Yash Murthy

July 20, 2022

Comp 4449

Midterm Paper

**Dataset Description**

The dataset, obtained from Kaggle, contained credit card transaction information from European countries over a two day period in September 2013. Altogether, the dataset contained 284,807 transactions. Exactly 492 of those transactions were classified as fraudulent. With only 0.172% of the total dataset bring classified as fraudulent, there is clearly a strong imbalance of non-fraudulent vs. fraudulent charges. This will be addressed in my analysis, as I explored two techniques to handle the issue of imbalanced data.

The dataset was already preprocessed using PCA transformation and contained no null values. All data was numerical and the majority of the columns were already standardized. There were 28 variables, V1-V28, that were the principal components obtained from the PCA analysis. The description of these 28 variables was withheld due to confidentiality restrictions. There were also two features that were not transformed, ‘Time’ and ‘Amount’. The amount variable was the total transaction amount and time meant the time elapsed from the original transaction in the dataset. Finally, we had the dependent ‘Case’ variable. This indicated a fraudulent charge,1 , or non-fraudulent charge, 0. 1

**Data Preparation**

As mentioned, most of the variables were obtained through PCA and already preprocessed. I chose to drop the time feature as I did not think it would be as important since we only have a two day window of transactions. I did ended up dropping approximately 10,000 rows of data after running a drop\_duplicates pandas function to avoid repetition in the analysis. I ran a heatmap to visualize the correlation among the target variable and the features, with the initial hope of dropping any variables that had a weak correlation with the outcome variable,  (< |0.1|). However, too many features were under this threshold so I decided to keep all variables in the analysis. The ‘Amount’ variable, which was not normalized, had a few large outliers that led to a large variance. I chose to apply two scalar techniques to the Amount, Standard Scaler and Log Scaler. The results of the scalar are shown below.

Chart, box and whisker chart

Description automatically generated

We see that there is minimal difference after the log scalar, so I used the log scaled data during my analysis portion.

**Analysis, Results & Discussion**

I chose three classification models for my analysis, Logistic regression, Decision Tree, and Random Forest. These techniques are popular in binary classification problems. I also used two packages, SMOTE (Synthetic Minority Oversampling Technique) and ADASYN (Adaptive Synthetic), to address the heavy imbalance in the outcome variable.

I began splitting the data in an 80-20 train-test split and ran these through the three classification models. I obtained classification reports of the three and looked at F1 score to decide which was the best model. In the case of imbalanced datasets, accuracy is not the best method to measure performance, as in this case if I created a model that only classified transactions as non-fraudulent, it would be accurate over 99% of the time. Both the decision tree and random forest performed at approximately 80% for the F1 score, and the logistic regression had a score of approximately 70%.

Text

Description automatically generated

Additionally, I used a confusion matrix for all three models in order to visualize the results.

Chart, box and whisker chart

Description automatically generated

It seems that there were few false positives in each of the models, and number of false negatives being slightly higher.

The final step of my analysis involved the SMOTE and ADASYN techniques to deal with the imbalance of the Class variable. Both of these techniques were new to me and it was interesting to learn the idea behind the algorithm. SMOTE is an oversampling technique that creates synthetic examples in minority class which are similar to those that already exist, versus simply replicating the existing data. The algorithm contains five main steps. Identifying the feature vector and its nearest neighbor, taking the difference between the two points, multiplying the difference with a random number between 0 and 1, identifying a new point on the line segment by adding the random number to the feature vector, and finally repeating the process for already identified feature vectors.2 ADASYN also generates synthetic data by using weighted distribution for the minority class examples according to their difficulty of learning. This can help reduce bias introduced by class imbalance and shift the classification decision boundary towards more difficult examples in an adaptive manner. 3

Upon entering the dataset into these algorithms, I obtained approximately 220,000 synthetic datapoints within the training set. I ran the new data into the original decision tree and random forest models.

Chart, box and whisker chart

Description automatically generated

We can see that the F1 score fell significantly from the original models, from .78 to .35. Similar results were seen in the decision tree, which fell from .80 to .21. Similar drop-offs were seen in the ADASYN results.

Chart

Description automatically generated

We see that the F1 score declined even more, from ,80 to .02 in the decision tree and from .78 to 0.06 in the random forest model. Both the SMOTE and ADASYN additional data led to a higher prevalence of false positive results, meaning that the transaction was classified as fraud when it was in fact legitimate. It was also interesting that the total number of false negatives decreased in both the SMOTE and ADASYN data. The false negatives in this case, mean that the transaction was classified as legitimate when it was in fact fraudulent. The false negative, type II error, is a bit more impactful as it lets fraudulent charges go undetected, which can be detrimental for both the individual and the credit card company. A type I error in this case means that the user may be alerted of a legitimate charge that was flagged as fraud, and then have to discuss it with the credit card company. The additional data provided by the SMOTE and ADASYN algorithms provided a decrease in the more detrimental error in this case.

The overall findings of this project were interesting to say the least. It is difficult to deal with data with such a heavy imbalance. SMOTE and ADASYN, along with under and oversampling techniques, can help address this class imbalance issue. For future research, I would hopefully explore a similar dataset with a bit more transparency. Although the data was already transformed using PCA, it would be nice to see the raw data so I have a better understanding of the given variables. Additionally, I would run additional models and visualize the AUC-ROC plots to further evaluate my models. I also read about the XG Boost classifier which I learned about a bit too late in the project timeline. Dealing with imbalanced data leads to issues when creating predictive models as such. A further understanding of the data may lead to better results if tackling a similar problem in the future.

**Works Cited**

Nian, Rui. “An Introduction to ADASYN (with Code!).” Medium, Medium, 13 Dec. 2019, https://medium.com/@ruinian/an-introduction-to-adasyn-with-code-1383a5ece7aa#:~:text=ADASYN%20(Adaptive%20Synthetic)%20is%20an,%E2%80%9Charder%20to%20learn%E2%80%9D%20examples. (3)

Satpathy, Swastik. “Smote: Overcoming Class Imbalance Problem Using Smote.” Analytics Vidhya, 6 Jan. 2021, https://www.analyticsvidhya.com/blog/2020/10/overcoming-class-imbalance-using-smote-techniques/. (2)

ULB, Machine Learning Group -. “Credit Card Fraud Detection.” Kaggle, 23 Mar. 2018, https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud. (1)