Fraud Credit Card Transaction Detection

DSCI 303 Final Project

(Apple) Xuanchen Li
xl102@rice.edu
Rice University

(Derek) Tiancheng Fu
tf18@rice.edu
Rice University

ABSTRACT

We aim to train a model that distinguishes between legitimate and fraudulent transactions specifically on credit cards. We performed data cleaning and feature engineering on our dataset, then compared the effect of logistic regression (baseline model), Support Vector Machine, random forest, and AdaBoost.

1 INTRODUCTION

Fraud detection has played a crucial part in building secure products for companies that offer financial services. By clearly, efficiently, and accurately identifying fraud transactions, companies are able to prevent clients from losing money. In this project, we aim to train a model that distinguishes between legitimate and fraudulent transactions specifically on credit cards. Such models can be used by banks and online payment companies to block malicious transactions and protect their customers' properties.

Our hypothesis is that fraudulent credit card transactions are associated with unusual patterns, and can be detected by identifying differences from a cardholder's normal behavior and transaction characteristics.

2 RELATED WORK

Financial institutions tend to use a combination of techniques and algorithms to detect fraudulent transactions, depending on the specific characteristics of the transactions. Specific client/customer data is highly private in the real world, hence we do not know the exact modeling process behind each fraud detection department of those companies. Some research papers are available, such as Credit Card Fraud Detection using Machine Learning Algorithms by V. Dornadula and S. Geetha, where they used SMOTE (an oversampling technique) and SVM (Support Vector Machine) for their model.

We aim to use this project as a practice to build a robust model with proper data cleaning and feature engineering process that will serve well when we get to work with industry-level datasets in the near future.

3 METHODS

We performed data cleaning and feature engineering on our dataset, then compared the effect of logistic regression (baseline model), Support Vector Machine, random forest, and AdaBoost.

3.1 Exploratory Data Analysis

We use the dataset "Credit Card Transactions Fraud Detection Dataset" [**?**] found on Kaggle. The dataset is created using a simulator, but should be highly similar to a real-world dataset in credit card transactions and contains both legitimate and fraudulent credit card transactions from 1/1/2019 to 12/32/2020. It contains 1,296,675 rows of data in training set and 555,719 row of data in training set. For each row, the dataset includes 10 numeric variables and 12 categorical variables. There is only one row of missing data in the training set. Since removing it from the dataset would have minimal effect, we simply dropped the row of data with missing values.

Table 1: Correlation between Numeric Variables and the Target Variable "is_fraud"

Variable Name	Correlation
is_fraud	1.000000
amt	0.219404
city_pop	0.002136
lat	0.001894
merch_lat	0.001741
merch_long	0.001721
long	0.001721
cc_num	-0.000981
zip	-0.002162
unix_time	-0.005078

As shown in **Table 1**, most numeric variables do not have a strong correlation with the target variable "is_fraud" other than the variable "transaction amount", which has a seemingly moderate relationship with the target variable "is_fraud". This indicates that standard linear models might not be able to produce the best results for this dataset. We could potentially look into tree algorithms for a better model. In addition, we also observe that some numeric variables are suitable for further feature engineering, such as variables "cc_num", "zip", and "unix_time".

We then looked at the categorical variables, but not much clear pattern is found from any individual variable. We plan to perform feature engineering on some of those variables, such as "category" (transaction category), "job" (jobs of the cardholders), "gender". We also found some categorical variables non-essential to our model and should be removed before we build the model. For example, categorical variables "first" and "last" record the first and last name of the cardholders and thus provide overlapping information with the customers' credit card numbers recorded in variable "cc_num".

3.2 Feature Engineering

In our exploratory data analysis, few variables exhibited strong correlation with the target variable. Therefore, we decided to use feature engineering to create new variables based on existing dataset, tailoring them to better suit the purpose of our project.

We first dropped columns that are unnecessary, namely “first”, “last”, “street” (street address), “city”, “state”, and “trans_num” (identifier for each transaction).

We then created a few new variables. The “dob” variable (date of birth of cardholder) was transformed into a numeric variable “age”. We split “unix_time” (time of the transaction) into “trans_weekday” and “trans_hour”, representing the day of the week and the hour of the day of the transaction, respectively.

After that, we applied count encoding to categorical variables “category”, “merchant”, “job”, “cc_num” and “zip”. This process involved creating a new variable for each of these variables, populated with the number of occurrence of their respective categorical values. Additionally, we used one-hot-encoding on the “gender” variable, creating two new variables “gender_F” and “gender_M”.

Finally, we concluded the feature engineering phase by oversampling our target variable “is_fraud”. We used SMOTE (Synthetic Minority Oversampling TEchnique) to address the imbalanced distribution between fraudulent and non-fraudulent transaction data points. This prevented our models from over-fitting to the predominant non-fraudulent transaction class, thereby enhancing both the performance and robustness of our models.

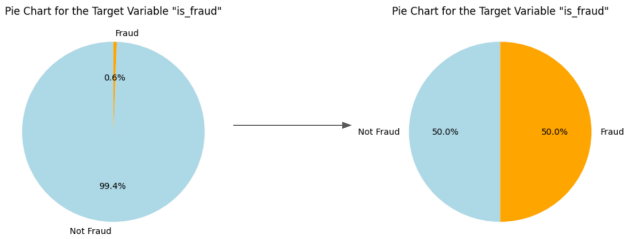

Figure 1: Using SMOTE to address data imbalance

3.3 Modeling

We used 4 different models and tried to compare their performances.

We made logistic regression as our baseline model, since it is the most classic and fundamental algorithm for doing binary classification on datasets.

We built a support vector machine (SVM) model. SVM is a model that’s been widely used in prior works related to fraud transaction detection. If SVM turns out to have good performance on the dataset, that means that there exists a clear "margin" between fraudulent and non-fraudulent transactions.

We used 2 tree-based algorithms, random forest and Adaptive Boosting (Adaboost). Our exploratory data analysis showed that most variables do not have a strong linear correlation with the target variable, so we hope to see if we can capture nonlinear relationships between variables through training tree-based models.

4 RESULTS

We first trained the four models without tuning the hyper-parameters. The confusion matrices of the models are shown in **Table 2**.

Our baseline model, the logistic regression model, achieved an accuracy of 95.4% with a precision of 4.15% and a recall of 84.62%. Support Vector Machine performed the best without tuning. It

Table 2: Confusion Matrix Across Models, before Tuning

Model	TN	FP	FN	TP
Logistic Regression	5290	254	2	11
SVM	5411	133	2	11
Random Forest	5312	232	1	12
Adaboost	5173	371	1	12

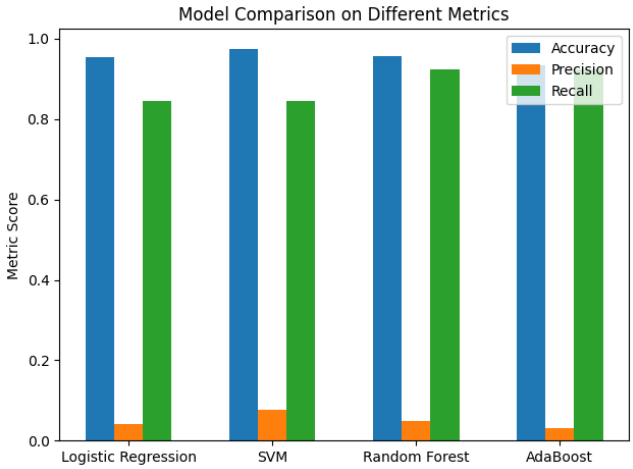

Figure 2: Model Accuracy, Precision and Recall, before Tuning

achieved a 97.57% accuracy—2.17% higher than the baseline model, and the highest among all four models we used. The random forest model performed 0.4% better than the baseline, with an accuracy of 95.81%. Adaboost performed 2.1% worse than the baseline model when we use a learning rate of 0.1 and 50 decision stumps, each with 1 decision node. We visualized the comparison between these models in **Figure 2**.

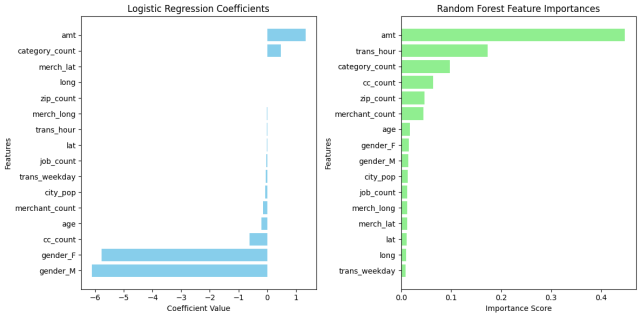

Figure 3: Feature Importance in Logistic Regression and Random Forest Model, before Tuning

We also extracted a side-by-side histogram **(Figure 3)** to compare the feature importance in our baseline logistic regression model and the random forest model. A few variables such as “amt” (transaction amount), “category_count”, “zip_count” showed relatively

high importance across both models. Additionally, the random forest was able to better utilize the count-encoded variables, such as "category_count", "cc_count", "zip_count", and "merchant_count" due to the nature of the tree-structured algorithm.

5 DISCUSSION

Before we invested our time into tuning the hyper-parameters, the SVM model and the random forest model performed better than the baseline logistic regression model. The SVM model had the highest accuracy on the testing dataset among all four models we used. However, training the model took more time than the others, and the result can be hard to interpret and visualize the model in higher dimensions, unless we use Principal Component Analysis (PCA). Meanwhile, the random forest model performs well in that it built a great balance between accuracy (95.81%), precision (4.92%), and recall (92.31%). Besides, as shown in **Figure 3**, it was able to capture information contained by the variables created via count-encoding. However, we believe we can further improve this model via hyper-parameter tuning.

6 CONCLUSION

7 CONTRIBUTION