Capstone Project Proposal – Credit Card Fraud Detection

Domain Context:

This project is intended to build a machine learning mechanism to detect fraud in credit card transactions. This has been done over several years as is evident from the work of Chan and Stolfo – Towards Scalable Learning with Non-Uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection available at http://www.aaai.org/Papers/KDD/1998/KDD98-026.pdf

Industry estimates indicate that 0.1% of credit card transactions globally are fraudulent [Source: Wikipedia]. Since typical fees charged by credit card companies and online payment providers are between 2% to 4% of total transaction values, even a small improvement in fraud detection contributes significantly to improved profitability and increased competitiveness in the marketplace.

In the last few years, there has been increased competition in the payment space with the proliferation of online payment service providers such as Paypal, Stripe, Amazon Pay, etc. that completely disintermediate credit card companies. As a result, there is increased pressure on profitability, which in turn requires credit card issuers to improve fraud detection either in an automated fashion using machine learning or using analyst teams.

Increasing fraud ultimately results in increasing costs for legitimate card holders, since companies need to recoup these costs in some way. By the same token, companies that do better at fraud detection are able to charge lesser fees to card holders and be more competitive in the marketplace, both to merchants that use these card providers as well as individual card holders.

More broadly speaking, these algorithms can be used for other imbalanced data sets that occur widely in nature and business, some of which are even more imbalanced e.g. factory production error rates (about 0.1%), server failure rates, cancer detection (0.45%), detecting oil slicks from satellite ocean images, etc.

Dataset:

We will use the dataset for credit card fraud available from Kaggle. This is available at the link - https://www.kaggle.com/dalpozz/creditcardfraud. This consists of real data from European cardholders in 2013. The data consists of 30 total features [i.e. 28 principal components identified through PCA, transaction amount, time [elapsed since first transaction] along with the Class for each transaction - 1 indicating fraud and 0 indicating normal transaction. There are a total of 492 frauds out of 284,807 total transactions in the data provided, hence this is a highly imbalanced dataset with the positive class amounting to only 0.172% of all transactions.

This dataset has been anonymized and only the principal components are provided, except for time and transaction amount. Hence, there is no potential for increasing the data volume or additional features. Since only the principal components are provided, there is also no intuition that can be derived

by looking at the principal component features themselves. However, the amount and time duration can potentially provide some clues to fraud patterns.

This dataset will be split into training, cross-validation and test datasets. In order to ensure that the imbalance is preserved, I plan to use StratifiedKFold with 10 folds i.e. a variation on the k-fold validation based on https://stackoverflow.com/questions/32615429/k-fold-stratified-cross-validation-with-imbalanced-classes

The difficulty in addressing this problem space is typically availability of training data. Companies are understandably unwilling to share their data considering privacy for their customers as well as making datasets available to third parties with mala fide intentions that can determine how to beat fraud detection algorithms that the companies have invested in developing over many years.

Problem Statement:

The primary goal of this project is to build a binary classification algorithm (i.e. transactions are fraudulent or not fraudulent), considering the transaction dataset as the input. In addition, a secondary objective is to determine the best performance metrics to be used to analyze.

Evaluation Metrics:

The metric that is used to measure classifier performance is accuracy – however, this does not apply in this scenario since the percentage of fraud transaction is so small i.e. **only 0.1727 % of the entire transaction dataset** – this is typical of the industry as mentioned earlier. A classifier that predicted zero fraud across the board would still have an accuracy above 99.9% and not be performing well at all.

The relevant measures that can be used to determine how well the classifier is performing in imbalanced data scenarios are Recall accuracy (i.e. % of fraud transactions captured by the model), ROC AUC score, fBeta measure, the Mathews Coefficient and the Kappa metric.

As the goal of the project is to identify all fraudulent transactions, the objective of this project is to achieve **a recall accuracy of 90% or better**. Along the way, we will also report and examine the various applicable performance metrics listed above that are applicable for imbalanced scenarios.

Solution Statement:

Supervised learning classifier algorithms -Random Forest, Logistic Classifier and Naïve Bayes with performance metrics of AUC score and Recall on the test results will be used to determine which performs best for further tuning. Depending on results achieved, we will examine neural networks or unsupervised algorithms such as anomaly detection to achieve desired performance.

The solution will consist of a series of steps that when executed in sequence enable the user to identify which transactions in a given data set are fraudulent. This will consist of the following steps.

1. Prepare the dataset by pre-processing the data for example, by scaling the features.
2. Train and validate the model with a subset of the data

3. Test the model with a different subset of the data and report the results.
4. Evaluate the results vs. the benchmark results and the project goals.
5. If the results are not satisfactory e.g. the recall accuracy is below 90%, the iteratively perform steps 2,3 and 4 until the desired results are achieved.

Benchmark Model:

This is one of the most researched areas in machine learning given the real-world applicability. Traditional models have been rule-based data mining approaches that look to identify human comprehensible fraud patterns that can be applied.

A benchmark model using the same dataset published on Kaggle shows a recall accuracy of 93% which means that it correctly identifies 93% of all fraudulent transactions in the dataset. [https://www.kaggle.com/joparga3/in-depth-skewed-data-classif-93-recall-acc-now]. In addition, the AUC score will be used to determine performance.

Project Design:

Exploration and Visualization:

The first approach will be to examine the dataset by randomly sampling a few samples. Second, we will visualize the dataset to understand variations across time and transaction amount, for instance. using libraries such as seaborn and matplotlib.

Data Preparation:

The next step will be to pre-process the data to make it suitable for classification. This will include scaling the features to ensure that the features with larger values do not have a disproportionate impact on the classifier outputs. The data will be separated into train, cross-validation and test datasets. Training will be used to train the models, cross-validation to evaluate the model and determine best performing models. The test dataset will be used to determine overall model performance.

Performance Baseline:

We will then use standard logistic classifiers such as Logistic Regression, Naïve Bayes and Random Forest to perform a quick and dirty analysis. The output of this step will be to identify one of the classifiers to use for further analysis. This also provides a baseline for the applicable performance metrics.

The following approaches will be used to improve/ tune the algorithm to achieve desired performance. These can be either data-based methods and/or algorithmic methods as described below.

Data-based Methods:

These modify the data fed into the classifier by resampling it and essentially increase the minority class either by under sampling the majority class i.e. removing transactions that are not fraud or oversampling the minority class (increasing the % of the fraud class. There are more sophisticated

methods such as nearmiss, condensed nearest neighbor, etc. that can also be tried to improve performance.

Algorithmic Methods:

Here, we will use ensemble methods such as boosting algorithms (e.g. Adaboost, XGBoost) that essentially extract information from prior misclassified data to improve the algorithm in the next step.

We expect a combination of the above to enable us to achieve desired performance metrics. If we have still not achieved desired performance, we will explore anomaly detection and neural networks in addition to the above.

Additional References Used:

- https://www.youtube.com/watch?v=X9MZtvvQDR4 - 2015 presentation showing approach from industry used for fraud detection.
- The LIME framework can be used to understand which features are contributing to the prediction. [reference: https://www.youtube.com/watch?v=HcaAKI1tVGM].