Fraud Detection

Hitesh Thadhani, Surjit Singh, Yerraguntla Aditya Sai
School of Information Studies, Syracuse University
Syracuse, NY, USA, {hthadhan, ssingh60, ayerragu}@syr.edu

https://fraud-detection-system.herokuapp.com/

Abstract

Financial institutions guarantee payments made online. They are obligated to make payments to the recipient of payment regardless of whether the transaction is legitimate or fraudulent. These organizations will be highly interested to know the legitimacy of a transaction before it is processed. Due to the scale and variety of online transactions, a human can't determine the legitimacy of the transaction. Data Mining techniques can be extremely helpful in aiding the process. Using the Cross-Industry Standard Process for Data Mining (CRISP-DM) we built machine learning models to accurately predict if a transaction is fraudulent or legitimate.

Introduction

Financial institutions have built digital payment solutions to make transactions free of cash. Credit and debit cards have been used by customers as early as the 1970s in North America. The adoption of digital payment methods has given rise to digital payment fraud. In the past, digital payments were made physically using the actual card at a retail store. Over the last couple of decades with the E-Commerce boom, customers started to make transactions online where a physical card is not required to make the transaction. The Card Not Present (CNP) transactions are guaranteed by various financial institutions which makes them liable for payment to the recipient of a transaction regardless of the legitimacy of the transaction. A lot of customers store their card information digitally which leaves them vulnerable in the event the information is obtained by nefarious third-party actors. Oftentimes the customers are unaware of the fraud until much later at which point the customer might get a refund, but the financial organization is stuck with the payment. By building a fraud detection system we intend to gauge the legitimacy of the transaction before the payment is processed. This early detection system will save the company millions of dollars every year and protect the customers from identity theft.

In this paper, we experiment and build several data mining/machine learning (ML) models that will help predict if new transactions are fraudulent or not. The dataset we obtained was highly imbalanced with majority of the transactions being legitimate. Training the imbalanced dataset and surpassing the null accuracy was difficult to overcome.
We followed the Cross-Industry Standard Process for Data Mining (CRISP-DM) for building the fraud detection system. Our process starts with exploratory data analysis, data pre-processing, building baseline ML models, building fine-tuned ML models, and performance evaluation. The process was iterative and carried out until the model performance was optimal.

Data Description

The dataset contains proprietary transaction and identity information of the customer of Vesta Corporation, an online CNP payment solutions company. The

dataset was made public through a Kaggle competition named IEEE-CIS fraud detection which was hosted by IEEE-CIS. The data was split into transactions and identity which are joined by TransactionID. Not all transactions have corresponding identity information. The transactions data contained 394 attributes and the identity data contained 41 attributes. The target attribute is **isFraud** (binary attribute) where 0 indicates that there is no fraud and 1 indicates a fraudulent transaction. The isFraud attribute is part of the transactions file. Once the transactions and identity datasets are joined on TransactionID we obtain a combined dataset with 435 attributes and 590,540 rows. There are 404 numeric attributes and 31 categorical attributes. The ratio of legitimate transactions to fraudulent transactions is 27.5: 1 which makes the dataset highly imbalanced.
The metadata for the attributes is extremely limited and the attribute names are not descriptive. The non-descriptive nature of the attributes was done to ensure regulatory compliance of Vesta Corporation’s user data. This adds a layer of difficulty as it makes meaningful feature engineering extremely difficult.

Card1-Card6 provides information about the card used for the transactions. ProductCD describes the type of products being purchased. P_emaildomain, R_emaildomain give information about the user accounts being used for the CNP transactions. DeviceInfo, DeviceType, Id1-id38 describe the identity of the user and their devices. C1-C14, M1-M9 are some of the features without a lot of metadata information.

The unavailability of identity information for all transactions introduces a lot of missing values for identity columns when the data is joined. There are many attributes in the transaction data which have more than 50% of the values as NULL. There are many attributes with significant NULL values that need to be handled before model construction.
The data needed to be reduced to a manageable size due to the limited computational resources available. We implemented stratified sampling and ensured the data distribution of the target variable remains unchanged. The stratified sample data contains 100,391 rows.

Exploratory Data Analysis

Exploratory data analysis is conducted to understand the data and observe potential patterns to explain the effect of the predictor variables on the target. EDA can help uncover the following

- Linear relationship between predictors and target
- Presence of outliers
- Collinearity between different predictors
- Understanding data distribution of the predictor and target

The image below shows the proportion of legitimate and fraudulent transactions and the amount of fraud across the transactions. The amount of fraud is similar to the proportion of fraudulent transactions. This indicates that the transaction amount alone is not sufficient to flag a transaction as fraudulent.

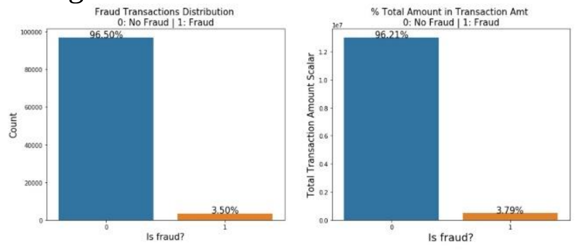


Figure 1. Target variable distribution

Card4 attribute indicates the company that issued the card i.e., visa, master card, discover, and American express. Discover has a much higher rate of fraudulent transactions at 7.2% when compared with 3.5% for Visa and Mastercard. Card4 attribute can be significant for predicting the target.

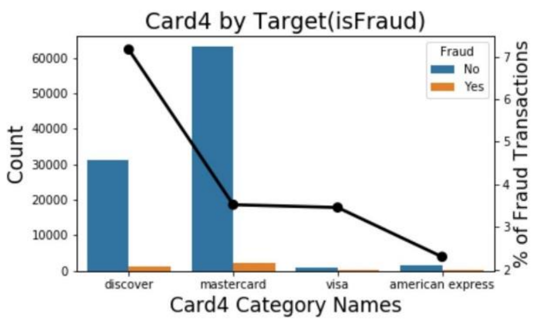


Figure 2. Card4 by Fraud

Card6 attribute indicates the type of card i.e. credit debit or both. The graph shows that most of the transactions occur using a debit card but credit cards tend to have a higher percentage of fraudulent transactions at 7% compared to debit card’s 2.5%. Card6 attribute can be significant for predicting the target.

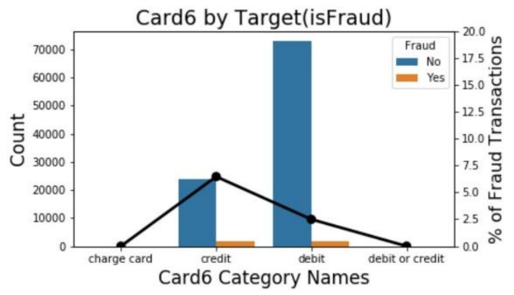


Figure 3. Card6 by Fraud

The _hours attribute is derived from other attributes in the original dataset. We observe a very interesting trend where the volume of transactions is at the lowest between 5 AM and 11 AM but the percentage of fraudulent transactions is much higher during those hours. _hours attribute can be significant for predicting the target.

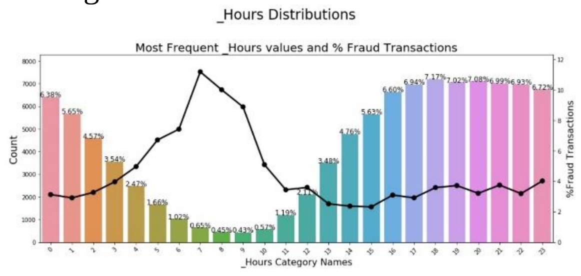


Figure 4. Count of Fraud Transactions every hour of the day

Data Processing

Real-world data coming from various sources has missing values, outliers and in an unsuitable format to be transformed to build machine learning models. There were 2 files for transaction and identity data. We merged these 2 files on the TransactionID being the common column to get the full dataset with 434 columns. Not every transaction has identity information associated with it which increases the number of null values in the dataset. The size of the dataset after merging was 600,000 rows and 434 features. Given the limited computational resources available, we decided to use Stratified Sampling to subsample our dataset while keeping the target variable distribution the same as the original dataset.

We then checked for missing values in our dataset and removed columns with more than 50% of the null values. It is not practical to impute these columns with 50% missing values. Most of the input features actual meaning was masked due to the security concerns due to which we had limited understanding of the dataset even after EDA. So, to impute the missing values in the remaining columns we used Multiple Imputation by Chained Equations (MICE) given the limited availability of the metadata for the input features. MICE imputes the missing values by running multiple regression models and each missing value is modeled conditionally depending on the observed (non-missing) values to complete the dataset. Dataset was scaled using StandardScaler before using MICE for imputing the missing values. For categorical variables, we used Mode to impute the missing values. For card4 and card6, which give information about the card issuer company and type of card respectively, we used groupby with TransactionAmt column to impute the missing card values depending on the median of Transaction Amount for each card. For the email id column, we used only the first part before the domain name instead of the full email address. For example, gmail.com, yahoo.com were replaced by Gmail and yahoo. Once the data imputation was complete for numerical and categorical variables separately, we concatenated the 2 datasets to get the final clean data.

We then checked for multicollinearity by plotting the correlation matrix and removed highly correlated columns from our dataset that do not add additional information to increase the model prediction capability. Machine learning models do not understand categorical variables directly and need to be transformed to numerical before being fed as input to the models. So, we used label encoding to encode the categorical variables as numeric.

We created new features as part of feature engineering by using TransactionDT which contains the date and time information in seconds. We created 2 new features named Day_of_week and Hour_of_day to indicate what day and hour of the week and day respectively when transactions happen. These features have significant importance in the model's prediction which can be concluded by the feature importance derived from the light GBM model run to confirm our hypothesis which we came up with during the Exploratory Data Analysis for the Hour_of_Day being an important feature.

SMOTE Oversampling

During our initial Exploratory data analysis, we found that our target variable isFraud is severely imbalanced with a distribution of 97% no fraud transactions and only 3% fraud transactions.

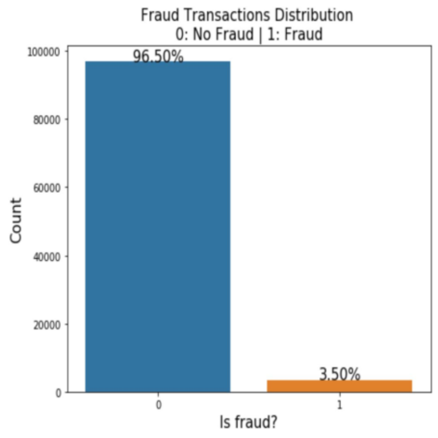


Figure 5. Target variable severely imbalanced

Models trained on a severely imbalanced dataset perform exceptionally well for the majority class than for the minority class. In such cases, accuracy is very high which is misleading as the model predicts no fraud for almost all the time even though a transaction is a fraud by neglecting the instances from the minority class.

One way to rectify this issue is by oversampling the minority class samples to create a balanced dataset for training the models. Synthetic Minority Oversampling Technique (SMOTE) is a technique to create synthesized duplicate examples in the minority class from the existing examples even though they may not add new information to the model but help the model to learn to distinguish between fraudulent and genuine transaction given the balanced training data. Models trained on this balanced data would reduce the number of "False Negatives" fraud transactions being predicted as no fraud for the validation data.

Now that the balanced training is created by splitting the dataset into train and test with an 80:20 ratio, we tried to resolve the problem of dimensionality. Our dataset had 400+ columns and before we run our models, we needed to find the features which contributed to the model's prediction. Running models with 400+ features without knowing which one of them is important is a waste of time and limited computational effort available to us. To reduce the dimensionality of our dataset to only include necessary features, we used Light Gradient Boosting Model with base hyperparameters. The feature importance from the Light Gradient Boosting model helped us to reduce the dimensionality of our dataset to 110 columns with feature importance greater than 0.

Machine Learning Models

We applied four different classification algorithms to the dataset: Logistic Regression, Random Forest, Extreme Gradient Boosting, and Naïve Bayes. These algorithms have shown promising results in the past for classification-based machine learning problems.

Logistic Regression

The Logistic Regression is used for binary classification problems. The objective of logistic regression is to model the mean of the response variable, given a set of predictor variables. The response variable of logistic regression is binary rather. In Logistic Regression, the logit function is used by the model to perform the binary classification. The sigmoid function is an S-shaped curve that is used to map the weighted inputs to 0 or 1.

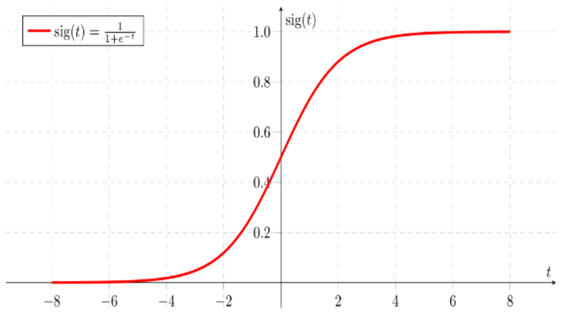


Figure 6.

We use a grid search for running logistic regression to find the best parameters for our model.

Random Forest

Random Forest is an ensemble learning which aims to improve classification accuracy by aggregating the predictions of multiple classifiers. Ensemble methods generate a set of base classifiers and perform a final classification by taking a vote based on the predictions made by each of the base classifiers.

Random Forest is specifically designed for Decision Tree classifiers. It votes on the values generated by a decision tree where each tree is built on an independent set of random vectors. This randomization is done to reduce the correlation between decision trees to improve the generalization error of the random forest.

A random subsection of attributes is selected for each split of each decision tree built for the random forest. Each tree is then allowed to grow to its maximum depth to minimize bias. It should be noted that this can sometimes lead to overfitting due to increased variance. Once all the trees have been built, a prediction can be made using majority voting.

Extreme Gradient Boosting

Extreme Gradient Boosting is another example of ensemble learning type of algorithms. Extreme gradient boosting belongs to the family of boosting algorithms and is a variation of gradient boosting (GBM) algorithm.

Extreme gradient boosting use tree-based models at its core. Trees are built serially one after another to combine multiple weak learners to build a strong learner. Trees are grown serially to reduce the misclassification in subsequent iterations by correcting the prediction errors made by prior trees.

XGBoost is an open-source implementation of the gradient boosting algorithm. XGBoost makes the training process faster and can be parallelized to perform better than the gradient boosting algorithm.

Naïve Bayes

Naive Bayes is a classification algorithm based on Bayes theorem and is a probabilistic type of algorithm. The Naive Bayes algorithm is based on the naïve assumption that all the features are independent of each other and there is no correlation among the various features. It also assumes that each feature is given the same importance in the model prediction along with independence among the features.

Despite the impossible assumption of the features being independent of each other, Naïve Bayes classifiaer works well on real-world machine learning problems and requires only a small amount of training data.

Naïve Bayes calculates the posterior probability for each class by multiplying the prior probability of the class and likelihood which is the probability of attribute given the class. The class with the highest posterior probability is the outcome of the prediction.

Model Evaluation

All the models are fine-tuned, tested, and compared to select the best model to predict a fraudulent transaction.

Evaluation Metric

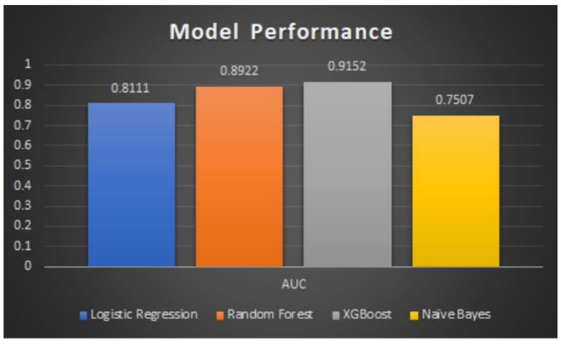
The optimal evaluation metric is one that can help us identify and block fraudulent transactions but also ensure that legitimate transactions are allowed to be completed. The false-positive rates and true positive rates need to be tracked to assess model performance. The Receiver operating characteristics curve is a curve that is ideal for evaluating a classification model. Tracking the true positive and false positive rates ensures that the models can be assessed in an unbiased way. This is essential since our dataset is highly imbalanced and we need to beat the Null accuracy.
The Area Under the Curve (AUC) is the metric we will use to compare the models. AUC is computed using the formula

$$A = \int_{x=0}^{1} \mathrm{TPR}(\mathrm{FPR}^{-1}(x))\,dx$$

where TPR is True Positive Rate and FPR is False Positive Rate.

Test Results

The following image shows the results of the tuned machine learning models on the test dataset.



The AUC scores of the Naive Bayes, Logistic Regression, Random Forest Classifier, Extreme Gradient Boosting models are 0.7507, 0.8111, 0.8922, 0.9152 respectively. The best performing model is the Extreme Gradient Boosting (XGB) model with an AUC score of 0.9152 (91.52%). This indicates that the XGB model generalizes well on unseen data. The parameters of the tuned Extreme Gradient Boosting model is

- Subsample: 0.4
- N_estimators: 500
- Max_depth: 18
- Learning_rate: 0.02
- Gamma: 0.8

Conclusion

The data provided had very limited metadata and non-descriptive attribute names which made it extremely difficult to conduct feature engineering and to make conclusive assumptions about the relationship between attributes. The models built on the stratified sample of the data perform well on unseen data. However, the data used is only a portion of the originally available dataset. Obtaining proper metadata, using additional computing resources, and the entire dataset to train the models will result in a better performing model.

References:

- https://www.amazon.com/Introduction-Data-Mining-Pang-Ning-Tan/dp/0321321367
- https://www.geeksforgeeks.org/naive-bayes-classifiers/
- https://www.datacamp.com/community/tutorials/xgboost-in-python
- https://machinelearningmastery.com/extreme-gradient-boosting-ensemble-in-python/
- https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/
- https://machinelearningmastery.com/what-is-imbalanced-classification/#:~:text=Imbalanced%20classification%20is%20the%20problem,classes%20in%20the%20training%20dataset.&text=Many%20real%2Dworld%20classification%20problems,spam%20detection%2C%20and%20churn%20prediction.
- https://machinelearningmastery.com/logistic-regression-for-machine-learning/