Machine Learning Engineer Nanodegree

Capstone Project

Mohammed AlAyyaf
February 23rd, 2019

I. Definition

Project Overview

Fraud detection is a challenging problem. The fact is that fraudulent transactions are rare; they represent a very small fraction of activity within an organization. The challenge is that a small percentage of activity can quickly turn into big dollar losses without the right tools and systems in place. Criminals are crafty. As traditional fraud schemes fail to pay off, fraudsters have learned to change their tactics. The good news is that with advances in machine learning, systems can learn, adapt and uncover emerging patterns for preventing fraud.

Here's an interesting paper that discusses credit card fraud detection:
https://www.aaai.org/Papers/KDD/1998/KDD98-026.pdf

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 as much of the fraudulent transactions as possible, while minimizing the incorrect fraud classifications.

Given the unbalanced nature of this dataset, we most likely need:

- to find a way to rescale/resample the dataset to be more balanced and in a way that allows us to approach the problem as a normal balanced classification problem, I will need to look into resampling methods/techniques that would best suit this problem, most probably it would be under-sampling, which deletes instances from the over-represented class in order to find an almost 50/50 representation for both classes.

- After that, I will test some supervised learning algorithms to find the best possible algorithm, while not consuming much computation power.

Metrics

Since our target is to catch the fraudulent transactions, and hence we would rather wrongly catch suspicious transactions than to mistake fraudulent transactions as normal

ones, then Recall (the ratio of correctly predicted positive observations to all the observations in actual class) should be our main metric, which can be represented as true positives / (true positives + false negatives). Yet, we will still aim to have a somewhat accurate model as much as possible.

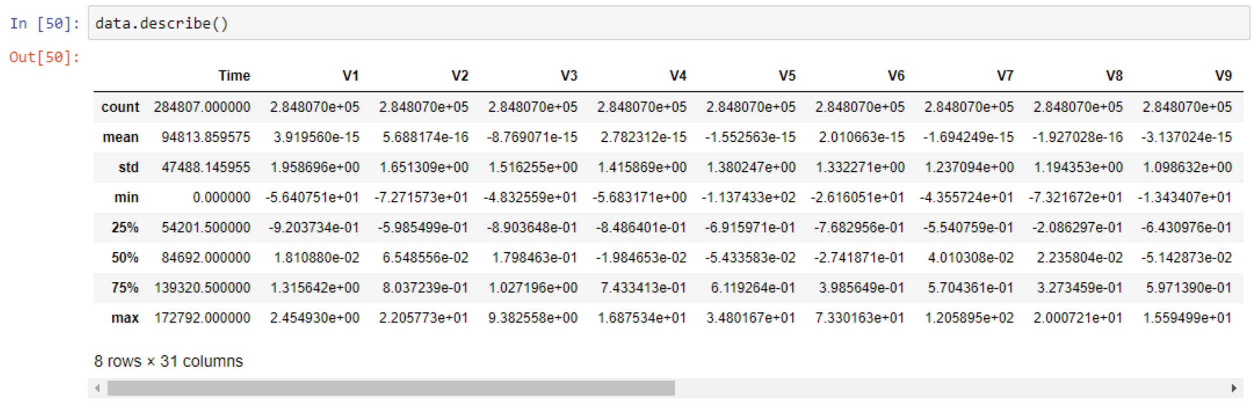
II. Analysis

Data Exploration

The datasets contain transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

It contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we cannot provide the original features and more background information about the data. Features V1, V2, ... V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-senstive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

Below is some statistics about the dataset:



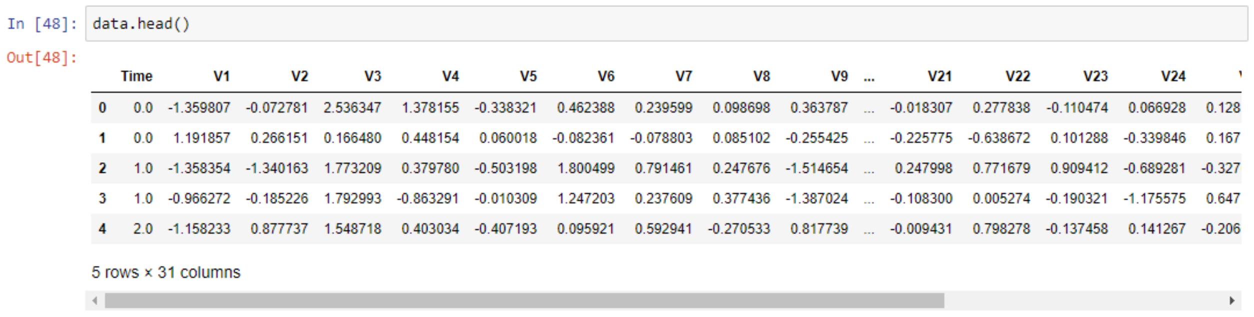
```
In [50]: data.describe()
```

Out[50]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05
mean	94813.859575	3.919560e-15	5.688174e-16	-8.769071e-15	2.782312e-15	-1.552563e-15	2.010663e-15	-1.694249e-15	-1.927028e-16	-3.137024e-15
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.380247e+00	1.332271e+00	1.237094e+00	1.194353e+00	1.098632e+00
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.137433e+02	-2.616051e+01	-4.355724e+01	-7.321672e+01	-1.343407e+01
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.915971e-01	-7.682956e-01	-5.540759e-01	-2.086297e-01	-6.430976e-01
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.433583e-02	-2.741871e-01	4.010308e-02	2.235804e-02	-5.142873e-02
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.119264e-01	3.985649e-01	5.704361e-01	3.273459e-01	5.971390e-01
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.480167e+01	7.330163e+01	1.205895e+02	2.000721e+01	1.559499e+01

8 rows × 31 columns

And here is a sample of the dataset:



```
In [48]: data.head()
Out[48]:
```

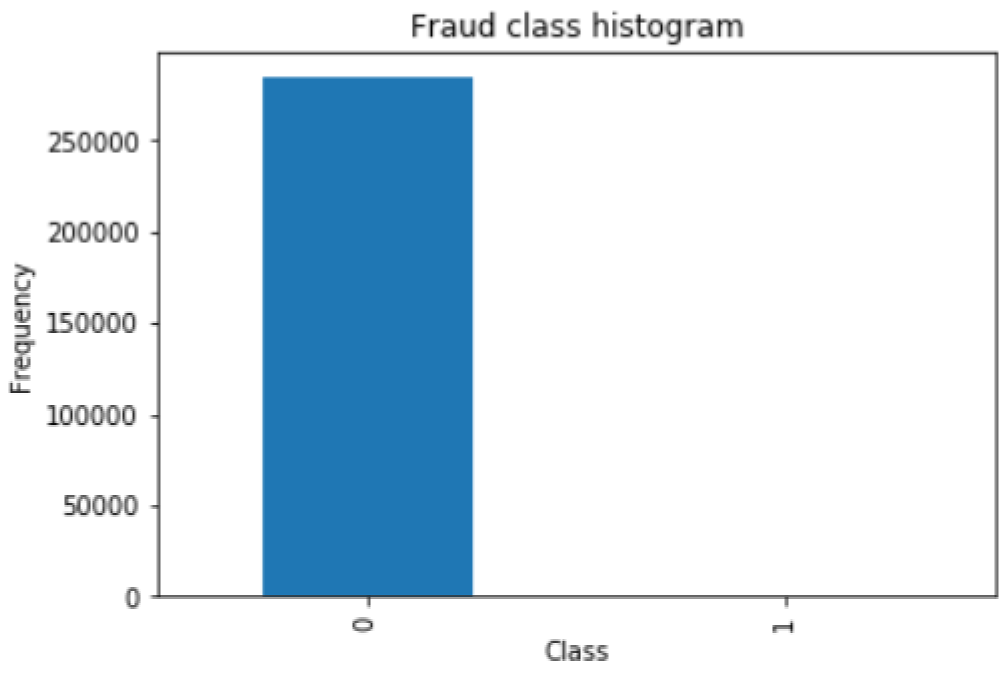
	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.128
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.167
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.327
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.647
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.206

5 rows × 31 columns

Data Source: https://www.kaggle.com/mlg-ulb/creditcardfraud

Exploratory Visualization

```
Count of Normal Transactions: 284315
Count of Fradulant Transactions: 492
```



The above graph shows the main character of this dataset (how incredibly unbalanced it is), Most of the transactions were Non-Fraud (99.83%) of the time, while Fraud transactions occurs (017%) of the time in the dataset, which is also the main challenge to consider when analyzing this dataset. This also emphasize on the importance of finding a way to handle this skew in the data in order to be able to fairly build the model around both class es equally

Algorithms and Techniques

The main challenge in this dataset is how to be able to build a predictive model on such a highly skewed dataset, therefore I had to work on a technique for best handling this skewed nature and to be able to work on the data as if it was normally distributed, and my choice fell on the "Under-sampling" technique, which deletes instances from the over-represented class in order to come up with almost 50/50 representation for both classes. Another technique used to handle this imbalance in the data, is choosing "Recall" (the ratio of correctly predicted positive observations to all the observations in actual class), as our evaluation metric, especially since we would rather wrongly catch suspicious transactions than to mistake fraudulent transactions as normal ones. As for the prediction algorithms used for detecting fraudulent transaction, I plan to start with "Decision Tree" and "Logistic Regression", Decision trees classify the examples by sorting them down the tree from the root to some leaf node, with the leaf node providing the classification to the example. Each node in the tree acts as a test case for some attribute, and each edge descending from that node corresponds to one of the possible answers to the test case. This process is recursive in nature and is repeated for every subtree rooted at the new nodes. While Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Unlike linear regression which outputs continuous number values, logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes.
It is the go-to method for binary classification problems (problems with two class values). In this post you will discover the logistic regression algorithm for machine learning.
I chose to start with these two algorithms given their ease and straight-forwardness, and their suitability for our case, before using any more complicated algorithms that would be beyond my available computational power.

Benchmark

Looking at the other scores achieved from other Kaggle contributors, I would say that recall and f1-score should be no less than 80%, especially as this is a case where it is considered sensitive to miss any fraudulent transactions.

III. Methodology

Data Preprocessing

First I normalized the Amount column, since the Amount column was not in line with the rest of the features. Then I dropped the Time and the not-normalized Amount column, since they won’t be used in our analysis and would create additional noise. After that, I implemented the under-sampling technique on the dataset, which will allow the dataset to have an equal sample of instances under each class.

Implementation

As planned, I started with the solving the imbalanced nature of this dataset, so I used the under-sampling technique to achieve a 50/50 ratio distribution between the two classes, I did so by first shuffling the dataset before creating the subsamples, to ensure minimum information loss from deleting the instances of the over-represented class, and used the pandas function “Sample” to achieve this, before manually slicing the dataset to two equally-sized datasets for each class, and then re-join them with the “Concat” function, then I reshuffled the new dataset to ensure more generalization.

After that, I carried out with splitting the dataset to train and test splits, using Sklearn’s “train_test_split”, then I started creating the classifiers also using Sklearn, but with adding “GridSearchCV” for parameter tuning. Finally, I used Sklearn’s “classification_report” function to evaluate our model with Precision, Recall and F1 score.

Probably the main complication faced during creating the model was the overfitting in the initial model before using the under-sampling approach, specially that the issue was not a matter of accuracy so to be addressed by enhancing or re-tuning the model, but thankfully my preparation and initial research on how to handle such datasets paid off.

Refinement

First I attempted to create model without the using the under-sample dataset, just to observe the contribution it will add to the model. And as expected, the model overfitted with the under-sampled dataset, where it had 100% score in all metrics in the test set. And after using the new under-sampled dataset, we had high scores on both the test and train sets. I looked up some hyper parameters sets to use for Grid Search, I needed to tradeoff between getting the best possible parameter set that will achieve the highest possible score, while maintaining a reasonable run time. Therefore, I aimed to include the most important parameters in both classifiers as a minimum requirement, some of the important parameters I sought to include are: criterion, max depth, and min samples leaf for the Decision tree classifier, and penalty, class weight and C for the Logistic regression model.

IV. Results

Model Evaluation and Validation

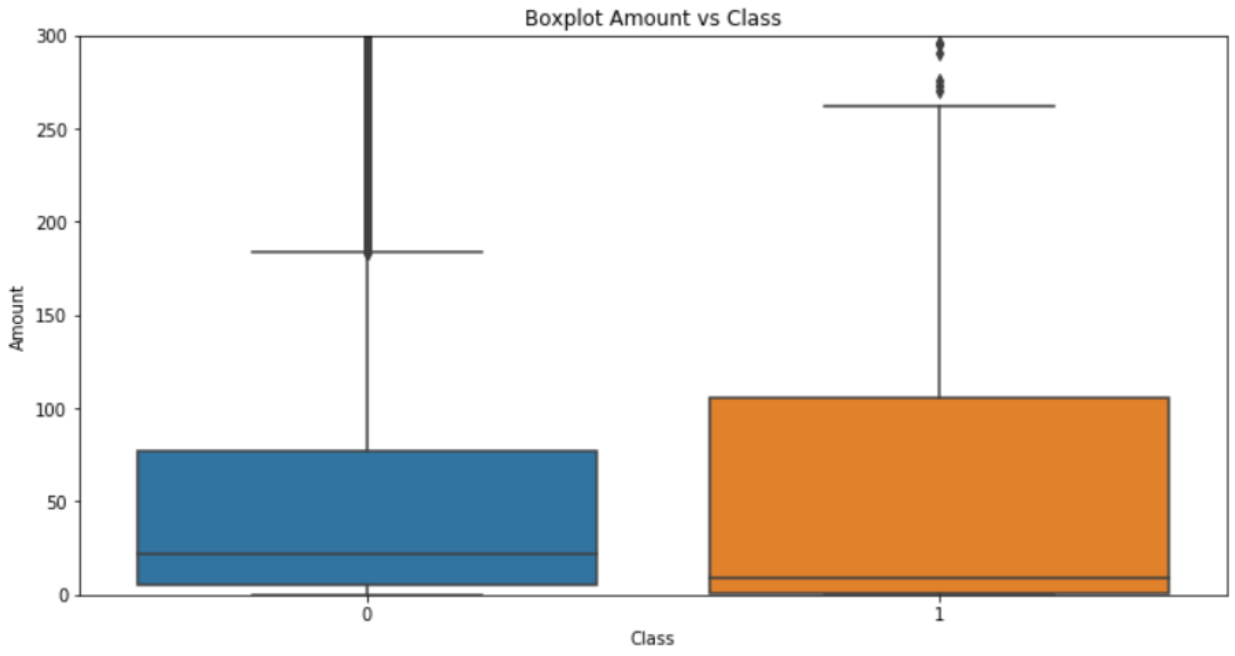
The final model is a Logistic Regression model, with the following parameter set (C= 0.1, class_weight= 'balanced', penalty= 'l1'), this parameter set was chosen based on running “GridSearchCV” function for hyperparameters tuning. This model achieved 0.96, 0.95 and 0.95 on precision, recall and F1 score respectively, and given the nature of our problem, and our benchmark, this model is capable of detecting a fraudulent transaction with a hi gh degree of accuracy. This fact that the scores for both training and testing sets are alm ost equal, is a strong sentiment that the model is considered robust and can generalize t o new data.

Justification

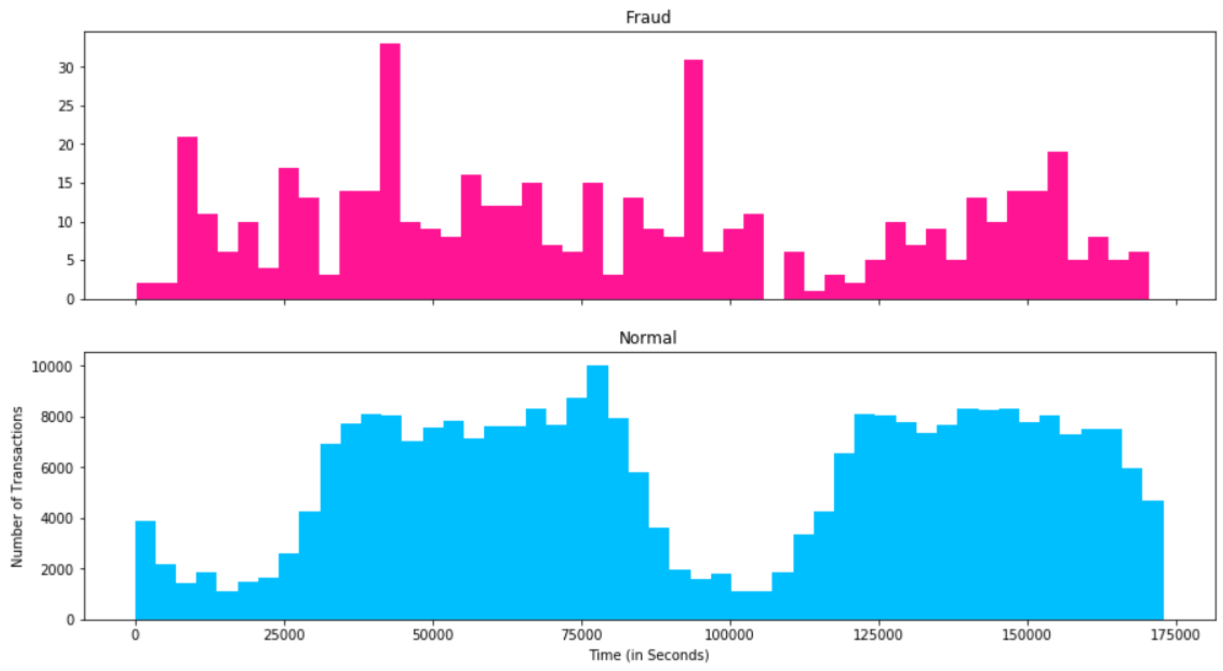
The developed model using Decision tree classifier achieved 92% in precision, recall and F1 score, while the Logistic regression model achieved an enhanced score of around 95 % in the same metrics. These scores are considered well above the established benchma rk and are somewhat good enough to solve the problem.

V. Conclusion

Free-Form Visualization



The above graph shows the relationship between the “Amount” and “Class” variables, in which it can be noted fraudulent transactions are much higher in amount than non-fraudulent ones.



The second graph shows three variables from our dataset (the Class, the Time and number of transactions), it can be noted that there is some sort of quiet period around 9000 second for normal transaction, while there is a spike in the number of fraudulent ones in the same period.

Reflection

In this project I attempted to suggest a solution for the credit card fraud problem, where only a small portion of the transactions are fraudulent ones, I used the under-sampling technique to address this issue (the imbalanced nature of the problem), and then used two famous supervised learning algorithms (Decision tree and Logistic regression) to build a predictive model that achieved a good score that helps to identify and detect fraudulent transactions. the main challenge of this problem was finding a method/technique that will help us deal with the problem as an equally labeled classes, as this not addressing this issue resulted overfitting. What I found interesting working through this project, is how amazingly these simple algorithms are capable of detecting fraud from only looking at transactions, as I previously assumed that machine learning algorithms will need a huge magnitude of features and feature engineering in order to achieve high results.

Improvement

A suggested area of improvement would be to try out more complex algorithms and see how would they work, as I assume that a dataset of this size would work using a stronger, more complicated algorithms, especially if some feature engineering has been implemented. Another area of improvement could be working on additional/larger parameter sets as inputs to the grid search function for parameter tuning, which would then be able to produce possible a better parameter set, and in turn a better score.