CREDITCARD FRAUD DETECTION

DISCOVERY:

INTRODUCTION:

The usage of credit cards for online and regular purchases is exponentially increasing and so is the fraud related with it. A large number of fraud transactions are made every day. Various modern techniques like Data Mining, Genetic Programming, etc. are used in detecting fraudulent transactions. This paper uses genetic algorithm which comprises of techniques for finding optimal solution for the problem and implicitly generating the result of the fraudulent transaction. The main aim is to detect the fraudulent transaction and to develop a method of generating test data.

A credit card is a thin handy plastic card that contains identification information such as a signature or picture, and authorizes the person named on it to charge purchases or services to his account - charges for which he will be billed periodically. Today, the information on the card is read by automated teller machines (ATMs), store readers, bank and is also used in online internet banking system. They have a unique card number which is of utmost importance. Its security relies on the physical security of the plastic card as well as the privacy of the credit card number.

There is a rapid growth in the number of credit card transactions which has led to a substantial rise in fraudulent activities. Credit card fraud is a wide-ranging term for theft and fraud committed using a credit card as a fraudulent source of funds in a given transaction. Generally, the statistical methods and many data mining algorithms are used to solve this fraud detection problem. Most of the credit card fraud detection systems are based on artificial intelligence, Meta learning and pattern matching. The

Genetic algorithms are evolutionary algorithms which aim to obtain the better solutions in eliminating the fraud. A high importance is given to develop efficient and secure electronic payment system to detect whether a transaction is fraudulent or not. In this paper, we will focus on credit card fraud and its detection measures.

A credit card fraud occurs when one individual uses other individuals’ card for their personal use without the knowledge of its owner. When such kind of cases takes place by fraudsters, it is used until its entire available limit is depleted. Thus, we need a solution which minimizes the total available limit on the credit card which is more prominent to frauds. And, a Genetic algorithm generates better solutions as time progresses. The complete emphasis is given on developing efficient and secure electronic payment system for detecting the fraudulent.

PROBLEM STATEMENT:

The Credit Card Fraud Detection Problem includes modeling past credit card transactions with the knowledge of the ones that turned out to be fraud. This model is then used to identify whether a new transaction is fraudulent or not. Our aim here is to detect 100% of the fraudulent transactions while minimizing the incorrect fraud classifications.

DATASET:

This dataset is taken from kaggle.(

Observations made on data set

1. The data set is highly skewed, consisting of 492 frauds in a total of 284,807 observations. This resulted in only 0.172% fraud cases. This skewed set is justified by the low number of fraudulent transactions.
2. The dataset consists of numerical values from the 28 ‘Principal Component Analysis (PCA)’ transformed features, namely V1 to V28. Furthermore, there is no metadata about the original features provided, so pre-analysis or feature study could not be done.
3. The ‘Time’ and ‘Amount’ features are not transformed data.
4. There is no missing value in the dataset

DATA PREPARATION:

## Overview

This section examines summary statistics for the fraud\_data.csv dataset. It then splits the dataset into training and test sets to train several models and evaluate their effectiveness in detecting fraud in credit card transactions. This project focuses on selecting the appropriate model evaluation metrics when classes are imbalanced.

Construction of the model and analysis are presented in the next section.

The analysis for this project was performed in Python.

This dataset is an imbalanced dataset

The data is in good shape, that is, there is no missing.

Data Clean (missing impute / outliers / normalize)

Feature Engineering (Categorical variables, transformation, correlation analysis)

The provided data is imbalanced, with positive rate around 0.17%.

If we use this data directly to feed the model, the model will prefer to predict all as 0 for a high accuracy of 0 prediction.

Imbalanced data with very low proportion of positive signals.

There are 284315 rows (99.8%) with y = 0, and only 492 rows (0.172%) with y = 1. So it is very imbalanced data.

Usually we have these methods to deal with imbalanced data: 1. Collect more data 2. Over-Sampling or Down-Sampling 3. Change the prediction thresholds 4. Assign weights

Over-sampling the data to get a balanced proportion of positive/negative valuesBefore oversampling, we will first take a random sample as Test data.

For non-fraud transactions, the average amount is 88. For fraud transactions, the average amount is 122. So, in average there will be 122 loss for a fraud. Suppose for each transaction, the company can get 2% transaction fee. That is, the average is 88\*2% = 1.76.

That means: if we predict a non-fraud as fraud, we might loss 1.76. However, if we miss to detect a fraud transaction, we will loss about 122.

Later we have to standardize the data using standard scaler

Since this is Fraud detection question, if we miss predicting a fraud, the credit company will lose a lot. If we miss predicting a normal transaction as Fraud, we can still let the exprt to review the transactions or we can ask the user to verify the transaction. So in this specific case, False Positive will cause more loss than False Negative.

We will use the imbalanced data directly in logistic regression. That is, the positive rate is about 0.172%. Accuracy is not good since if all predicted as 0, the accuracy for 0 is very high. So, here recall, precision, roc and confusion\_matrix are listed to compare model performance.