampling techniques are not successfully improves the performance of classification. In addition, Feature selection algorithms have no remarkable changes on recall, accuracy, and AUC metrics. Overall, Select K Best (k=15) is best feature selection technique, but not as good as no feature selection. Moreover, Random Forest is the best performed classifier for precision, accuracy, F1 Score and MCC. Especially, it has significant difference in precision, F1 Score and MCC metrics. On the other hand, XGBoost is other best performed classifier considering recall and AUC.

Cansidering each ML models, we can also conclude that:

- XGBoost exhibited strong overall performance. Particularly, the XGBoost model with Tomek links and no feature selection demonstrated high MCC and accuracy, showcasing its robustness in handling imbalanced datasets.
- Random Forest models, while competitive, showed varying performance under different conditions. Notably, Random Forest with SMOTE and no feature selection achieved a balanced trade-off between precision and recall, indicating its adaptability to imbalanced data.
- Logistic Regression showed competitive results in recall metric, especially when coupled with SMOTE and Select K Best feature selection. But there is no a balance between other metrics. For example precision is 0.0553 and F1 score is 0.1040. These metrics are too low so

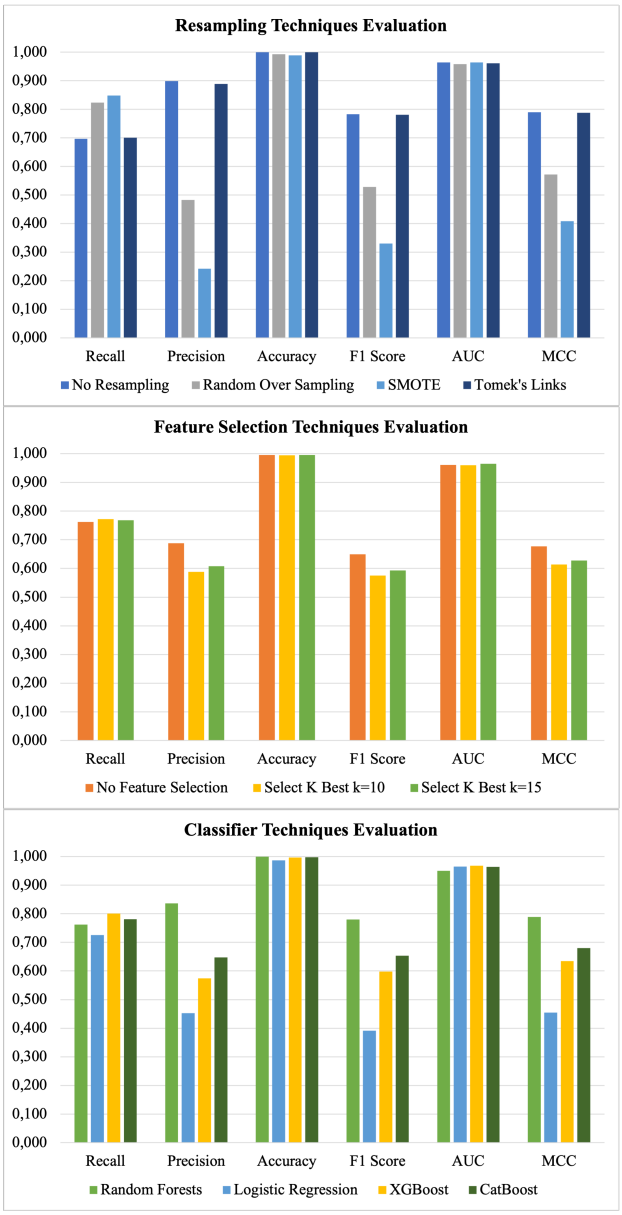


Fig. 4. Resampling Techniques Evaluation, Feature Selection Techniques Evaluation, Classifiers Evaluation

the model is not actually useful in terms of imbalanced classification.

- CatBoost demonstrated competitive performance across different resampling and feature selection strategies. Models with no resampling and no feature selection achieved highest precision and high recall, emphasizing CatBoost's potential for imbalanced datasets.

In addition to evaluating model performance, the study assessed the computational efficiency of each approach, considering the training times for the models. Time results in seconds are shown in Table [**?**]. The training time, defined as the time taken for the ML models to learn patterns from the training data, is a critical factor in real world applications, particularly in scenarios where fast model deployment is essential.

TABLE II
TIME EFFICIENCY EVALUATION

	Train_Time	Test_Time	Total_Time
Random Forests	26,319	0,135	26,453
Logistic Regression	7,626	0,010	7,636
XGBoost	7,516	0,026	7,542
CatBoost	8,035	0,025	8,060
No Resampling	5,880	0,066	5,946
Random Over Sampling	4,328	0,036	4,364
SMOTE	10,008	0,058	10,066
Tomek's Links	29,280	0,036	29,316
No Feature Selection	13,903	0,057	13,960
Select K Best k=10	11,608	0,045	11,653
Select K Best k=15	11,611	0,045	11,657

Considering model time efficiency, among the models evaluated, Random Forests exhibited a total time of approximately 26.45 seconds, with a notable training time of 26.32 seconds. Despite being computationally intensive during training, the test time and overall time remain relatively low. Logistic Regression demonstrated the shortest total time, approximately 7.64 seconds, with a brief training time of 7.63 seconds. The model's simplicity and efficiency are reflected in its swift computation during both training and testing phases. XGBoost had a total time of around 7.54 seconds. Its efficient training time 7.52 seconds contributes to its overall time effectiveness. CatBoost's total time was approximately 8.06 seconds, with a training time of 8.04 seconds and it showcases competitive time performance.

Also, models trained without resampling techniques displayed a mean of a total time of about 5.95 seconds, emphasizing the reduced computational overhead in the absence of resampling processes. Random oversampling models incurred a total time of around 4.36 seconds, indicating a relatively faster training process compared to other resampling methods. SMOTE, despite its effectiveness in handling imbalanced data, required a longer total time of approximately 10.07 seconds, primarily due to the additional data generation steps involved.Models with Tomek's Links demonstrated a total time of about 29.32 seconds, highlighting a more computationally intensive process during training.

Furthermore, models without feature selection techniques had a total time of approximately 13.96 seconds. The absence of feature selection contributes to a relatively higher overall time. Feature selection with Select K Best (k=10) resulted in a total time of around 11.65 seconds, showcasing a balance between reduced feature dimensions and computational efficiency. Similarly, Select K Best (k=15) exhibited a total time of about 11.66 seconds, demonstrating comparable time efficiency with a slightly larger feature set.

Model	Recall	Precision	Accuracy	F1 Score	AUC	MCC	Train Time	Test Time	Total Time
XGBoost - Tomek - Select K Best k=15	0,7535	0,8560	0,9994	0,8015	0,9748	0,8028	26,024	0,018	26,042
XGBoost - Tomek - Select K Best k=10	0,7746	0,8462	0,9994	0,8088	0,9711	0,8093	26,025	0,017	26,043
XGBoost - Tomek - No Feature Selection	0,7606	0,9391	0,9995	0,8405	0,9695	0,8449	27,857	0,026	27,883
XGBoost - SMOTE - Select K Best k=15	0,8521	0,1180	0,9891	0,2074	0,9693	0,3148	1,204	0,018	1,222
XGBoost - SMOTE - Select K Best k=10	0,8662	0,0944	0,9859	0,1702	0,9583	0,2833	1,325	0,037	1,362
XGBoost - SMOTE - No Feature Selection	0,8521	0,1845	0,9935	0,3033	0,9687	0,3947	2,124	0,034	2,159
XGBoost - Random - Select K Best k=15	0,8380	0,3790	0,9974	0,5219	0,9656	0,5626	0,995	0,016	1,011
XGBoost - Random - Select K Best k=10	0,8310	0,3287	0,9969	0,4711	0,9558	0,5215	0,860	0,020	0,880
XGBoost - Random - No Feature Selection	0,8239	0,4875	0,9983	0,6126	0,9667	0,6330	1,295	0,017	1,313
XGBoost - No Resampling - Select K Best k=15	0,7254	0,8655	0,9994	0,7893	0,9750	0,7920	0,749	0,037	0,786
XGBoost - No Resampling - Select K Best k=10	0,7746	0,8661	0,9994	0,8178	0,9710	0,8188	0,598	0,037	0,636
XGBoost - No Resampling - No Feature Selection	0,7606	0,9231	0,9995	0,8340	0,9725	0,8376	1,136	0,038	1,174
Random Forest - Tomek - Select K Best k=15	0,7606	0,9153	0,9995	0,8308	0,9427	0,8341	38,113	0,101	38,213
Random Forest - Tomek - Select K Best k=10	0,7676	0,9160	0,9995	0,8352	0,9408	0,8383	37,330	0,089	37,418
Random Forest - Tomek - No Feature Selection	0,7183	0,9533	0,9995	0,8193	0,9364	0,8273	37,324	0,101	37,424
Random Forest - SMOTE - Select K Best k=15	0,8099	0,4733	0,9982	0,5974	0,9682	0,6183	28,475	0,131	28,606
Random Forest - SMOTE - Select K Best k=10	0,7958	0,3404	0,9971	0,4768	0,9615	0,5193	29,746	0,141	29,887
Random Forest - SMOTE - No Feature Selection	0,7958	0,7793	0,9993	0,7875	0,9672	0,7871	45,417	0,232	45,649
Random Forest - Random - Select K Best k=15	0,7535	0,9386	0,9995	0,8359	0,9598	0,8408	12,908	0,099	13,007
Random Forest - Random - Select K Best k=10	0,7606	0,9391	0,9995	0,8405	0,9297	0,8449	12,972	0,096	13,068
Random Forest - Random - No Feature Selection	0,7535	0,9469	0,9995	0,8392	0,9456	0,8445	13,663	0,105	13,768
Random Forest - No Resampling - Select K Best k=15	0,7394	0,9375	0,9995	0,8268	0,9599	0,8324	19,688	0,158	19,846
Random Forest - No Resampling - Select K Best k=10	0,7535	0,9469	0,9995	0,8392	0,9515	0,8445	19,594	0,145	19,739
Random Forest - No Resampling - No Feature Selection	0,7324	0,9541	0,9995	0,8287	0,9407	0,8357	20,595	0,221	20,816
Logistic Regression - Tomek - Select K Best k=15	0,5634	0,8602	0,9991	0,6809	0,9628	0,6958	25,531	0,007	25,538
Logistic Regression - Tomek - Select K Best k=10	0,5634	0,8511	0,9991	0,6780	0,9655	0,6920	25,156	0,015	25,171
Logistic Regression - Tomek - No Feature Selection	0,5704	0,8438	0,9991	0,6807	0,9658	0,6934	26,955	0,006	26,960
Logistic Regression - SMOTE - Select K Best k=15	0,8803	0,0544	0,9743	0,1024	0,9633	0,2152	0,804	0,017	0,821
Logistic Regression - SMOTE - Select K Best k=10	0,8873	0,0553	0,9745	0,1040	0,9649	0,2179	0,782	0,010	0,792
Logistic Regression - SMOTE - No Feature Selection	0,8803	0,0514	0,9727	0,0971	0,9638	0,2090	4,287	0,014	4,301
Logistic Regression - Random - Select K Best k=15	0,8873	0,0553	0,9745	0,1040	0,9658	0,2179	0,672	0,008	0,681
Logistic Regression - Random - Select K Best k=10	0,8873	0,0561	0,9749	0,1055	0,9650	0,2196	0,540	0,005	0,544
Logistic Regression - Random - No Feature Selection	0,8873	0,0525	0,9731	0,0991	0,9653	0,2121	3,160	0,013	3,173
Logistic Regression - No Resampling - Select K Best k=15	0,5634	0,8602	0,9991	0,6809	0,9628	0,6958	0,814	0,012	0,826
Logistic Regression - No Resampling - Select K Best k=10	0,5634	0,8511	0,9991	0,6780	0,9655	0,6920	0,617	0,008	0,625
Logistic Regression - No Resampling - No Feature Selection	0,5704	0,8438	0,9991	0,6807	0,9656	0,6934	2,194	0,010	2,204
CatBoost - Tomek - Select K Best k=15	0,7254	0,8655	0,9994	0,7893	0,9662	0,7920	25,989	0,018	26,007
CatBoost - Tomek - Select K Best k=10	0,7042	0,8696	0,9993	0,7782	0,9688	0,7822	26,534	0,018	26,552
CatBoost - Tomek - No Feature Selection	0,7394	0,9459	0,9995	0,8300	0,9675	0,8361	28,527	0,018	28,544
CatBoost - SMOTE - Select K Best k=15	0,8662	0,1866	0,9935	0,3071	0,9609	0,4004	1,355	0,020	1,375
CatBoost - SMOTE - Select K Best k=10	0,8662	0,1515	0,9917	0,2579	0,9583	0,3603	1,443	0,023	1,466
CatBoost - SMOTE - No Feature Selection	0,8310	0,4126	0,9977	0,5514	0,9607	0,5846	3,131	0,018	3,149
CatBoost - Random - Select K Best k=15	0,8380	0,4798	0,9982	0,6103	0,9637	0,6334	1,259	0,020	1,279
CatBoost - Random - Select K Best k=10	0,8310	0,4069	0,9977	0,5463	0,9587	0,5806	1,250	0,018	1,268
CatBoost - Random - No Feature Selection	0,7887	0,7179	0,9991	0,7517	0,9560	0,7521	2,361	0,017	2,379
CatBoost - No Resampling - Select K Best k=15	0,7254	0,8803	0,9994	0,7954	0,9689	0,7988	1,199	0,047	1,246
CatBoost - No Resampling - Select K Best k=10	0,7183	0,8870	0,9994	0,7938	0,9689	0,7979	0,955	0,038	0,993
CatBoost - No Resampling - No Feature Selection	0,7324	0,9630	0,9995	0,8320	0,9657	0,8396	2,417	0,045	2,463

Fig. 5. Results of 48 experiments

V. Conclusion

The study evaluated the performance of multiple machine learning models under different resampling techniques and feature selection strategies. The models, including XGBoost, Random Forest, Logistic Regression, and CatBoost, were assessed based on key metrics such as recall, precision, accuracy, F1 score, AUC, and MCC. An overview of the paper's contribution is provided below:

1) The experiments revealed that different resampling strategies significantly influenced the models' ability to handle imbalanced data. Tomek links, SMOTE, random oversampling, and no resampling exhibited distinct impacts on key performance metrics. While Tomek links showcased balanced results across metrics, SMOTE and random oversampling showed mixed outcomes.
2) SMOTE and random oversampling was less effective than Tomek links and without resampling techniques for XGBoost and CatBoost models. This suggests that oversampling might have introduced noise and led to overfitting, particularly for models with high learning capacity.
3) The role of feature selection strategies is examined, with a particular focus on comparing Select K Best models and without feature selection. The importance of feature selection in improving the interpretability and effectiveness of the model is clarified by this investigation.
4) Systematic evaluation of ML models under diverse conditions provided a comprehensive understanding of their strengths and weaknesses in handling imbalanced data sets.
5) Highest recall model was a Logistic Regression model but it also showed lower precision, F1 score and MCC compared to other models. The lower precision implies a higher rate of false positives, indicating that the model tends to predict instances as belonging to the minority class even when they do not. While the high recall of the Logistic Regression model highlights its sensitivity to detecting positive instances, practitioners should carefully consider the specific application requirements.
6) While XGBoost exhibited competitive performance, the Random Forest algorithm demonstrated stability across various metrics, making it a reliable choice for scenarios where stability is more important than computational efficiency.
7) CatBoost models performed similarly to XGBoost but with slightly lower recall and higher training time. This suggests that while offering comparable performance, XGBoost might be a more efficient choice for imbalanced classification.
8) Using customized ML pipeline, the optimal configurations for each model are identified, considering both performance and computational efficiency.
9) The results suggest that models with simplified resampling techniques and feature selection can offer not only competitive performance but also improved time efficiency, contributing to their practical utility in real-world applications.

In conclusion, this study provides valuable insights into the effectiveness of various machine learning models for imbalanced classification of fraudulent transactions while investigating the importance of resampling and feature selection techniques. Expected future areas of research could be in examining the effects of hybrid sampling approaches and other feature selection techniques with more than 15 features. This study has some limitations that should be addressed in future work. Firstly, the data set used here might not be representative of all real-world scenarios, and the results might not generalize to other domains. Secondly, a limited range of hyper parameters and model architectures are only explored. Further research could investigate the impact of different tuning strategies and explore more advanced models for potentially even better performance.

References

[1] A. Mniai, M. Tarik, and K. Jebari, "A novel framework for credit card fraud detection," *IEEE Access*, vol. 11, pp. 112 776–112 786, 2023.

[2] S. K. Hashemi, S. L. Mirtaheri, and S. Greco, "Fraud detection in banking data by machine learning techniques," *IEEE Access*, vol. 11, pp. 3034–3043, 2023.

[3] A. Ruchay, E. Feldman, D. Cherbadzhi, and A. Sokolov, "The imbalanced classification of fraudulent bank transactions using machine learning," *Mathematics*, vol. 11, no. 13, p. 2862, Jun. 2023. [Online]. Available: http://dx.doi.org/10.3390/math11132862

[4] I. D. Mienye and Y. Sun, "A machine learning method with hybrid feature selection for improved credit card fraud detection," *Applied Sciences*, vol. 13, no. 12, p. 7254, Jun. 2023. [Online]. Available: http://dx.doi.org/10.3390/app13127254

[5] S. Makki, Z. Assaghir, Y. Taher, R. Haque, M.-S. Hacid, and H. Zeineddine, "An experimental study with imbalanced classification approaches for credit card fraud detection," *IEEE Access*, vol. 7, pp. 93 010–93 022, 2019.

[6] U. Jabeen, K. Singh, and S. Vats, "Credit card fraud detection scheme using machine learning and synthetic minority oversampling technique (smote)," in *2023 5th International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2023, pp. 122–127.

[7] D. S. Sisodia, N. K. Reddy, and S. Bhandari, "Performance evaluation of class balancing techniques for credit card fraud detection," in *2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, 2017, pp. 2747–2752.

[8] E. Ileberi, Y. Sun, and Z. Wang, "A machine learning based credit card fraud detection using the ga algorithm for feature selection," *Journal of Big Data*, vol. 9, no. 1, p. 24, Feb 2022. [Online]. Available: https://doi.org/10.1186/s40537-022-00573-8

[9] E. Bayhan, A. G. Yavuz, M. A. Güvensan, and M. E. Karsligıl, "The effect of feature selection on credit card fraud detection success," in *2021 29th Signal Processing and Communications Applications Conference (SIU)*, 2021, pp. 1–4.

[10] R. F. Lima and A. C. M. Pereira, "A fraud detection model based on feature selection and undersampling applied to web payment systems," in *2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, vol. 3, 2015, pp. 219–222.

[11] K. Modi and R. Dayma, "Review on fraud detection methods in credit card transactions," in *2017 International Conference on Intelligent Computing and Control (I2C2)*, 2017, pp. 1–5.

[12] R. Bin Sulaiman, V. Schetinin, and P. Sant, "Review of machine learning approach on credit card fraud detection," *Human-Centric Intelligent Systems*, vol. 2, no. 1, pp. 55–68, Jun 2022. [Online]. Available: https://doi.org/10.1007/s44230-022-00004-0

[13] T. Thomas, A. P. Vijayaraghavan, and S. Emmanuel, *Machine learning approaches in cyber security analytics*. Springer, 2020.

[14] A. Mishra and C. Ghorpade, "Credit card fraud detection on the skewed data using various classification and ensemble techniques," in *2018 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, 2018, pp. 1–5.

[15] D. Varmedja, M. Karanovic, S. Sladojevic, M. Arsenovic, and A. Anderla, "Credit card fraud detection - machine learning methods," in *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*, 2019, pp. 1–5.

[16] "Credit card fraud detection dataset," http:https://www.kaggle.com/mlg-ulb/creditcardfraud, accessed: 2023-11-14.

[17] E. F. Swana, W. Doorsamy, and P. Bokoro, "Tomek link and smote approaches for machine fault classification with an imbalanced dataset," *Sensors*, vol. 22, no. 9, p. 3246, Apr. 2022. [Online]. Available: http://dx.doi.org/10.3390/s22093246

[18] K. M. Hasib, M. S. Iqbal, F. M. Shah, J. Al Mahmud, M. H. Popel, M. I. H. Showrov, S. Ahmed, and O. Rahman, "A survey of methods for managing the classification and solution of data imbalance problem," *Journal of Computer Science*, vol. 16, no. 11, p. 1546–1557, Nov. 2020. [Online]. Available: http://dx.doi.org/10.3844/jcssp.2020.1546.1557

[19] D. Elreedy and A. F. Atiya, "A comprehensive analysis of synthetic minority oversampling technique (smote) for handling class imbalance," *Information Sciences*, vol. 505, pp. 32–64, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025519306838

[20] R. Mohammed, J. Rawashdeh, and M. Abdullah, "Machine learning with oversampling and undersampling techniques: Overview study and experimental results," in *2020 11th International Conference on Information and Communication Systems (ICICS)*, 2020, pp. 243–248.

[21] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM Comput. Surv.*, vol. 50, no. 6, dec 2017. [Online]. Available: https://doi.org/10.1145/3136625

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, no. null, p. 2825–2830, nov 2011.

[23] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[24] A. Liaw and M. Wiener, "Classification and regression by randomforest," *Forest*, vol. 23, 11 2001.

[25] S. Xuan, G. Liu, Z. Li, L. Zheng, S. Wang, and C. Jiang, "Random forest for credit card fraud detection," in *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, 2018, pp. 1–6.

[26] D. Jurafsky and J. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 02 2008, vol. 2. [Online]. Available: https://web.stanford.edu/ jurafsky/slp3/5.pdf

[27] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 785–794. [Online]. Available: https://doi.org/10.1145/2939672.2939785

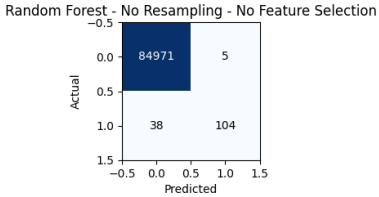
[28] Y. Zhang, J. Tong, Z. Wang, and F. Gao, "Customer transaction fraud detection using xgboost model," in *2020 International Conference on Computer Engineering and Application (ICCEA)*, 2020, pp. 554–558.

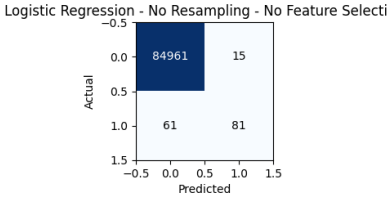
[29] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: Unbiased boosting with categorical features," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS'18. Red Hook, NY, USA: Curran Associates Inc., 2018, p. 6639–6649.

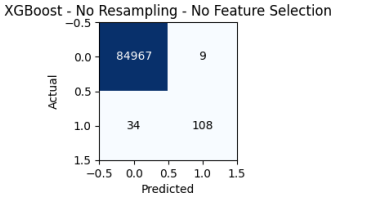
[30] Y. Chen and X. Han, "Catboost for fraud detection in financial transactions," in *2021 IEEE International Conference on Consumer Electronics and Computer Engineering (ICCECE)*, 2021, pp. 176–179.

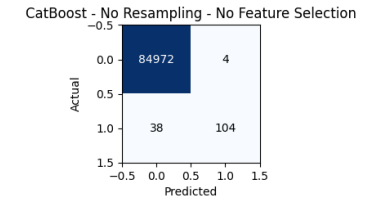
VI. Appendix

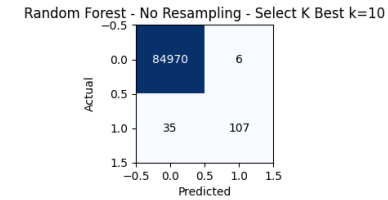
A. Confusion matrices of experiments

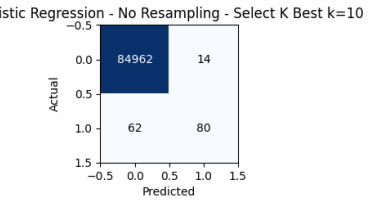


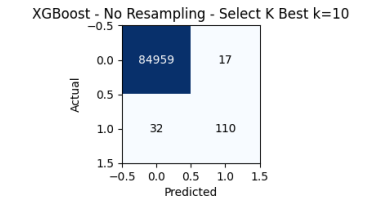


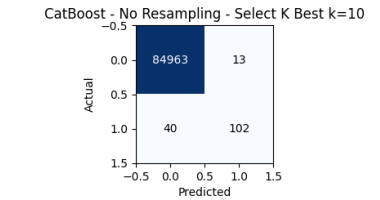


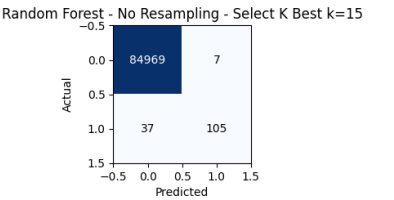


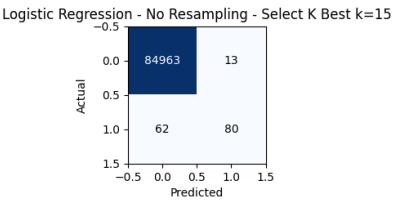


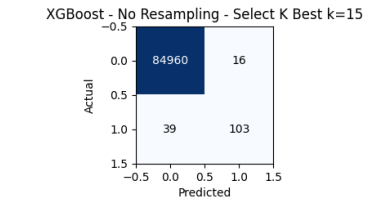


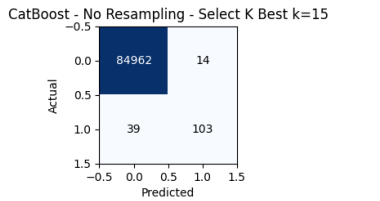


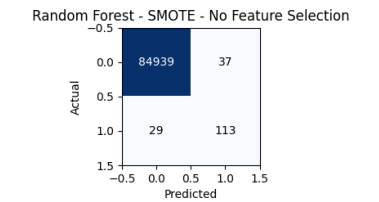


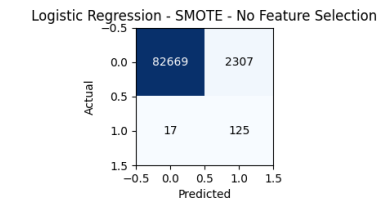


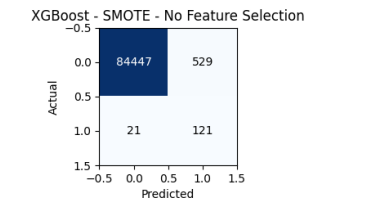


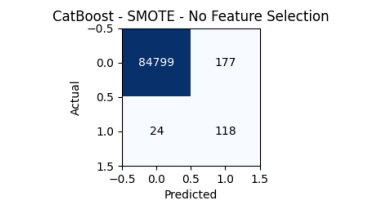


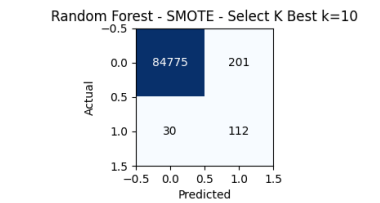


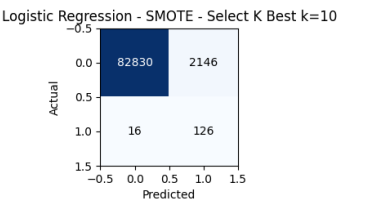


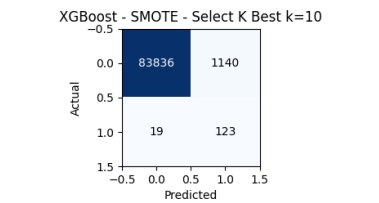


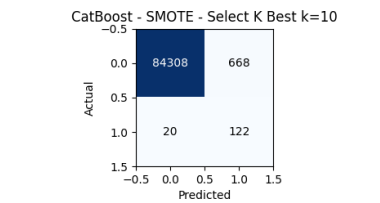


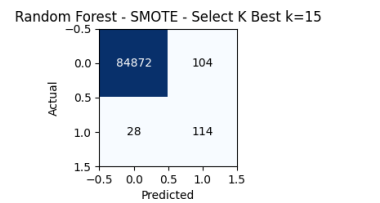


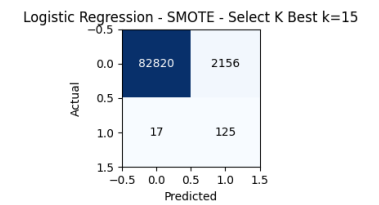


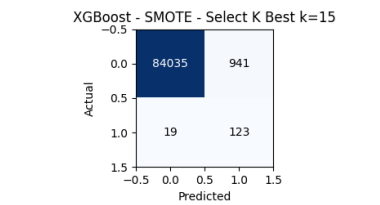


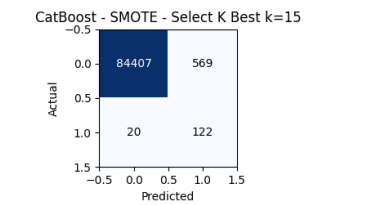






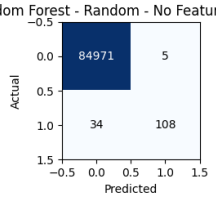




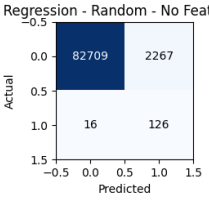


B. *Confusion matrices of experiments, cont*

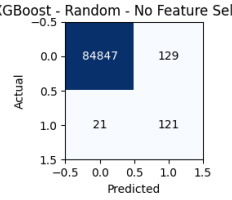
Random Forest - Random - No Feature Selection



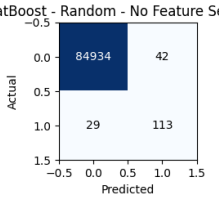
Logistic Regression - Random - No Feature Selection



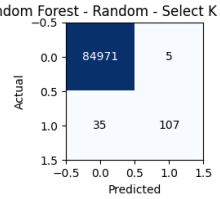
XGBoost - Random - No Feature Selection



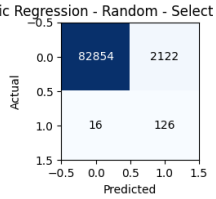
CatBoost - Random - No Feature Selection



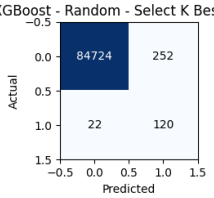
Random Forest - Random - Select K Best k=10



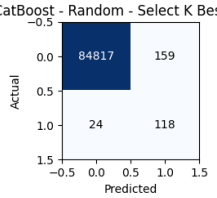
Logistic Regression - Random - Select K Best k=10



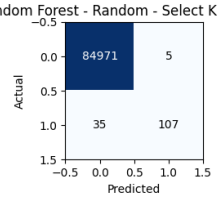
XGBoost - Random - Select K Best k=10



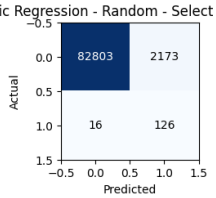
CatBoost - Random - Select K Best k=10



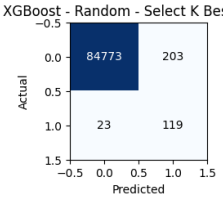
Random Forest - Random - Select K Best k=15



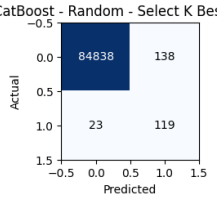
Logistic Regression - Random - Select K Best k=15



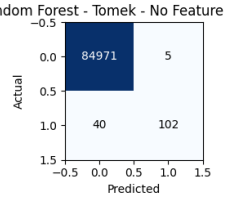
XGBoost - Random - Select K Best k=15



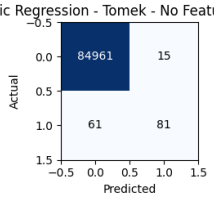
CatBoost - Random - Select K Best k=15



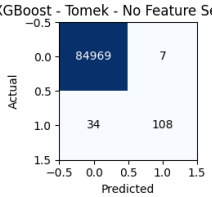
Random Forest - Tomek - No Feature Selection



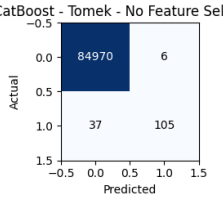
Logistic Regression - Tomek - No Feature Selection



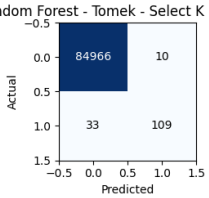
XGBoost - Tomek - No Feature Selection



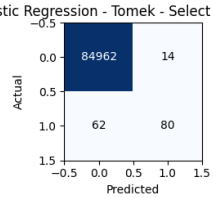
CatBoost - Tomek - No Feature Selection



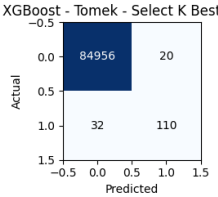
Random Forest - Tomek - Select K Best k=10



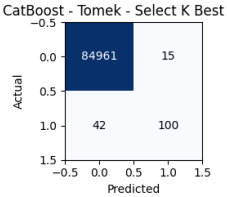
Logistic Regression - Tomek - Select K Best k=10



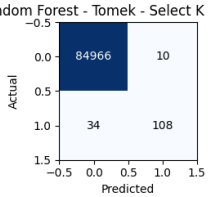
XGBoost - Tomek - Select K Best k=10



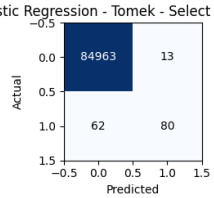
CatBoost - Tomek - Select K Best k=10



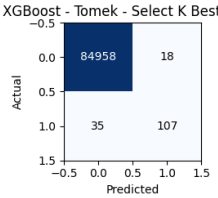
Random Forest - Tomek - Select K Best k=15



Logistic Regression - Tomek - Select K Best k=15



XGBoost - Tomek - Select K Best k=15



CatBoost - Tomek - Select K Best k=15