Real-Time Credit Card Fraud Detection

Rutgers University has made this article freely available. Please share how this access benefits you. Your story matters. [https://rucore.libraries.rutgers.edu/rutgers-lib/61935/story/]

This work is the AUTHOR'S ORIGINAL (AO)
This is the author's original version of a work, which may or may not have been subsequently published. The author accepts full responsibility for the article. Content and layout is as set out by the author.

Citation to this Version: .

Article begins on next page
Real-Time Credit Card Fraud Detection

Lakshya Sahai  
Department of MSIS  
Rutgers Business School  
Newark, NJ, United States  
lakshya.sahai@rutgers.edu

Kemal Gursoy  
Department of MSIS  
Rutgers University  
New Brunswick, NJ, United States  
kgursoy@business.rutgers.edu

Abstract—Credit card fraud is growing along with the development of technology in today’s world. Hundreds of researches have been made in the past for more than four decades and still, the problem is critical, affecting large financial and banking companies. In this paper, we are focusing on the design and development of an advanced real-time credit card fraud detection framework with the help of big data technologies from one of the leading public cloud providers, Microsoft Azure. We have designed the complete framework to satisfy most current fraud detection systems which could process a large amount of data in real-time and improve the accuracy by implementing multiple machine learning algorithms. We further implemented multiple layers of the workflow including the ingestion layer, streaming layer, processing and transformation layer, model training and scoring layer and storage layer. With the help of these, we were able to build massive storage, fast detection, model training, and real-time fraud detection system. This paper aims in designing the modern credit card fraud detection system and has been tested by a sample dataset to achieve our goal.

Keywords—Credit card fraud, Microsoft Azure, SGD Classifier, Extreme Random Trees

I. INTRODUCTION

According to the 2019 Federal Reserve Payments Study [1], an estimate of total card payment has reached the mark of 131.2 billion with a value of $7.08 trillion in 2018. There is a growth in credit card payments at an expedit rate of 8.9% per year from 2015 to 2018. With this, there is a theaterial increase in credit card fraud losses as well and has reached the mark of $27.85 billion in 2018, according to the Nilson Report 2019 [2]. In general, we are more protected using a credit card on the internet whereas when we walk into a retailer shop, we are handing our cards to a human, human steal, they can easily steal the information from the card through the magnetic strip and use it for fraudulent activities. When the magnetic strip was created, identity theft was an issue and so the data was not properly encrypted and once the criminals figured it out it became the huge market for them. To deal with this, credit companies introduced chip embedded on the card which basically consists of identity information of the cardholder in encrypted form. Although, the problem is not yet solved. Skimmers are actively involved in such activities and it will continue to grow if the magnetic strip is still there on the back of our cards. In a country like China, credit card fraud is deviated compared to a country like the US. China prefers mobile device payments from small purchases to bigger assets with a payment of as high as USD 47,500. This payment method is really a good substitute for other payments like a credit card since it is more secured with multi-factor authentication. According to a China UnionPay survey [3], more than 60% of consumers had been exposed to mobile payment security threats and to combat such threats even on such kind of secured payment method, there is a huge scope to improve the fraud detection system by reducing the likelihood of false positives.

Types of Fraud [4]:

Phishing Scams – This is one of the most common and frequent types of online fraud of phishing e-mail scams. The scam emails are designed by fraudsters to look like an email from banks, financial institutions, social media or sometimes in the name of friends and relatives with the motive to obtain sensitive information.

Counterfeit Credit Cards – Counterfeiting involves producing a fake in the name of the original. That means duplication of real credit card through the magnetic strip or the embedded chip.

Identity Theft – This happens when sensitive information like SSN, driving license, account number or any similar identity of a person leaks out to a fraudster.

Fraudulent application – This is a case of fraud when a customer requests for a new card from the bank based on someone else’s identity or fake identity.

In this paper, we are proposing an end to end solution for the credit card fraud detection system in order to improve the accuracy and to improve the ability to handle billions of transaction processing capability. To fight fraud, we need to fine-tune the existing detection system in order to improve both the needs since they have become more important with the outrage growth of transactional data. With every payment transaction, the size of stored historical transactions is reaching Petabytes or even Exabytes of data. According to recent trends, with the emergence of big data technologies, fraud detection systems have been improved a lot compared to the past decade. Since the payment transactions process has moved from physical cards to virtual cards hence implementing big data tools alone will not help in solving our problem. We need to focus on making a hybrid solution to address the challenges that we are facing in current fraud detection systems.

The solution to our problem involves the design and development of workflows by implementing different components to imbibe different detection models using a few machine learning algorithms in order to improve the accuracy of the whole system.

The rest of the paper is structured as follows–

In section II, we discussed similar researches. Section III covers the methodology that we used in our research. Section
II. RELATED WORK

In the existing system, we researched when the credit card fraud problem first occurred, the main prevention solution was revolving around “how can we protect our credit card from being stolen”. In order to reduce the incidence of credit card fraud, the issuer ensured that controls were designed to prevent the theft of credit cards or use of stolen cards, at the same time directed to take immediate action in case fraud has already occurred and further investigating the fraud without delay. There were no special preventive methods for detecting credit card fraud four decades ago except carrying credit card safe. Several methods have already been proposed since 1990. The rest of this section describes various machine learning models that have been implemented already and discussion about existing detection systems. In paper [5], they researched about three supervised classification models: logistic regression, gradient boosted trees, deep learning. The results were amazing as they found deep learning with the highest recall and highest accuracy which outperforms the other two classification models. Logistic regression did not perform well due to the reason of hidden relationships. In contrast to this, [6] analyzed various machine learning models like logistic regression, random forest, Naïve Bayes and found logistic regression with the highest recall. Well, the highest accuracy does not mean the best model always. We also need to look upon the weighted score of precision and recall which concludes the closest model. Another research [7] has shown the comparison of K-NN, Naïve Bayes and Logistic Regression Classifier and concluded that logistic regression does not perform well for highly skewed data as it really tends to overfit the data. This is where we put Stochastic Gradient Descent [8] to make a better model. Mohammed Zakariah [9] has helped with the intent to understand the random forest model and how it can be used in various use cases. Authors of paper [10] helped to deal with the imbalanced datasets by implementing SMOTE and bagging with random forest for producing the best overall accuracy by considering f score, AUC, ROC, precision and recall measures. Ghosh and Reilly [11] have set a remarkable example where they used neural networks in multiple layers to detect fraud and how they have improved the performance in terms of accuracy and early detection.

Hundreds of researches have been made focusing on credit card fraud implementing tons on models and ways to improve the accuracy and overall detection system. Many machine learning models like Auto Encoder (AE), Restricted Boltzmann Machine (RBM) [12], Bayes Minimum Risk [13], Decision Tree, Support Vector Machine (SVM), Logistic Regression, Probabilistic Neural Network (PNN), Group Method of Data Handling (GMDH) [14] have been focused numerous times and some of them are cost-effective and some are really expensive computationally. In this paper, we are going to fuse some pre-existing models which are cost-saving and the framework to carry out our experiment which we implemented is available at the lowest cost in the market.

III. METHODOLOGY

In this paper, we are performing analytics on the sample dataset which has been taken from Kaggle [15]. This dataset has been collected by the Université Libre de Bruxelles during a research collaboration of the worldline and machine learning group for detecting frauds. The dataset contains a credit card transaction made in 2013 in Europe. It contains 284,807 transactions including only 0.172% of fraud transaction which makes it a highly imbalanced dataset.

There is a total of 31 variables, due to the confidentiality issue, the dataset is PCA transformed to hide the customer’s sensitive information. We will be dealing with only 3 features in this dataset i.e. ‘Amount’, ‘Time’, and ‘Class’. Feature ‘Amount’ indicates the transaction amount, ‘Time’ indicates the time elapsed between every other transaction and the first transaction and the feature ‘Class’ is the target variable indicating 1 in case fraud occurs and 0 if not.

IV. EXPERIMENT

In this paper, we are implementing the most advanced big data tool and technologies provided by one of the largest public cloud provider, Microsoft Azure. When it comes to the first and foremost thing ‘data’, most data engineers, data scientists realize the need for smart and intelligent application systems to meet their business needs. This is where Microsoft Azure brings the platform with advanced big data tools. We have designed a prototype which is an end to end data pipeline solution for detecting real time credit card fraud detection. This paper further discusses all the big challenges currently faced by credit card providers, banks or financial institutions. The focus of our research remains intact with various machine learning algorithms with the help best solutions available.

We have designed a high-level architecture of the entire real-time credit card fraud detection system.

This is true that we do not have access to real time data hence we focused on what we really implemented and that is why we are just proposing a solution for the actual business problem in the real world which can be developed and implemented in production. Below is the infrastructure, built on the Microsoft Azure platform where each component is playing a vital role in order to achieve our goal. This design includes multiple layers of workflows like data ingestion in real-time i.e. streaming layer, batch layer for building and training machine learning models, processing layer for transforming data coming in huge amount storage layer for high scalability and fault tolerance, and fraud alert through visualization.

To begin with, let’s start with data ingestion. This is one of the biggest challenges faced by data engineers for developing intelligent application systems.
A. Azure HDInsight Kafka:

Azure HDInsight Kafka [16] is one of the powerful data ingestion frameworks that Azure offers. We can stream data directly to Azure Databricks from Kafka for processing and transforming the data. Apache Kafka is an open-source data ingestion framework from multiple data sources and aggregating the data at the same time to a single data source to distribute to the consumers. There are three major components inside Kafka i.e. event producer, topics and event consumer. Event producers direct records in the form of messages to the Kafka cluster where they are stored in topics. The topic is like a log where event producers append messages. To link Kafka and Databricks, we need to implement a virtual network peering between both the services so that the two clusters (Kafka cluster and Databricks cluster) are connected. Virtual network peering is the way to communicate between isolated resources. As soon as the private communication network is established between Azure Databricks and Azure HDInsight Kafka [17], we start consuming messages from Kafka. This stream of messages is called DStream. We made sure that whenever the spark streaming job inside Databricks is restarted, it will not read the data from the starting. It will rather read from where it had stopped. This is how we initialize spark streaming jobs and start consuming messages from Kafka.

B. Azure Databricks

Azure Databricks [18] is one of the most powerful big data processing frameworks available to us to date. Databricks is designed and developed in collaboration with the founders of open source Apache Spark. With Azure Databricks, users can analyze, train, build, score and deploy multiple kinds of applications. The typical use cases of Azure Databricks include Interactive Analytics, Data Integration, Machine Learning and Stream Processing.

![Apache Spark Ecosystem](image)

To start with Azure Databricks, we create Azure Databricks workspace. The workspace is essentially space where compute clusters and notebook reside. We have a dataset stored in the file system. The file system can be anything like Azure Blob Storage or Azure Data Lake. Here, our data resides in the Azure Data Lake Store. To perform ETL operation on our data, we imported data from the file system to the Databricks cluster by using spark SQL API. We ran an initial import spark job to perform ETL operations on our dataset and now that we have our data cleaned and ready to be trained. Next, we implemented auto-machine learning (Auto ML) SDK in the Databricks cluster. The reason for choosing Azure Databricks is, it provides on top standard apache-spark including Azure Active Directory integration with Role-Based Access Control, the collaborative feature of workspace and git integration, running scheduled jobs for automation, and build, train and deploy machine learning models at scale.

C. Azure Machine Learning

Azure Machine Learning [19] is a platform as a service provided by Azure allowing data engineers and data scientists to train, score, evaluate, automate machine learning models at scale. For model deployment, we used machine learning service as our solution but we do have other deployment solutions as well like Azure Kubernetes Service or web apps. To start with Azure ML service, we created a linked workspace from Azure Databricks to connect the two services. As we know that our data is ready for training, before that, we used SMOTE (Synthetic Minority Oversampling Technique) to make a balanced dataset, since it is highly imbalanced. With the help of the Auto ML feature, we were able to balance our dataset easily in just a few steps.

Now that we have a balanced dataset, we implemented two machine learning models.

a) Stochastic gradient descent (SGD) Classifier:

SGD Classifier [20] uses stochastic gradient descent as a solver that uses different loss functions and penalties for classification. SGD classifier has been around in the community for a long time until recently it can be seen under the spotlight gaining attention for dealing with large scale datasets. The reason for using the SGD Classifier over Logistic Regression is that the Logistic Regression uses gradient descent which is a deterministic approach, meaning, no matter how many iterations you perform on the training dataset you will get almost the same results. On the other hand, the SGD classifier uses a stochastic gradient descent approach, meaning, you will never use the entire training dataset at once instead you will pick random sets and generate different results always. And this makes it faster than logistic regression and a better choice.
b) Extreme Random Trees:
In a random forest, we grow multiple trees such that each tree comprises of the square root of the total number of features present. Additionally, we make use of bootstrap samples or samples with replacement this is where we have out of the bag error score that we can use later on for validation.
Now, how is it different from Extreme Random Trees? The first difference is in selecting samples. In extreme random trees [21], we select samples for every decision tree without replacement. So all the samples are unique in nature. The total number of features selected still remains the same that is the square root of the total number of features.
Another difference lies in the fact that is instead of computing the locally optimal split for a feature combination, a random value is selected for the split for the extra tree. So the point is rather than not spending time in finding out the best splitting point, we randomly picked a splitting point and this leads to more diversified trees and fewer splitters to evaluate when training an extremely random forest.
When extreme random tree classifier was tested with readily available datasets, we observed that when we had noisy features in our dataset extra tree classifier seem to outperform random forest. However when all the features are relevant and when trained both extreme random trees and random tree methods seem to have achieved the same performance be in terms of accuracy, AUC, ROC or F1 score that we are chasing.

<table>
<thead>
<tr>
<th>Measures</th>
<th>SGD Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>0.99930</td>
</tr>
<tr>
<td>Recall</td>
<td>0.99933</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0.99933</td>
</tr>
<tr>
<td>AUC</td>
<td>0.96209</td>
</tr>
<tr>
<td>F1</td>
<td>0.99926</td>
</tr>
</tbody>
</table>
focussing on a specific classification threshold, we can implement a hybrid approach which blends multiple analytic techniques from different disciplines to provide far more powerful and accurate fraud detection system. We can always test the new hypotheses, validate the old ones and listen to the possibilities suggested by the data.

VI. ACKNOWLEDGMENT
We would like to thank the readers for giving it a read and reviewers for their valuable comments and suggestions. Grateful for the wonderful support of Prof Kemal Gursoy for introducing this problem and the dataset.

VII. REFERENCES
[5] Gabriel Rushin ; Cody Stancil ; Muyang Sun ; Stephen Adams ; Peter Beling, “Horse race analysis in credit card fraud—deep learning, logistic regression, and Gradient Boosted Tree”
[6] Dejan Varmedja ; Mirjana Karanovic ; Srdjan Sladojevic ; Marko Arsenovic ; Andras Anderla, “Credit Card Fraud Detection - Machine Learning methods”
[8] Tong Zhang , “Solving large scale linear prediction problems using stochastic gradient descent algorithms
[12] A Pumsirirat, L Yan, “Credit card fraud detection using deep learning based on auto-encoder and restricted boltzmann machine”

D. Azure Cosmos DB
Azure Cosmos DB [22] is a NoSQL database, which means that they are designed to handle the massive scale of data, are globally distributed and multi-model service and can be really awesome for transactional processing but when it comes to simple aggregated queries, they tend to be very expensive. Now, the reason for introducing Cosmos DB here is because of its strength that is, multi-master capacity, global distribution for high throughput ingest and low latency service. We are storing the pre-scored transactions make it highly available and low latency less than 10ms which is a real customer-centric solution. So, we add up applicable customer regions, we can scale up and scale down to handle the workload, we can choose from multiple consistency level option it provides. Azure Cosmos DB is connected to the Databricks through the Spark Connector.

E. Power BI
Microsoft Power BI is a powerful visualization tool for dashboards and alerts.

V. CONCLUSION
Real-time credit card fraud detection is a real challenge the banking industry is still facing in today’s era. The reason could be many, but the most challenging things are handling huge data generated due to trading transactions every day. In this paper, we proposed a solution for handling the massive workloads and processing them in real-time by applying machine learning algorithms and pipelines and built highly available, scalable, fault-tolerant fraud detection systems. With the help of this, we were able to achieve better accuracy than the previously existing system. We expect possible future work which is considerable. Instead of

<table>
<thead>
<tr>
<th>Measures</th>
<th>ExtremeRandom Trees</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>0.99698</td>
</tr>
<tr>
<td>Recall</td>
<td>0.99849</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0.99849</td>
</tr>
<tr>
<td>AUC</td>
<td>0.93489</td>
</tr>
<tr>
<td>F1</td>
<td>0.99774</td>
</tr>
</tbody>
</table>
[13] Alejandro Correa Bahnsen; Aleksandar Stojanovic; Djamila Aouada; Björn Ottersten, “Cost Sensitive Credit Card Fraud Detection Using Bayes Minimum Risk”

[14] G. Ganesh Sundarkumar; Vadlamani Ravi; V. Siddeshwar, “One-class support vector machine based undersampling: Application to churn prediction and insurance fraud detection”


[22] https://docs.databricks.com/data/data-sources/azure/cosmosdb-connector.html