A NOVEL SAMPLING TECHNIQUE AND GRADIENT BOOSTING TREE-BASED APPROACH FOR CROSS-CHANNEL FRAUD DETECTION

A Thesis

by

Uğur Dolu

Submitted to the
Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Data Science

Özyeğin University
May 2022

A NOVEL SAMPLING TECHNIQUE AND GRADIENT BOOSTING TREE-BASED APPROACH FOR CROSS-CHANNEL FRAUD DETECTION

Approved by:

Assist. Professor Emre Sefer, Advisor
Department of Computer Science
Özyeğin University

Assoc. Professor Okan Örsan Özener
Department of Industrial Engineering
Özyeğin University

Prof. Oğuz Kaynar
Management Information Systems
Sivas Cumhuriyet University

Date Approved: 26 May 2022

To my family...

ABSTRACT

The most recent research on hundreds of financial institutions uncovered that only 26% of them have a team assigned to detect cross-channel fraud. Due to the developing technologies, various fraud techniques have emerged and increased in digital environments. Fraud directly affects customer satisfaction. For instance, only in the UK, the total loss of fraud transactions was £1.26 billion in 2020. In this study, we come up with a Gradient Boosting Tree (GBT)-based approach to efficiently detect cross-channel frauds. As a part of our proposed approach, we developed an algorithm able to generate an optimized training set to train the model and overcome imbalanced data problems. This solution made it easier for the model to understand the concept drift, another major problem arising from changing customer behavior. We boost the performance of our GBT model by integrating additional demographic, economic, and behavioral features as a part of feature engineering. Hyperparameter tuning methods find the best parameters for the model. The cross-channel fraud detection performance of the model is evaluated on a real banking dataset which is highly imbalanced in terms of fraud which is another challenge in the fraud detection problem. We use our trained model to score real-time cross-channel transactions by a leading private bank in Turkey. As a result, our approach can catch almost 75% of total fraud loss in a month with a low false-positive rate and acceptable call count.

ÖZETÇE

Yüzlerce finans kurumu üzerinde yapılan en son araştırmalara göre, bu kurumların sadece %26'sının kanallar arası dolandırıcılığı tespit etmek için atanmış bir ekibe sahip olduğu ortaya çıktı. Gelişen teknolojiler neticesinde dijital ortamlarda çeşitli dolandırıcılık yöntemleri ortaya çıkmıştır ve var olan yöntemler gelişmiştir. Sahte işlemler, direkt olarak müşteri memnuniyetine etki eder. Örneğin, 2020 yılında sadece Birleşik Krallıkta dolandırıcılık işlemlerinden ortaya çıkan toplam kayıp 1.26 milyar £'du. Bu çalışmada, kanallar arası sahtekarlıkları verimli bir şekilde tespit etmek için Gradient Boosting Tree (GBT) tabanlı bir yaklaşım uygulandı. Çalışmada önerilen yaklaşımın bir parçası olarak, modeli eğitmek ve dengesiz veri sorunlarının üstesinden gelmek için optimize edilmiş bir eğitim seti oluşturabilen bir algoritma geliştirdik. Bu çözüm, modelin değişen müşteri davranışından kaynaklanan bir diğer önemli sorun olan kavram kaymasını anlamasını kolaylaştırdı. Öznitelik mühendisliğinin bir parçası olarak var olan özniteliklere ek demografik, ekonomik ve davranışsal öznitelikleri entegre ederek GBT modelinin performansı arttırıldı. Hiper parametre optimizasyon yöntemleri, model için en iyi parametrelerin bulunmasında kullanıldı. Dolandırıcılık tespit problemindeki bir diğer zorluk olan, dolandırıcılık açısından oldukça dengesiz olan gerçek bir bankacılık veri seti üzerinde bu çalışmada geliştirilen kanallar arası dolandırıcılık tespit yönteminin performansı ölçüldü. Türkiye'nin önde gelen özel bankalarından birinin gerçek zamanlı ve tarihsel çapraz kanal işlemlerini puanlamak için eğtimli modelimizi kullanıyoruz. Bu çalışmanın sonucunda uygulanan yaklaşım, düşük bir yanlış pozitif oranıyla dolandırıcılık işlemleri nedeniyle bir ayda gerçekleşen toplam kaybın neredeyse %75'ini yakalayabilir.

ACKNOWLEDGEMENTS

This thesis becomes a reality with the kind support and help of many individuals. I would like to extend my sincere thanks to all of them.

First of all, I would like to say thank you to Assistant Professor Emre Sefer and Associate Professor Okan Örsan Özener for their support and guidance. I would also like to acknowledge and give my warmest thanks to Professor Oğuz Kaynar from my thesis defense jury.

Secondly, I would like to say thank you to Yapi Kredi Technology as my current company. They always supported me during my thesis and master's program. Also, I want to give my thanks to my current team leader Bilge Köroğlu for her valuable support and flexibility during my thesis period. At the same time, I appreciate thanking my former teammate Burak Bastem for his valuable support and time.

Finally, I am extremely grateful to my family; my mother Ayşe Dolu and my father Veysel Dolu, and my sister İpek Nur Dolu for their love, prayers, caring, sacrifices continuous support, and understanding throughout my education life and for preparing me for my future. I would also like to express special thanks to my fiancee Zeynep Atasoy for all the moral support she has given me. Last but not the least, a debt of gratitude is also owed to our friend Murat Turhan, who has been with us throughout the ups and downs of my university life and guided me through it. My thanks and appreciations also go to my colleagues and people who have willingly helped me out with their abilities.

I acknowledge that this thesis is based on a paper that I co-authored with my supervisor and is now being reviewed.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

For banks and financial institutions, fraud is an unavoidable loss. Customers value reliability when choosing a bank to keep their money. Therefore, banks lose reputation and money as a result of fraudulent transactions. Each year, the number of fraudulent transactions and the amount of money lost due to fraud is an upward trend. When fraudulent transactions from around the world are assessed, the financial cost of fraud in 2021 is £4.37 trillion [1], up from £3.89 trillion in 2019 [2]. The method used by swindlers changes over time. Phishing, social engineering, verbal persuasion, and computer viruses like Trojan are the most popular approaches. In 2020, the total loss due to fraudulent transactions in the UK alone was £1.26 billion. Considering the fraud-based transactions, 38% are carried out by remote banking, 45% by credit card, and 16% by social engineering and phishing.

In 2020, the total amount of fraud loss for online and internet banking was £197.3 m. On the other hand, when the first six months of 2020 and 2021 are compared, it is observed that the total loss due to the fraudulent transactions has increased from £79.7 m to £133.4 m, which corresponds to a 67% increase. Internet banking, mobile banking, and telephone banking are the three types of remote banking. Fraud transactions carried out through telephone banking have shown a downward trend over the years. When the first six months of 2020 and 2021 are compared, a 50% decrease was observed in the number of transactions. In 2020, the total financial loss due to social engineering fraudulent methods was £479m. However, the total loss from fraud increased from £207.8m to £355.3m, with a dramatic 70% increase compared to the first six months of 2020 and 2021 [3, 4].

To persuade victim customers, fraudsters prefer to use remote banking channels

along with social engineering techniques. More than 40 million transactions take place each month, according to analysis from our real dataset from a private bank in Turkey. Only about 800 transactions of monthly data, nonetheless, are flagged as fraudulent. The ratio of fraud transactions is approximately 0.002. In addition to it, several academic studies have used synthetic or static data. This type of work isn't appropriate for real-world scenarios. The fundamental problem is that fraud methods and channel client behaviors change over time, yet the model is trained using only synthetic or static data [5].

The main complication in the channel fraud problem is the extremely imbalanced class distribution and concept drift. One of the main objectives of this study is to overcome these two problems. The transactions contained in the highly imbalanced data set in terms of class distribution may depend on accounts or channels. Although these transactions include very few fraudulent transactions, they have many legitimate monetary transactions. Therefore, the resulting problem is called unbalanced class distribution. To solve this problem, various methods were employed, namely undersampling [6] and oversampling [7]. However, these techniques are still problematic because the underlying dataset is exceedingly imbalanced, and instances of the dataset individually carry important information (such as transactions belonging to the same account or customer).

In this study, we experimented with different transaction-based, account-based, and customer-based equalization techniques to alleviate the imbalanced data problem. Many traditional machine learning algorithms can be easily broken by the majority class in imbalanced data. The algorithm will generate scores according to the majority class and lead to high misclassification rates in the minority class. We introduce boosting models such as gradient boosting trees (GBTs) to overcome this issue. We employ GBT in our study. The key idea is to ensemble weak decision trees, a commonly used machine learning method for binary classification on imbalanced data. In summary, the contributions of this study are as follows:

- Our approach supports multiple transaction channels and is capable of detecting frauds between cross-channels.
- Our approach employs feature engineering, concatenating transaction details with customers' demographic and financial information.
- Our approach includes hyper parameter optimization and selection of an appropriate GBT model.
- Our approach does not need any additional maintenance, the auto train algorithm will generate the training set from historical data and will retrain the model. Our method will be applicable in realistic scenarios since: 1- An automated training mechanism helps to adapt to new behaviors of account holders and fraudsters, 2- Real data has all fraudsters and customer habits, so our approach will not struggle with aging, 3- We generate training sets according to sampling method, which balances out the data and improves the performance of the model.

This study is organized as follows: Chapter 2 discusses the previous works. Chapter 3 gives information about the development environment, data set, and study. In addition, it includes the methodology of generating training sets, feature engineering, encoding, and algorithms. The performance of the techniques is given in Chapter 4 in detail, which is followed by the conclusion in Chapter 5.

CHAPTER II

PREVIOUS WORK

The cross-channel fraud detection systems have to deal with two main issues: binary classification on an imbalanced dataset and handling the concept drift. In this chapter, in addition to providing a brief overview of the aforementioned problems, we also go through some of the existing applicable fraud detection algorithms for various types of transactions, such as credit cards. Although there are many studies and academic literature on credit card fraud, there are no academic studies specific to channel fraud detection. For this reason, we approached the problem not only in terms of cross-channel fraud but also by looking at all fraud methods.

2.1 Similar Products on Market

Many commercial solutions have been developed to protect financial institutions and their clients from swindlers, and many academic studies have been established in this field. FICO is a global leader in this sector, offering a variety of software solutions. The majority of products on the market use rule-based algorithms and solutions to detect fraudulent transactions. The rule-based algorithms are effective, although they are insufficient to catch dynamically changing user behaviors and fraud strategies that change in connection with these conduct. The rule-based systems require periodic maintenance to ensure continued success in fraud detection. Within the scope of this maintenance, the relevant business units analyze customer and fraudulent behavior and update their rules in line with new analyzes, which requires a significant amount of human effort. Additionally, the most recent research shows that only 26% of all financial institutions have a team allocated to detect cross-channel frauds [8].

In this study, a GBT-based model has been developed that is not affected by

dynamically changing behaviors, does not need periodic maintenance, and supports rule-based algorithms.

2.2 Concept Drift

Only a few studies have focused on the concept drift problem in fraud detection tasks in the literature. For example, the article [9] proposed a method based on a sliding window and an ensemble of classifiers to overcome this credit card fraud detection problem. In another study, a fraud detection system based on a concept drift management approach has been presented to retain new concepts on the transaction streams using the cardholder profiles [10]. However, because to the lack of real benchmark datasets, both of these research studies were tested on synthetic datasets.

This study worked on an approach that will automatically generate a training set. Details on this approach are described in the following Chapters. We tried to adapt the concept shift to the dataset in the most effective way. The conventional fraud detection systems are developed according to existing fraud methods and cannot quickly adapt to new techniques.

2.3 Imbalanced Data

Furthermore, learning from unbalanced data for binary or multi-class classification is a common difficulty in real-world problems where the models are trained on a dataset with imbalanced distribution of classes [11]. Imbalanced data, in real-world applications, suffer from class imbalance problems. A predictive model trained on unbalanced data is more likely to predict majority class samples while misclassifying minority class samples correctly. Due to this misclassification problem, many studies in this field have tried to find solutions with different approaches. The purpose of these studies is to propose different strategies to generate subsets of the training set for binary classification in fraud detection.

For instance, the research [12] shows how class distribution in the training set

affects credit card fraud detection models' prediction performance. In their experiments, training sets with varying the number of fraudulent transaction distributions from 10% to 90% for each month were utilized to generate a meta-learning model. The training set with $50\% - 50\%$ equal class distribution achieved the most appropriate performance in reducing prediction loss and training time. The study [13] showed how the performance improves when the majority class is undersampled on the training set. On the other hand, some studies have employed account-based undersampling methods to construct the training set [14].

In this work, we also compared transaction- and account-level undersampling strategies by equalizing the counts of fraudulent and legitimate transactions while making sure that all the transactions of both types cover the same time interval in the training sets. Similarly, we approach the highly imbalanced class distribution problem with undersampling methods. However, instead of randomly undersampling data, we generate a training set by equalizing fraudulent and legitimate credit card counts, and transactions of each credit card cover the equal time range [15]. We adapted this solution to the channel fraud detection problem, trying to equalize the number of fraudulent and legitimate accounts instead of credit card counts. Additionally, this structure was tested with different ratios between a swindler and reliable account counts.

2.4 Feature Engineering

Feature Engineering, in real-time transactions, customer behaviors are crucial, their habits can evolve over time. To minimize the error that arises from behaviors [16], the features such as "familiar device", "is account whitelisted or blacklisted", and "is device used for another account before" are integrated to the main transaction dataset. Additionally, to detect more accurate behavioral data, we have also included historical maximum and minimum credit scores, customer age, and financial age [17]. Furthermore, combining transactions with commercial or individual customer-type flags has improved the model's prediction performance

in a positive way.

2.5 Algorithm

Cross channel fraud detection framework [5] proposed a graph analysis extracted from realized transactions in real-time. Their research examines the shortest paths between transactions and strongly-connected components in the transaction graph to detect fraudulent transactions. Another work applies a recurrent neural network [18] for cross-channel fraud detection. A study tests performance differences between two random forest models in credit card fraud detection problems [19]. One model includes behavioral features while the other is not, and the study evaluates performance differences between models. In the literature, there are not many examples of GBTs to detect cross-channel frauds. In this paper, we come up with a GBT model capable of generating a prediction score for each transaction from internet banking, mobile banking, ATM, and telephone banking. We mainly employ GBTs as they have outperformed the other boosting algorithms.

It is extremely important to optimize the chosen algorithm as well as the algorithm selection. In this context, we applied hyper-parameter optimization methods. The Optuna [20] is a handy framework with different parameter optimization options. Among these optimization options, it can make multiple attempts faster than traditional parameter optimization methods with features such as early pruning of parameter optimization according to outputs of various algorithms.

CHAPTER III

METHODOLOGY

In this Chapter, we first give an overview of our study by defining its components. Then we go over our generating training set algorithm, feature engineering, encoding mechanism, and GBT-based training algorithm.

3.1 Development Environment

The configuration of the development environment we used to develop this project is as follows: a CPU with 80 cores, around 700-800 GB of RAM, and approximately 2 TB of SSD drive. Red Hat Enterprise Linux is used as the operating system on this device. Since the cross-channel fraud detection problem requires big data environments, we used NoSQL [21] database solutions to store, analyze, and access data with low latency. NoSQL solutions also allowed us to access data with low latency. This study used Redis as a key-value store and MongoDB as a document database from NoSQL methods.

Redis [22] is used for lookup tables and aggregation studies. Clustered Redis is very responsive and has a low response time. During preprocessing, Redis reduces calculation time while matching customer accounts and customer ids from lookup tables. MongoDB [23] is used for storing main and additional data. During preprocessing, additional data for transactions are getting from MongoDB.

As a scripting language, we used Python. We enriched Python with multiprocessing and threading. Also, NumPy and Pandas are used for data and mathematical operations. NumPy is used to support multiprocessing because passing data frames between cores consume time.

3.2 Overview of Study

This study has two main phases. These are offline algorithm training and real-time fraud detection on incoming transactions. First, a training set is generated using the methods and algorithms described in the section 3.3 for the offline training algorithm. After the training set is created by the algorithm, the GBT model is trained with this specific training set. In the real-time detection phase, transactions and metadata are fed to apply pre-processing steps, i.e., data sanitization and handling the missing values. Then, if the customer number is known, additional attributes of the customer are added to the main dataset. If it is not, the account number is used for matching the customer number first. Then, additional features are added. After the customer number mapping phase, the pipeline will add the demographic, economic, and behavioral attributes to the core data for the receiver and the sender customers. The encoder will handle the categorical string values, and convert them to the encoded integer to feed trained GBT model. Finally, the model generates a score for each transaction to classify them as fraud or not. The real-time detection pipeline is illustrated in Figure 1.

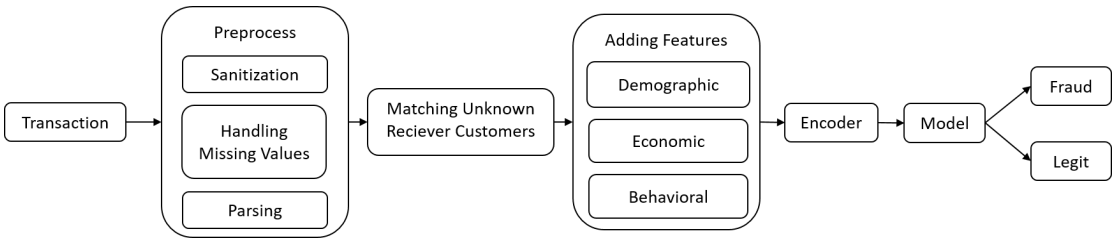

Figure 1: A flow of a transaction in the study

3.2.1 Dataset Description

Main data has more than one hundred columns and features. After the unique columns are dropped from the data frame, a sample of significant columns and their descriptions are given in Table 2. Columns are used to add additional features to the dataset, and their explanations are detailed in Table 1. These columns are used to match customer data.

Table 1: Columns to match customers' monetary transactions and additional data

Column	Description
Sender Client No	Used for matching extra features.
Sender Account No	Used for matching senders' client no.
Receiver Client No	Used for matching extra features
Receiver Account No	Used for matching senders' client no.

Some of the essential features used in the model are given in Table 2. Amount features are fed to the model as is, and other features need encoding for the model.

Table 2: Key features and descriptions of the model

Feature	Description
Access Channel	Accessing Channel for Sender (ATM, Mobile, etc.)
Amount	Transaction amount
In-Bank Transfer	Is the transfer between accounts belonging to the same bank?
Transaction Fee	Transaction fee amount
Transaction Channel	Transaction channel (EFT, Transfer, etc.)
Sender Account Branch	Senders' branch information.
Receiver Account Branch	Receivers' branch information.
International Transfer Flag	Is the international transfer?
Transaction Type	Type of transaction (Tax, Card Payment, etc.)
SMS Notification Flag	Is SMS Notification permitted?
Connection Type	Connection Type (WiFi, Mobile Data, etc)
Device Whitelist Flag	Is the device whitelisted?
ATM Flag	Is an ATM transfer?
IMEI Valid Flag	Is IMEI valid?
Familiar Account Flag	Is receiver account familiar?
Login Duration	The login duration during transfer
Mobile App Version	The version of the mobile app
Mobile Device Brand	Brand of the mobile device
Mobile Device Model	Model of the mobile device
Same Device Last 30 Day	Has the same device been used in the last 30 days?

3.2.2 Preparing the Data

After accessing the main data, the standard Extract Transform Load (ETL) processes are applied to the data. The data is ready to use in the pipeline. First of all, some columns of main data contain more than one feature. At the beginning of the pipeline, these columns will split and parse and create necessary fields. After the combined fields are parsed, the standard sanitization and handling missing values methods are applied to the dataset.

Secondly, the pipeline needs to match customer numbers from customers' account numbers if customer numbers are not available. Customer numbers are used to match customers' additional demographic, economic, and behavioral features to monetary transactions. The pipeline will extract and add those features to the main data.

Finally, all preprocessing steps are completed, and the concatenated dataset is ready for model input as a training or test set. Only string values need to encode for the model structure.

3.3 Generating Training Sets

We experimented with two types of data balancing mechanisms: transaction-level and account-level. In transaction-level balancing, we pick all the fraudulent transactions within a time frame and randomly select the same number of transactions from the legitimate ones. The swindlers and legitimate transactions belong to the same customers who have been a sender or receiver of a fraudulent transaction at least once during the training set.

For the account-level balancing, we employ the technique used by study [11]. Suppose at least one transaction is identified as a fraudulent transaction in a given time range. In that case, we get all transactions of the receiver and sender of the fraudulent transaction and label them as fraudulent accounts. If any customer has no fraudulent transaction, we denote the sender and receiver customers as legitimate accounts. First, we started by equalizing the fraudulent and legitimate account numbers to create a training set from their transactions. Even though it still results in a high-class imbalance, it preserves the patterns while decreasing a lot of initial imbalance [11].

This study performed tests by equating fraudulent and legitimate accounts at different rates to reach more detailed results. Because higher sampling of legitimate accounts will increase the diversity of legitimate transactions in the training

set, and as a result, it is believed to improve model performance on detecting non-fraud transactions. The generation of a training set is automated in our study to create new training sets and train new models by itself as time passes. Since the performance of a model decreases over time due to concept drift caused by a change in the behavior of account holders and/or fraudsters, this automation mechanism helps to adapt to new behaviors and prevents a dramatic decrease in prediction performance.

3.4 Feature Engineering

The metadata of any cross-channel transaction contains numerical, categorical, and textual information. All related data is embedded in a feature vector and this vector is fed to the model to get the prediction for the transaction. The textual fields may contain more than one different value. These fields are parsed to extract features. For instance, a feature, which indicates belonging to blacklist or white-list of a customer, is fed the model with the value "1" and "0". The parsed fields refer to "1" for blacklist and "0" for white-list. Another example, a field containing "AZ" means, "A" for a device like ATM and "Z" for login duration which indicates more than 29 minutes. The Model uses numerical fields as they are but they are encoded as categorical fields. More detail for encoding is given in Section 3.5.

In addition, demographic, economic and behavioral features for receiver and sender of a transaction are added. For instance, features like customers' age, financial age are some of the features which belong to demographic information. Credit scores, client types, historical payment habits are also included to estimate the fraud probability of a transaction. They are called as behavioral features. After we determine the preliminary model features, the GBT model is trained with behavioral, economic and demographic features. We also studied the feature importance.

As a result of Feature Engineering studies, additional attributes were added to

the metadata of transactions to determine customer behaviors better and improve the model's performance. Additional features are detailed in Table 3.

Table 3: Demographic, economic, and behavioral features and explanations

Feature	Description
Client Type	Type of client(Bank employee, Individual, Commercial)
Client Segment Type	Client segment type(Commercial/Individual/SME/ME)
Customer Age	Age of customer
Financial Age	The financial age of the client.
Max Last Month Credit Score	Highest credit score in the last month
Max 3 Month Credit Score	Highest credit score in the last three month
Max 6 Month Credit Score	Highest credit score in the last six month
Max 12 Month Credit Score	Highest credit score in the last twelve month
Min Last Month Credit Score	Lowest credit score in the last months
Min 3 Month Credit Score	Lowest credit score in the last three months
Min 6 Month Credit Score	Lowest credit score in the last six months
Min 12 Month Credit Score	Lowest credit score in the last twelve months
SIM Block Day	How many days have passed since the last SIM block?

3.5 Encoding

Transaction data contains both text and numeric values, as previously stated. Numeric values can be directly fed into the model, while text values must be encoded. Label encoder preferred as the coding method. A dictionary, which contains a mapping for text and its corresponding encoded value, is created from the history for each encoded feature of the dataset. The feature values are arranged in ascending order in the encoded list. For the real-time encoding, the value -9999 is added as a response to all unseen values. This ensures that a real-time scoring process remains consistent.

3.6 Algorithm

In order to detect fraud in cross-channel transactions, we needed to choose an appropriate model. Based on the models used in previous credit card fraud and other fraud detection methods, we thought tree-based decision models were the most suitable option for our problem. We tested random forest, isolation forest, and XGBoost algorithms from tree-based model algorithms.

The results of the models are detailed in Table 4. Among the points, we paid

attention to in model selection is its high RTVDR value and its high ADR at low AFPR. At the same time, we preferred to keep the NFTR value as low as possible.Detailed information on model evaluation metrics is described in Section 4.2.

Table 4: Comparison of base models in the portfolio

Model	ADR	AFPR	RTVDR	NFTR
Random Forest	0.63	229.51	0.73	0.003
XGBoost	0.53	144.62	0.72	0.001
Isolation Forest	0.34	3013.69	0.63	0.08

After selecting the appropriate model, hyper-parameter optimization is performed to set model parameters according to cross channel fraud detection problem. Optuna is a framework for hyper-parameter optimization and is chosen to use in our study. It has many pruning algorithms to stop training steps early. Thanks to that, Optuna can test more parameter sets than other methods at an equal time.

CHAPTER IV

EXPERIMENTS

Our proposed approach is focused on generating prediction scores for cross-channel transactions using GBT-based models and identifying fraudulent transactions from legitimate ones. Various decision tree algorithms are tested during the research period of our study. However, XGBoost [24] beats Random Forest, and also other classifiers. The following chapter describes the experiments and gives detailed results of our research.

4.1 Experimental Setup

First of all, all experiments are performed on the real data set of a leading private bank in Turkey. The data set is divided into two main parts: the training and test sets. The test set data includes six-month channel transactions from May 2021 to October 2021. During the test set period, approximately 4000 transactions out of over 260 million were fraudulent. Table 5 provides a detailed explanation for each month in the test set.

Table 5: The legitimate and fraudulent transaction counts of test sets used in experiments

Test Set Name	Trx CNT	Fraud CNT	Ratio
2021-05	40013025	644	0.0016%
2021-06	40867466	670	0.0016%
2021-07	42121304	690	0.0016%
2021-08	44368519	951	0.0021%
2021-09	46337966	826	0.0018%
2021-10	48144671	863	0.0018%

Secondly, there are two critical points while creating the training set in this study. These important points are the time interval of the training set and the ratio of legitimate accounts to fraudster ones. The dataset we reserved for education has

two years of historical data. To best evaluate this historical data, we needed to best determine the time span of the training set. If we include too many retrospective operations in the training set, the behaviors learned by the model may differ from the behaviors in the test set. For this reason, the performance differences were measured after the model algorithms formed from the training sets of different time intervals were tested on the test sets. The study results on these training sets are given in Section 4.3.1.1. Another point of considering the training set is the ratio between legitimate and fraudster customers. The number of fraudulent and legitimate customers at different rates affects the model's performance. Since under-sampling was made in imbalanced data, training sets that included healthy and fraudulent customers at various rates were tested in the model algorithm. The detailed results of the study are given in Section 4.3.1.2.

4.2 Evaluation Metrics

In most regression and classification problems, the error rate is the most commonly preferred success measurement metric to compare the performances of different algorithms. However, the error rate calculation may fail to provide the correct performance measure in such imbalanced datasets. In addition, the minimization of the error rate during learning may not improve Wilcoxon-Mann-Whitney statistics (or AUC scores). Therefore, AUC measures such as the area under the receiver operating characteristics curve (AUROC) is better-suited evaluation metric of binary classification with imbalanced data.

Similar to the market leader as FICO, we used Account Detection Rate (ADR), Real-Time Value Detection Rate (RTVDR), and Non-Fraud Transaction Review Rate (NFTR) in our evaluation metrics [25]. The ADR indicates the rate of identified swindler accounts compared to total fraud accounts. The RTVDR refers to the rate of detected fraud amount compared to the total fraud amount. The NFTR is crucial because it directly expresses the rate of non-fraud transactions which are marked as fraud.

4.3 Results

This section includes results and charts from more than 250 experiments conducted throughout the study. These experiments were conducted on real banking data that is highly imbalanced and dynamically changing. The inferences from the experiments are conveyed in a refined and precise way. The performance of the algorithm, which emerged as a result of all studies, is given in Section 4.3.6.

4.3.1 Generating Training Sets

There are approximately two years of historical data that can be used for the test set in the study. In order to determine the training set to be used in the model, studies were carried out to find the best training set interval and the optimum class distribution ratio. The relevant results are detailed in the following subsections. As a result of these studies, an algorithm was developed that generates the training set used in the model.

4.3.1.1 Training Set Time Span

The training set consisting of too much historical data affects the model performance. We carried out this study to determine the most suitable training set interval for the developed algorithm. First, we fixed the fraud and legitimate customer ratio to 1x and trained models with training sets with different time intervals. In short, we chose one healthy customer for each customer involved in the fraudulent transaction, and we received all the transactions of this customer group that we created within the specified time period. Detailed information about the training sets produced in this study is given in Table 6.

Table 6: Information about the training sets used in the study

Set ID	Trx CNT	Fraud CNT	Ratio	Start Year Month	End Year Month	Duration
A	960001	8072	0.8408%	2020-06	2021-05	11 Month
B	1101890	8769	0.7958%	2020-05	2021-05	12 Month
C	1253876	9403	0.7499%	2020-04	2021-05	13 Month
D	1373276	9832	0.7159%	2020-03	2021-05	14 Month
E	1486651	10099	0.6739%	2020-02	2021-05	15 Month
F	1634989	10444	0.6387%	2020-01	2021-05	16 Month
G	1745335	10444	0.5983%	2019-12	2021-05	17 Month
H	1892329	10444	0.5519%	2019-11	2021-05	18 Month
I	2006358	10444	0.5205%	2019-10	2021-05	19 Month
J	2070217	10444	0.5044%	2019-09	2021-05	20 Month
K	2095193	10444	0.4984%	2019-08	2021-05	21 Month

Many experiments were performed to determine the time interval of the training set. The summary results of these experiments are detailed in Figure 2 below. As can be understood from these results, the training sets with 16-month and 17-month intervals marked as squares and triangles gave the best results.

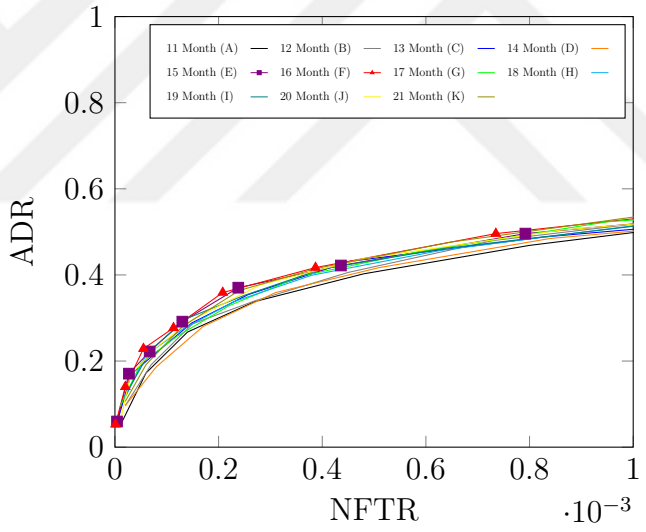

Figure 2: Comparison of effects of various training sets on models' performance

The model's overall performance in detecting fraudulent transactions is given in Figure 3. This model is trained with 16 months training set. The training set details are shown in Table 6 under set id F.

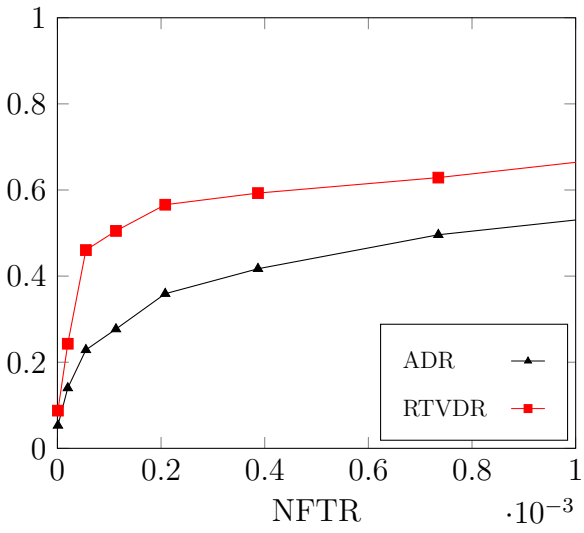

Figure 3: The performance in May 2021 of the model trained with 16 months training set

These results show that low NFTR has a higher ratio of ADR and RTVDR. Instead of using a model trained with a random training set, the model trained with a training set with a 16 months interval performs better. Therefore, we determined the training set interval as 16 months in our study.

4.3.1.2 Training Set Ratio

After finding the optimal time interval for the training set due to the experiments, this study was applied to find the optimum ratio between fraudulent and legitimate customers. Under-sampling is done while generating the training set. While creating the under-sampling algorithm, we divided the accounts that perform channel transactions into two main parts, fraudulent accounts, and legitimate accounts. If a customer's transaction is tagged as a fraudulent transaction, we label that customer as a swindler. If a customer's transaction is not marked as fraudulent, that customer is clean and legit. A pool of fraudulent and legitimate customers is created for the specified time period, and customer selection is made according to the rate to be tried. Detailed information about the training sets produced in this study is given in Table 7.

Table 7: The ratio and count comparison between legitimate and fraudulent transactions in training sets

Set ID	Trx CNT	Fraud CNT	Ratio	Cust. CNT	Fraud Cust. CNT
0.4x	875806	6364	0.7266%	8941	6260
0.66x	973039	6363	0.6539%	10431	6259
1x	1140574	6366	0.5581%	12518	6262
1.5x	1398754	6370	0.4554%	15645	6265
2.33x	1769689	6368	0.3598%	20861	6264
3x	2109139	6370	0.302%	25035	6266
4x	2572307	6376	0.2478%	31294	6272
5.66x	3346256	6385	0.1894%	41719	6281
9x	4938697	6389	0.1293%	62572	6285
19x	9698653	6428	0.0627%	125101	6322
25x	12459404	6427	0.0515%	162608	6321
30x	14779718	6448	0.0436%	193843	6343
40x	19554781	6449	0.0329%	256321	6345
50x	24289872	6474	0.0266%	318738	6370

Experiments were conducted with training sets with many different legitimate and fraudulent customer ratios. The training set time interval in all training sets was 12 months to make good comparisons. The first part of the comparison, which has eight different ratios, is given in Figure 4.

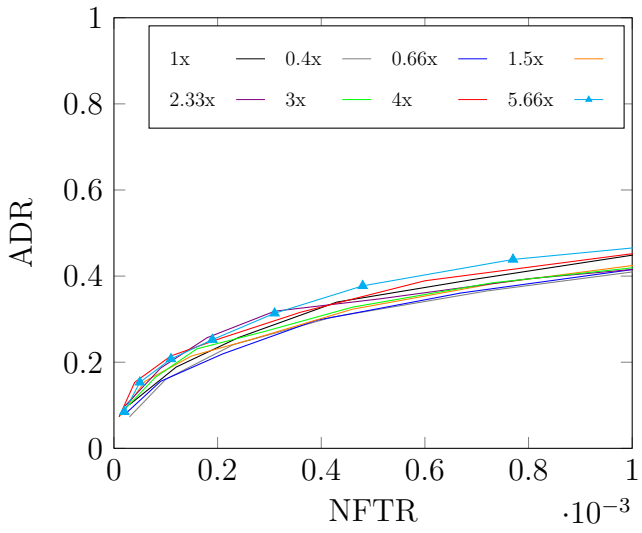

Figure 4: Comparison of effects of various training sets with different legitimate and fraudulent transaction ratios on models' performance

The second part of the comparison which has six different ratios is given in Figure 5. Figure 4's best result, ratio 5.66x, is added to the comparison in this chart and marked with a triangle. The ratio 19x marked with a square has higher ADR at the same level of NFTR.

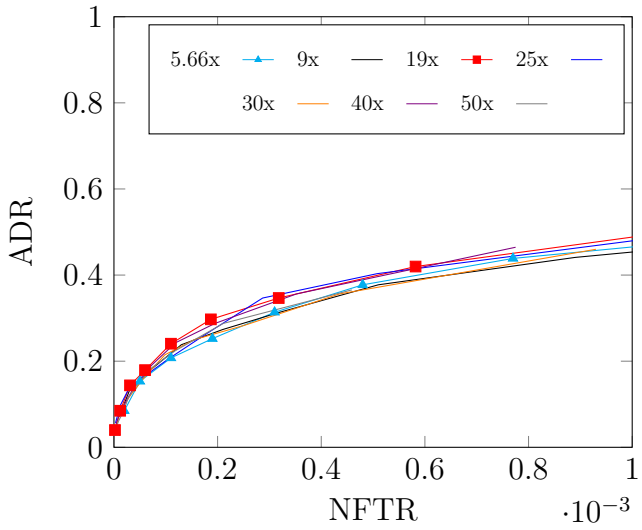

Figure 5: Comparison of effects of various training sets with different legitimate and fraudulent transaction ratios between 5x to 50x on models' performance

The fraud detection performance of the model trained with a 19x ratio for test set May 2021 is given in Figure 6.

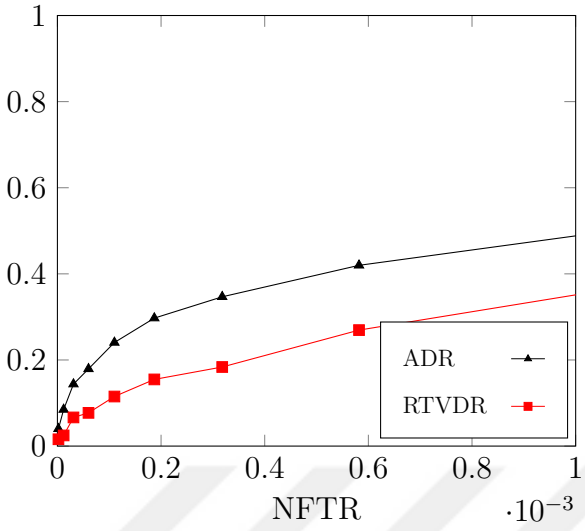

Figure 6: The May 2021 performance of the model trained with the 19x ratio training set

The training interval was fixed at 12 months, and various training sets with different legitimate and fraudulent customer ratios were created. Therefore, in the later stages of the study, we decided to build our training set containing 19 times more legitimate customers than fraudulent customers with a training interval of 16 months.

4.3.2 Random Forest vs XGBoost

Studies were carried out to determine the most appropriate algorithm for the model infrastructure. In the study, random forest and XGBoost, which are tree-based algorithms, were tried. Also, isolation forest was tested, a slightly different algorithm, but it did not get a result that could compete with random forest or XGBoost. As we learned from previous credit card fraud detection studies, we predicted that the XGBoost algorithm would yield better results than the Random Forest algorithm. The results of our experiment were in parallel with our inference.

The same training set was used to train models based on XGBoost and Random forest algorithm. Those models were tested to find the appropriate model

algorithm for fraud detection on cross-channel transactions. The results shown by both models for the May 2021 test set are given in Figure 7. It has been observed that the model based on XGBoost is more successful than the Random Forest model.

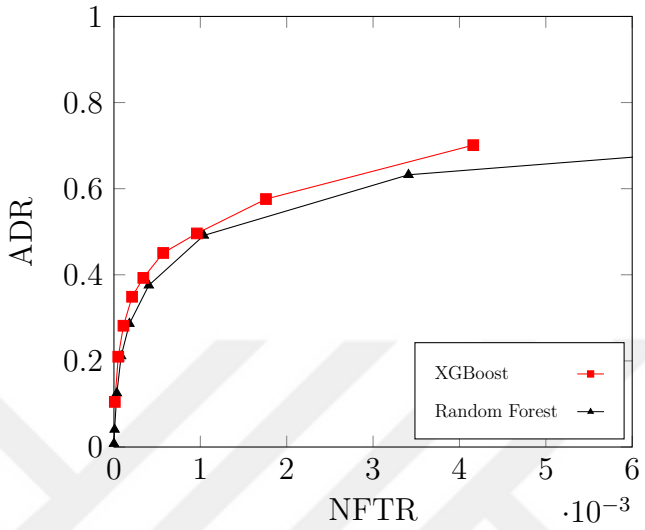

Figure 7: The XGBoost and Random Forest models' performance comparison in May 2021 test set

The same comparison for the June 2021 test set is noted in Figure 8. As can be seen, the performance of the XGBoost-based model continues in the same way.

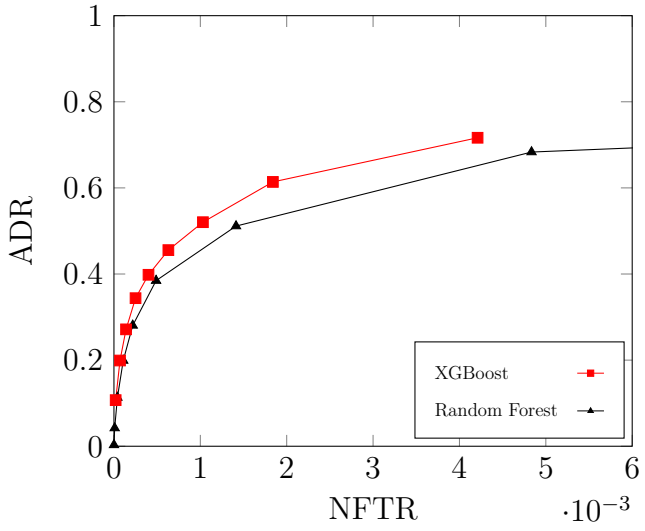

Figure 8: The XGBoost and Random Forest models' performance comparison in June 2021 test set

The results for the July 2021 test set are shown in Figure 9. The model based on XGBoost algorithm has slightly better performance than the Random Forest model.

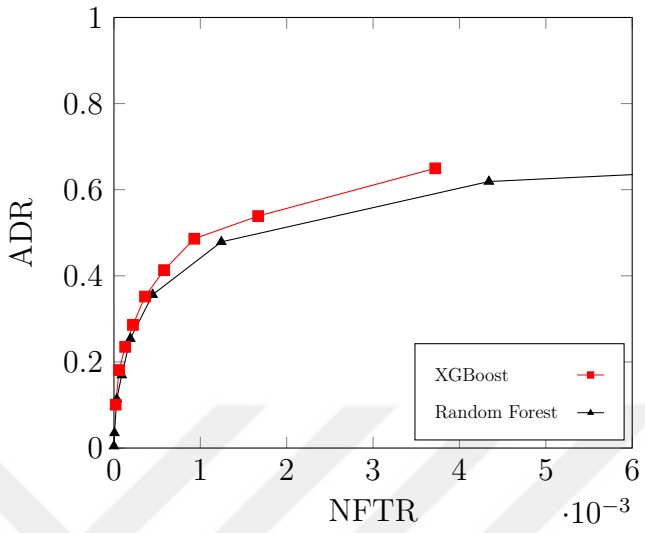

Figure 9: The XGBoost and Random Forest models' performance comparison in July 2021 test set

When all the results are reviewed, the model based on the XGBoost algorithm performs better.

4.3.3 Hyper-parameter Tuning

At First, we used grid search and random search to adjust the parameters used in the high-performance XGBoost model. Traditional hyper-parameter tuning methods could not provide the desired performance in adjusting model parameters and improving the performance of the model. Tuning the correct parameters requires a lot of experimentation, as the cross-channel transaction data is highly imbalanced. In this study, the Optuna framework is used for hyper-parameter tuning. Thanks to that, it can try more parameter sets in the same time than traditional methods.

The output of the hyperparameter tuning framework is given in Table 8. Those are the final values of tuned parameters used in the model.

Table 8: The output of hyperparameter tuning

Parameter	Value
tree_method	exact
objective	binary:logistic
learning_rate	0.07
eval_metric	auc
seed	93
reg_lambda	3
reg_alpha	3

4.3.4 Concept Drift

As mentioned above, customer behaviors also change over time. The model's performance will gradually deteriorate as a result of behavioral changes. To put it another way, the model becomes outmoded. To prevent the aging of a training model and adopt the models' performance to change customer behaviors, the training process should be renewed, and the model algorithm should be updated accordingly. The retrained model will compensate for the performance degradations.

Many training sets were experimented with during the concept shift study. Detailed information about these training sets is given in Table 9.

Table 9: The information about training sets used for proving the Concept Drift

Set ID	Trx CNT	Fraud CNT	Ratio	Start Year Month	End Year Month	Duration
I	1634587	10394	0.6358%	2020-01	2021-05	16 Months
II	1681144	10688	0.6357%	2020-02	2021-06	16 Months
III	1796527	11091	0.6173%	2020-03	2021-07	16 Months
IV	13339287	10394	0.077%	2020-01	2021-05	16 Months
V	15532174	11357	0.0731%	2020-04	2021-08	16 Months

This study tested the performance differences of models trained with training sets with different time intervals generated with the same algorithm.

Compared performance of models trained with training sets I and II on the test set June 2021 is given in Figure 10. The model with the more up-to-date training set performs better.

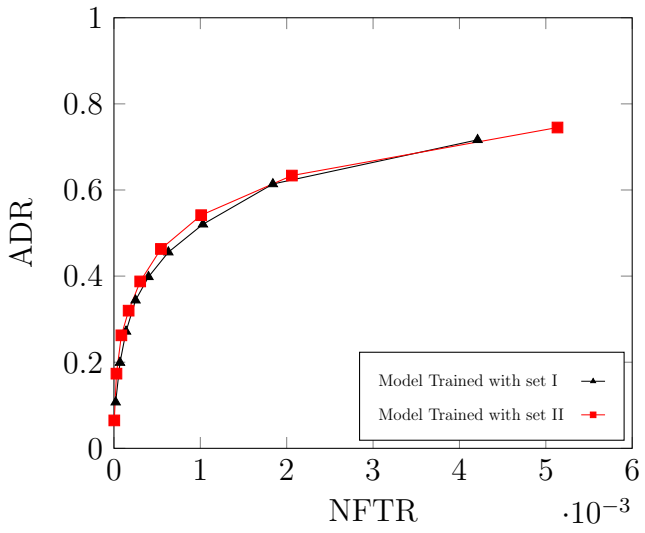

Figure 10: June 2021 performance comparison of models trained with training sets I and II to validate concept drift

Figure 11 compares the performance of models trained using training sets I, II, and III on the test set July 2021. The deterioration in model performance is more pronounced in months when customer behavior changes significantly. For example, disruption is more significant in months when customers are heavily on vacation or have long public holidays. Although the results of all three training sets are close, we notice that newer training sets give better results.

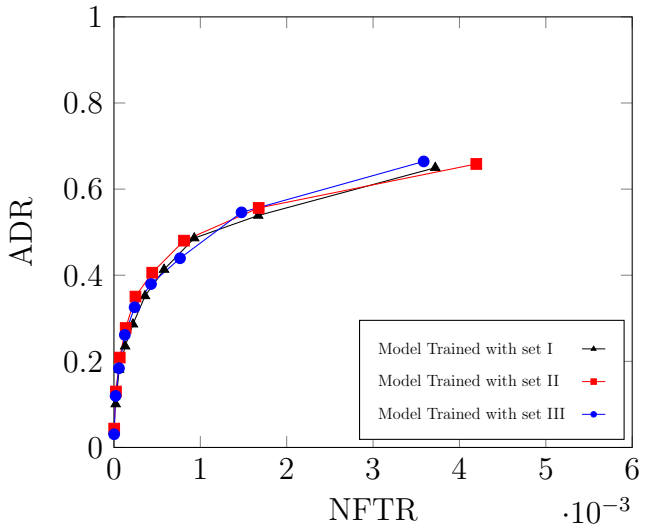

Figure 11: July 2021 performance comparison of models trained with training sets I, II, and III

The performance comparison between two models trained with two different training sets, IV and V, is given in Figure 12. As an expected result, the newer training set has higher ADR at the same level of NFTR. August 2021 data was used as the test set.

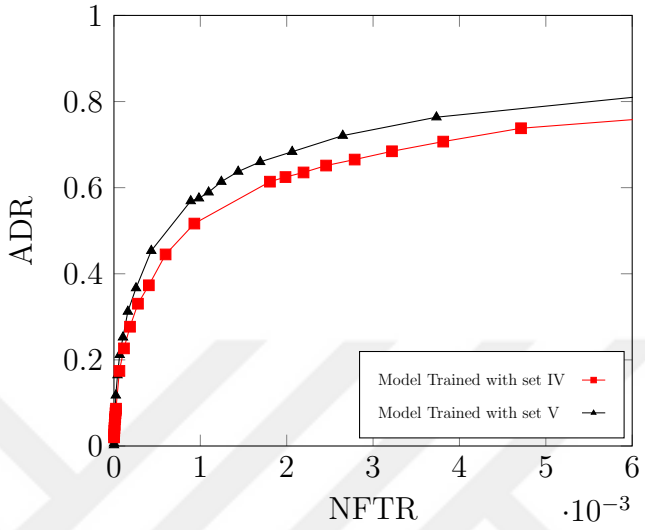

Figure 12: August 2021 performance comparison of models trained with training sets IV and V to prove concept drift

As shown in Figure 13, the model trained with the newer training set gives better results. September 2021 was used as the test set.

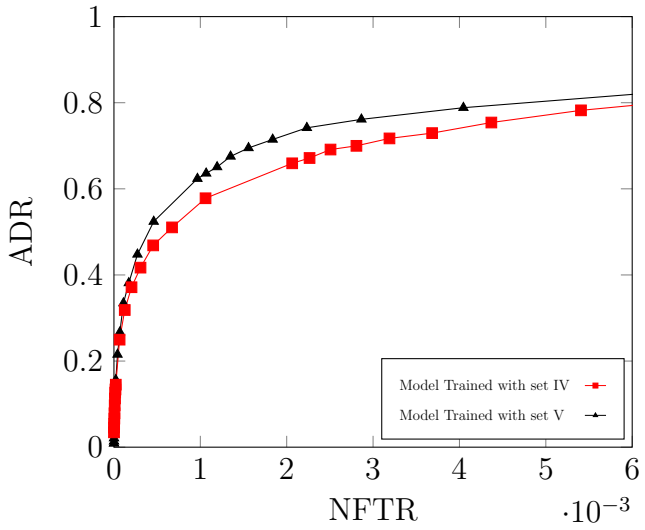

Figure 13: September 2021 performance comparison of models trained with training sets IV and V to prove concept drift

Figure 14 shows the results of the two models on the October 2021 test data.

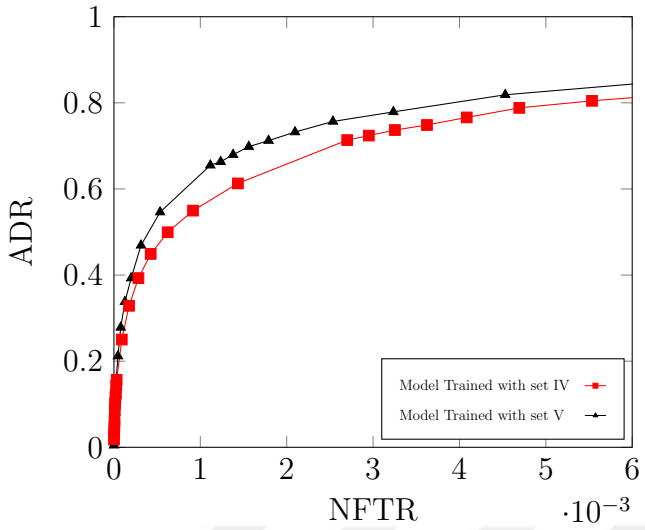

Figure 14: October 2021 performance comparison of models trained with training sets IV and V to prove concept drift

As shown in the figures above, the model trained with the new training dataset outperforms the model acquainted with the old dataset. When we look at the behavior of all models, we get a better ADR performance when the NFTR values are equal.

4.3.5 Feature Engineering

In this project, many studies were carried out within the scope of feature engineering. Many new attributes have been added to enrich the main data frame. At the same time, the impact of existing and added features on the model's performance was measured, and unimportant features were removed from the model.

4.3.5.1 Additional Features

The benefit of demographic, economic, and behavioral features are convenient to characterize customer habits. As an economic feature, we added customers' previous credit scores from the credit bureau of Turkey. Since this score is a general summary of the customer's financial information, we decided to add it to the model. Historical credit scores are included as new features up to 12 months

from the transaction date. This feature is a good indicator of the financial situation of the senders and receivers of transactions. Added features and their descriptions are explained in Table 10.

Table 10: The detailed information about additional features

Feature	Description	Type
Segment Type	Customer segment (SME/ME/etc.)	Demographic
Client Type	Client type of customer (Individual/Commercial)	Demographic
Age	Customer's age	Demographic
Financial Age	Customer's financial age	Demographic
Account Branch	Branch location of account	Demographic
Min t-0 Month Credit Score	The historical minimum credit score of the current month	Economic & Behavioral
Max t-0 Month Credit Score	The historical maximum credit score of the current month	Economic & Behavioral
Min t-1 Month Credit Score	The historical minimum credit score of the last month	Economic & Behavioral
Max t-1 Month Credit Score	The historical maximum credit score of the last month	Economic & Behavioral
Min t-3 Month Credit Score	The historical minimum credit score of the last three months	Economic & Behavioral
Max t-3 Month Credit Score	The historical maximum credit score of the last three months	Economic & Behavioral
Min t-6 Month Credit Score	The historical minimum credit score of the last six months	Economic & Behavioral
Max t-6 Month Credit Score	The historical maximum credit score of the last six months	Economic & Behavioral
Min t-12 Month Credit Score	The historical minimum credit score of the last twelve months	Economic & Behavioral
Max t-12 Month Credit Score	The historical maximum credit score of the last twelve months	Economic & Behavioral

The detailed results and proof can be found in Figure 15.

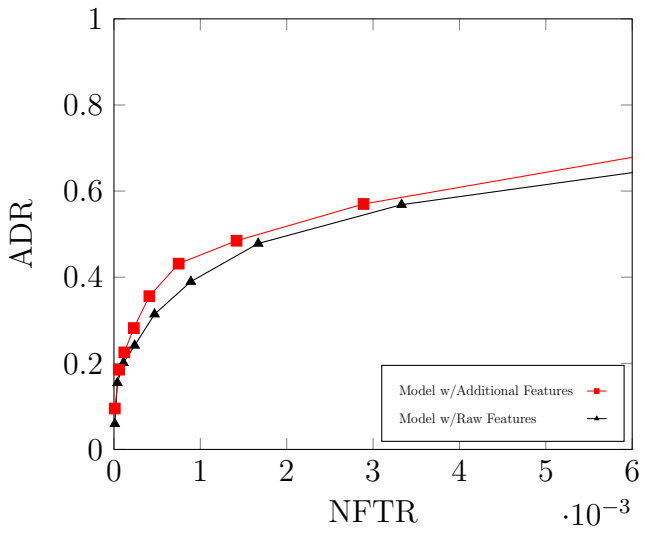

Figure 15: The fraud detection performance comparison of models with raw and additional features

The model's performance increased after adding the specified features to the

training set with the same features. The economic, demographic, and behavioral attributes added to the model positively affected fraud detection performance in cross-channel transactions.

4.3.5.2 Feature Importance

We analyzed integrated feature importance methods of XGBoost to optimize features of the model. Highly ranked features are chosen according to XGBoost [24] feature importance methods. This method contains different feature importance analysis approaches, including cover, gain, total cover, total gain, and weight. They are blended and created a finalized optimized feature list for the model. Adding additional features to the raw attributes of transactions gives us a data set with 172 features. We sorted them by descending order for each feature importance method and took the union of the first 70 features in each list. The resulted feature list contains 96 features. The first 20 features according to feature importance output are given in Table 11.

Table 11: The most important features and descriptions

Feature	Description
ATM Branch Match	Is Atm in the same city as the accounts branch?
Whitelist Level	Level of whitelist
Malware Name	Malware name from Trusteer
Trusteer Risk Score	Score from Trusteer
Whitelist Last Seen Date	Last used date of whitelisted device
Whitelist Flag	Is device whitelisted?
Familiar Account	Is receiver account familiar?
Connection Type	Connection type (WiFi, Mobile Data, etc)
Malware Flag	Is malware detected?
Amount	Transaction amount
Transaction Type	Type of transaction
Age	Customer's age
Financial Age	Customer's financial age
SIM Unblock Date	SIM unblock age
Segment Type	Customer segment (SME/ME/etc.)
Account Branch	Branch location of account
Min t-3 Month Credit Score	The historical minimum credit score of the last three months
Transaction Channel	Indicates transaction channel
Remote Access Flag	Is any remote access application installed?
IBAN Counter	Same device used for how many different IBANs

The performance comparison between models can be inspected in Figure 16.

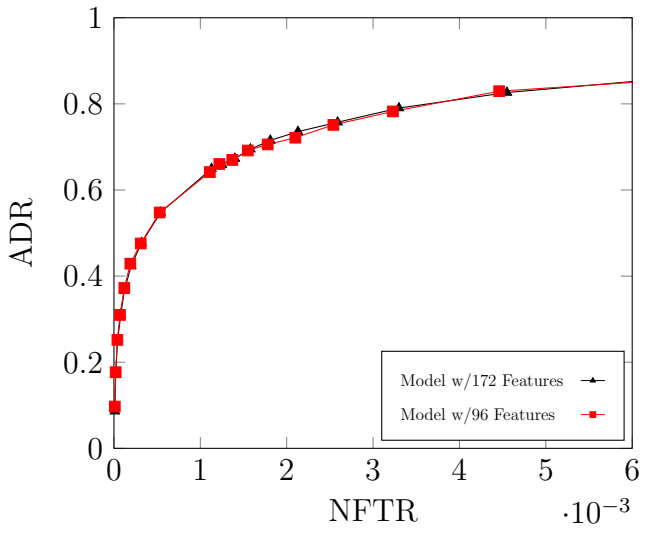

Figure 16: The fraud detection comparison between optimized and all feature models

The optimized model gives almost the same performance as the model with all features. Thanks to feature importance optimization, the model can train with limited features, and it means fewer preprocess computation is required for training and test sets.

4.3.6 Overall Performance

To summarize, we studied generating training sets, feature engineering, and algorithm selection throughout this study. When we combine all techniques above, the overall fraud loss amount detection performance of the model increased dramatically. Also, the algorithm shows this performance with a low false-positive rate. The detailed confusion matrices, figures, and result tables are given below.

The performance of the model for the August 2021 test set is given below in Figure 17. Approximately 75% RTVDR is seen in 0.001, which is an acceptable NFTR level. This indicates that the developed algorithm detected 75% of the total loss amount due to fraud in August 2021.

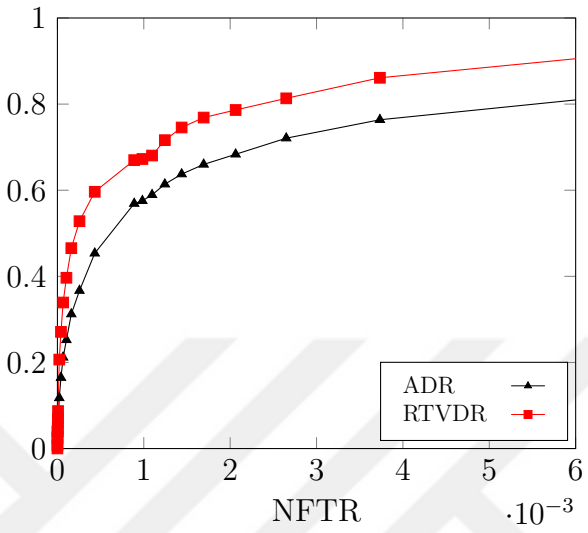

Figure 17: The developed model's overall fraud loss detection performance for August 2021

The confusion matrix according to the selected threshold is given below in Table 12.

Table 12: Confusion matrix of August 2021 on selected threshold

Actual/Prediction	Legit Transaction	Fraudulent Transaction
Legit Transaction	44303751	63817
Fraudulent Transaction	349	602

The result table is detailed in Table 13. The threshold of 0.06 is selected according to Call Count. The Call Count should be between 60000 and 70000.

Table 13: Performance analysis of the model in August 2021 for all thresholds

Threshold	Call Count	AFPR	ADR	RTVDR	NFTR
0.01	285599	239.73	0.818	0.9136	0.0064196
0.02	166379	155.98	0.764	0.8611	0.0037337
0.03	118226	119.53	0.721	0.8134	0.0026493
0.04	92196	99.47	0.683	0.7863	0.0020634
0.05	75744	85.42	0.660	0.7688	0.0016932
0.06	64419	75.80	0.637	0.7457	0.0014384
0.07	55751	68.60	0.614	0.7162	0.0012435
0.08	49179	63.44	0.589	0.6806	0.0010959
0.09	44198	58.73	0.575	0.6725	0.0009839
0.1	39984	54.03	0.569	0.6698	0.0008891
0.2	19609	34.33	0.453	0.5964	0.0004323
0.3	11692	26.20	0.367	0.5280	0.0002557
0.4	7436	20.23	0.312	0.4654	0.0001610
0.5	4870	16.83	0.252	0.3967	0.0001044
0.6	3139	13.12	0.212	0.3392	0.0000663
0.7	1984	10.84	0.165	0.2710	0.0000412
0.8	1069	8.13	0.118	0.2067	0.0000216
0.9	380	6.87	0.049	0.0870	0.0000075
0.91	323	7.16	0.041	0.0698	0.0000064
0.92	251	7.07	0.032	0.0621	0.0000050
0.93	199	6.46	0.028	0.0474	0.0000039
0.94	142	5.27	0.024	0.0410	0.0000027
0.95	110	4.35	0.021	0.0388	0.0000020
0.96	77	3.75	0.017	0.0267	0.0000014
0.97	48	2.62	0.014	0.0243	0.0000008
0.98	24	3.00	0.006	0.0079	0.0000004
0.99	9	3.50	0.002	0.0002	0.0000002

The algorithm's fraud detection performance in cross-channel transactions for September 2021 is shown in Figure 18. According to the developed algorithm, the model trained revealed high ADR and RTVDR at low NFTR. The model continues to behave consistently.

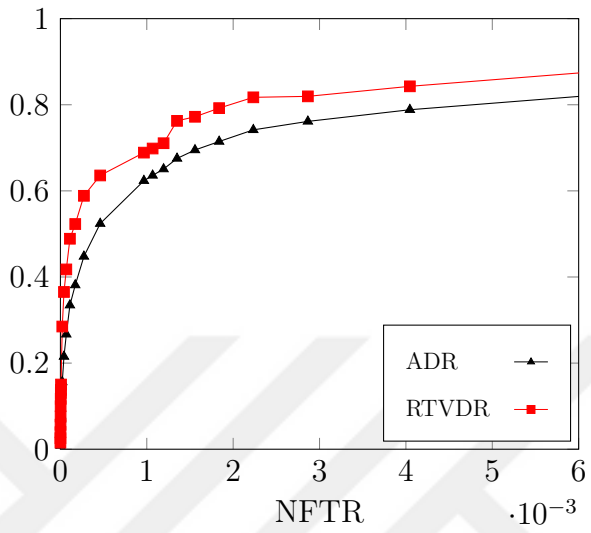

Figure 18: The developed model's overall fraud loss detection performance for September 2021

The confusion matrix of the selected threshold for September 2021 is presented in Table 14.

Table 14: Confusion matrix of September 2021 on selected threshold

Actual/Prediction	Legit Transaction	Fraudulent Transaction
Legit Transaction	46274576	62564
Fraudulent Transaction	268	558

The result table is shown in Table 15. According to the fraud detection business unit, the monthly limit of calling customers or creating alarms for fraudulent transactions is between 60k and 70k. In the result table, Call Count refers to that value. The threshold is selected according to these limits. For instance, the threshold is determined as 0.07 for September 2021. During September, 63122 alarms were generated from the algorithm, and 76% of the total fraud loss was detected.

Table 15: Performance analysis of the model in September 2021 for all thresholds

Threshold	Call Count	AFPR	ADR	RTVDR	NFTR
0.01	321440	293.17	0.834	0.888	0.0069221
0.02	188127	189.75	0.788	0.843	0.0040459
0.03	133388	142.74	0.761	0.820	0.0028650
0.04	104103	116.03	0.742	0.817	0.0022334
0.05	85747	100.31	0.715	0.792	0.0018377
0.06	72771	88.24	0.695	0.772	0.0015581
0.07	63122	79.42	0.675	0.763	0.0013502
0.08	55886	73.57	0.651	0.711	0.0011945
0.09	50000	67.82	0.636	0.698	0.0010677
0.1	45312	63.11	0.624	0.689	0.0009667
0.2	21771	37.56	0.524	0.636	0.0004605
0.3	12983	27.11	0.448	0.589	0.0002722
0.4	8277	20.82	0.381	0.523	0.0001718
0.5	5376	15.64	0.335	0.489	0.0001101
0.6	3388	12.55	0.267	0.418	0.0000684
0.7	2092	9.62	0.215	0.365	0.0000413
0.8	1131	7.10	0.156	0.285	0.0000216
0.9	394	4.94	0.079	0.150	0.0000071
0.91	336	4.47	0.073	0.138	0.0000059
0.92	276	3.73	0.069	0.131	0.0000047
0.93	229	3.63	0.059	0.117	0.0000039
0.94	173	3.00	0.052	0.100	0.0000028
0.95	137	2.91	0.042	0.076	0.0000022
0.96	92	2.42	0.032	0.057	0.0000014
0.97	51	1.82	0.021	0.039	0.0000007
0.98	32	1.73	0.014	0.026	0.0000004
0.99	16	1.00	0.009	0.015	0.0000002

The model's performance in fraud detection for October 2021 is given in Figure 19. The model using the algorithm we developed continues to behave consistently.

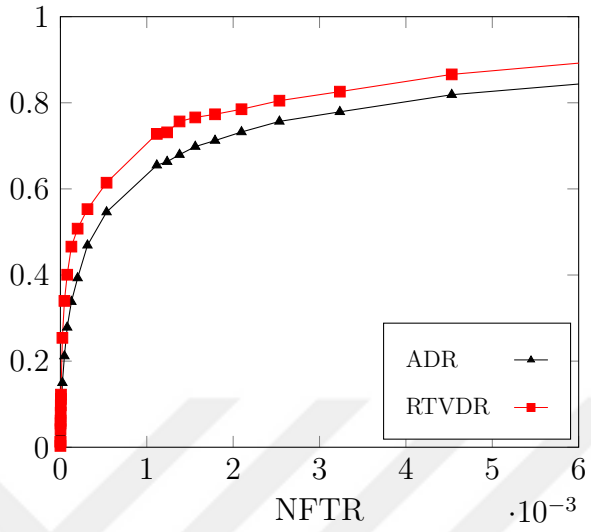

Figure 19: The developed model's overall fraud loss detection performance for October 2021

The confusion matrix of the selected threshold for October 2021 is detailed in Table 16. As can be seen from the table, while there are 863 fraudulent transactions in total, there are approximately 48 million healthy transactions. The developed model detected 585 of 863 fraud transactions at our chosen threshold.

Table 16: Confusion matrix of October 2021 on selected threshold

Actual/Prediction	Legit Transaction	Fraudulent Transaction
Legit Transaction	48077408	66400
Fraudulent Transaction	278	585

The detailed result table for October 2021 is given in Table 17. For October 2021, the most appropriate threshold value was chosen as 0.08 according to the business unit constraints. The model generated 66985 alarms at this threshold and detected 75% of the total loss amount of all fraudulent transactions. At this threshold level, ADR is 68%, and AFPR is 78.11.

Table 17: Performance analysis of the model in October 2021 for all thresholds

Threshold	Call Count	AFPR	ADR	RTVDR	NFTR
0.01	368703	296.98	0.871	0.921	0.0076427
0.02	218822	196.88	0.819	0.866	0.0045305
0.03	156423	151.36	0.779	0.826	0.0032351
0.04	122717	123.92	0.757	0.805	0.0025354
0.05	101561	107.32	0.732	0.785	0.0020964
0.06	86798	94.99	0.712	0.773	0.0017901
0.07	75760	85.25	0.698	0.766	0.0015611
0.08	66985	78.11	0.680	0.757	0.0013792
0.09	60055	72.32	0.663	0.731	0.0012355
0.1	54291	66.56	0.655	0.728	0.0011160
0.2	26199	40.46	0.546	0.614	0.0005344
0.3	15487	28.67	0.469	0.553	0.0003133
0.4	9930	22.35	0.393	0.508	0.0001993
0.5	6412	17.11	0.338	0.466	0.0001272
0.6	4020	13.13	0.278	0.401	0.0000786
0.7	2414	10.60	0.212	0.340	0.0000464
0.8	1234	7.72	0.150	0.254	0.0000230
0.9	379	4.93	0.070	0.122	0.0000066
0.91	308	4.07	0.067	0.112	0.0000052
0.92	256	4.15	0.055	0.097	0.0000043
0.93	199	3.70	0.047	0.080	0.0000033
0.94	155	3.42	0.039	0.062	0.0000025
0.95	126	3.10	0.034	0.055	0.0000020
0.96	84	2.43	0.027	0.042	0.0000013
0.97	54	2.47	0.018	0.013	0.0000008
0.98	27	1.70	0.012	0.013	0.0000004
0.99	7	0.75	0.005	0.003	0.0000001

When the months of August, September, and October were tested with the developed algorithm, the results were consistent with the previous results. The fraud detection performance of the model trained with the training set created according to the training set sampling technique remains constant.

Due to the concept drift, the model's performance decreases over time. Thanks to the retraining mechanism, the prediction performance of the model trained with current data increases in parallel with the experiments.

The model detected about 75% of the fraud loss amount at the reasonable alarm level, called call count.

CHAPTER V

CONCLUSION

As a consequence of the emerging technologies, the variety of transaction channels has increased, but only 26% of financial institutions are tracking the frauds in their transaction channels [26].

5.1 Summary of Findings

Our proposed model tracks and scores the monetary transactions to indicate the possibility of fraudulent transactions from the cross channels in real-time. An AI-powered software solution is provided for the outcome of rarely occurred fraudulent transactions, which properly handles the imbalanced class and concept drift problems. Thanks to the developed algorithm to generate training sets, the generated training sets are very consistent for the retrain model and prevent concept drift. Thanks to the training set sampling techniques, the highly imbalanced class distribution has been prevented from negatively affecting the model's performance. As a result of added features to the model as the output of the feature engineering study, the developed model algorithm can better understand and classify customer behavior. Our model detects cross-channel frauds quite accurately over real banking datasets. It can catch almost 75% of total fraud loss with a low false-positive ratio and overcome aging with continuous training. As a consequence of the low false-positive ratio of the output produced by the model, customer satisfaction is not adversely affected. The private bank reviewed the outputs of this study and measured the performance of the generated model algorithm, and decided to integrate it into their systems to score cross-channel transactions.

5.2 Future Work

The proposed learning strategy and the trained algorithm will be integrated into the live system. We are developing an API service for a private bank to score cross-channel transactions with a 50 ms response time. This structure will also train a new model with the model's features developed in the study every month by using the training set sampling methods used in this study and periodically updating the model algorithm that scores the operations. Our solution will become good support to conventional rule-based solutions that require constant maintenance and analysis.

REFERENCES

[1] J. Gee and M. Button, "The financial cost of fraud 2021: The latest data from around the world." crowe.com `https://www.crowe.com/uk/insights/financial-cost-fraud-data-2021` (accessed March 12, 2022).

[2] J. Gee and M. Button, "The financial cost of fraud 2019: The latest data from around the world." crowe.com `https://www.crowe.com/uk/insights/financial-cost-of-fraud-2019` (accessed March 10, 2022).

[3] K. Worobec, "2021 fraud - the facts 2021." ukfinance.org `https://www.ukfinance.org.uk/policy-and-guidance/reports-publications/fraud-facts-2021` (accessed February 24, 2022).

[4] K. Worobec, "2021 half year fraud update." ukfinance.org `https://www.ukfinance.org.uk/policy-and-guidance/reports-publications/2021-half-year-fraud-report` (accessed February 24, 2022).

[5] I. Molloy *et al.*, "Graph analytics for real-time scoring of cross-channel transactional fraud," in *International Conference on Financial Cryptography and Data Security*, pp. 22–40, Springer, 2016.

[6] S.-J. Yen and Y.-S. Lee, "Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset," in *Intelligent Control and Automation*, pp. 731–740, Springer, 2006.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[8] M. Urban, "Why managing cross-channel fraud is a must." fico.com `https://www.fico.com/blogs/why-managing-cross-channel-fraud-must` (accessed March 21, 2022).

[9] A. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G. Bontempi, "Credit card fraud detection and concept-drift adaptation with delayed supervised information," in *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2015.

[10] D. Malekian and M. R. Hashemi, "An adaptive profile based fraud detection framework for handling concept drift," in *10th International ISC Conference on Information Security and Cryptology (ISCISC)*, pp. 1–6, IEEE, 2013.

[11] A. Yeşilkanat, B. Bayram, B. Köroğlu, and S. Arslan, "An adaptive approach on credit card fraud detection using transaction aggregation and word embeddings," in *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pp. 3–14, Springer, 2020.

[12] K. Philip and S. Chan, "Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection," in *Proceeding of the Fourth International Conference on Knowledge Discovery and Data Mining*, pp. 164–168, 1998.

[13] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, "Data mining for credit card fraud: A comparative study," *Decision Support Systems*, vol. 50, no. 3, pp. 602–613, 2011.

[14] J. Jurgovsky *et al.*, "Sequence classification for credit-card fraud detection," *Expert Systems with Applications*, vol. 100, pp. 234–245, 2018.

[15] B. Bayram, B. Köroğlu, and M. Gönen, "Improving fraud detection and concept drift adaptation in credit card transactions using incremental gradient boosting trees," in *19th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 545–550, IEEE, 2020.

[16] V. Dheepa and R. Dhanapal, "Behavior based credit card fraud detection using support vector machines," *ICTACT Journal on Soft computing*, vol. 2, no. 4, pp. 391–397, 2012.

[17] J. O. Sinayobye, F. Kiwanuka, and S. Kaawaase Kyanda, "A state-of-the-art review of machine learning techniques for fraud detection research," in *IEEE/ACM Symposium on Software Engineering in Africa (SEiA)*, pp. 11–19, 2018.

[18] Y. Patel, *Cross channel fraud detection framework in financial services using recurrent neural networks.* PhD thesis, School of Computing and Digital Media, London Metropolitan University, London, UK, 2019.

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

[20] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2623–2631, 2019.

[21] C. Győrödi, R. Győrödi, G. Pecherle, and A. Olah, "A comparative study: Mongodb vs. mysql," in *13th International Conference on Engineering of Modern Electric Systems (EMES)*, pp. 1–6, IEEE, 2015.

[22] J. Carlson, *Redis in action.* Simon and Schuster, 2013.

[23] A. Boicea, F. Radulescu, and L. I. Agapin, "Mongodb vs oracle–database comparison," in *Third International Conference on Emerging Intelligent Data and Web Technologies*, pp. 330–335, IEEE, 2012.

[24] 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*, pp. 785–794, 2016.

[25] FICO, "Reducing fraud losses with enhanced cnp models on the fico® falcon® platform," 2018.

[26] M. Urban, "New survey reveals top fraud threats and vulnerabilities." fico.com https://www.fico.com/blogs/new-survey-reveals-top-fraud-threats-and-vulnerabilities (accessed March 20, 2022).

VITA

Uğur Dolu received his B.Sc. degree in Electrical and Electronics Engineering from Ozyegin University, Istanbul, Turkey, in 2019. He started his M.Sc. in the Electrical and Electronics department. He made a horizontal transfer from there to Data Science under the supervision of Emre Sefer at Ozyegin University. In 2020, he started to work in Yapi Kredi Technology as an R&D Engineer in Istanbul, Turkey.