Credit card fraud detection using a three-layered neural network and density based anomaly detection algorithm (August 2018)

Author: Asvinth A,
AITS Machine Learning Engineer Intern – https://ai-techsystems.com/

***Abstract* — Credit card frauds are increasing day by day. People are getting nervous about the unknown activity happening in their accounts. Tracing these kind of frauds from a huge database has become a headache for the banks. This paper focus on finding those irregular activities and comparing it using a three layered neural network with a density based anomaly detection algorithm. For this work, machine learning group-ULB has provided a dataset which consist 284,807 transaction happened within two days. The proposed model uses a three layered neural network and knn classifier for finding the frauds and compares their efficiency.**

***Index Terms*— KNN Classifier, Neural network, SMOTE, Under sampling**

I. Introduction

Fraud detection has became the hottest topic in banking sectors. Why it became so difficult to find out this fraud is because of the high transaction rate happening in each second. It is very difficult to find out the anomalies residing inside such a huge data. Finding out how efficient is neural network and k nearest neighbor is the aim of this paper.

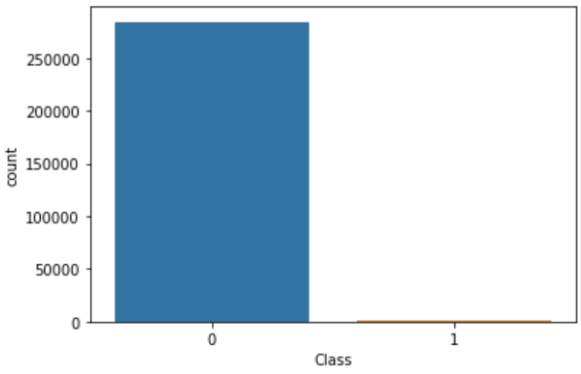
The dataset consist of 492 frauds out of 284,807. As it is highly unbalanced models may not work accurately. There are two different balancing methods used in this paper under sampling method and the SMOTE method for over sampling. Due to privacy reasons dataset does not include the transaction details. All of the features went through data dimensionality reduction technique called Principle component analysis.

This paper is further divided into 3 sections data exploration, Pre-processing, Model evaluation and analysis.

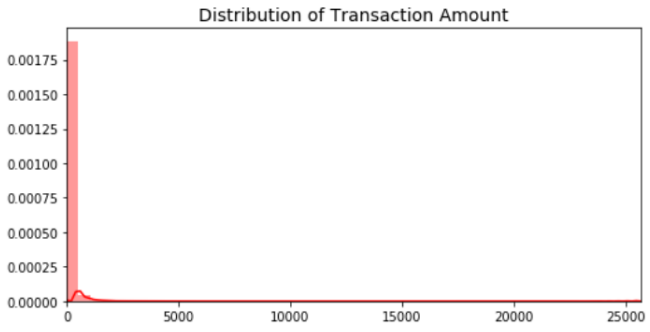
.

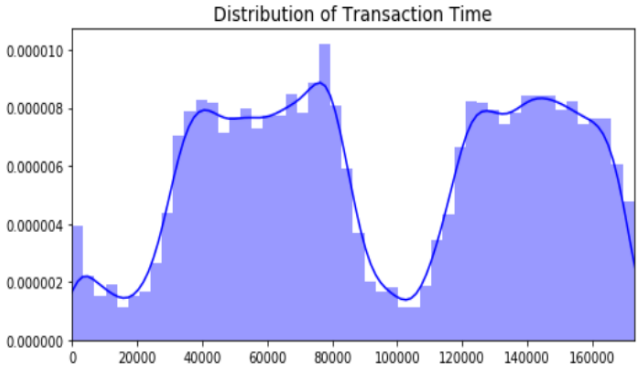
II. Data Exploration

The dataset consist of 99.83% of non-fraud activities and 0.17 % of fraud activities. This imbalance may cause our model to overfit since it will "assume" that most transactions are not fraud.



The number of fraud is only 492. The next two attributes are dates and the time. There was a need of rescaling the data.



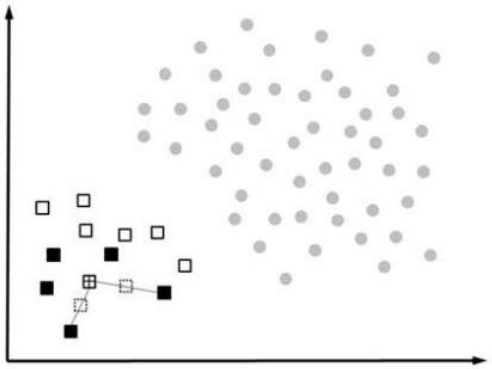


The data is highly skewed and scaling will help in making the model more accuracy

III. PRE-PROCESSING

The dataset is highly imbalanced and there are attributes which we need to scale it. Time and amount should be scaled as the other attributes. For data scaling robust scalar function is used from the sklearn library.

The next thing is making the dataset uniform. Two approaches has been implemented first one is the random under sampling, and next one is the oversampling technique using SMOTE. Under sampling is a technique which helps to reduce the attribute count by removing the over populated values. After preprocessing the data got reduced to 492 values for both classes. There is a high risk of implementing this technique, though we are reducing the data there is high chance of loosing important data and model may not work accurately. Oversampling is another technique which we will over populate the existing data for making it as 50-50 distribution. SMOTE stands for Synthetic Minority Oversampling Technique. It creates synthetic observations using the exiting the minority.



For each of the points in minority class the smote calculates the k nearest neighbors. The algorithm takes samples of the feature space for each target class and its nearest neighbors, and generates new examples that combine features of the target case with features of its neighbors.

IV. MODEL EVALUATION

The dataset tested using 3 cases. First is with out any under sampling or oversampling. Second one with under sampling and the third is with over sampling.

A. Neural network

The algorithm uses a three layered network. The model is simple as it has only one single input layer(number of nodes equal to the number of input features). Including one hidden layer of 32 nodes and output layer with 0/1 as output. For the final output we used binary cross entropy and the optimizer is adam. For implementing this model Keras framework has been used.

B. K nearest neighbors:

This is a lazy learning algorithm in which all of the work happens at the time of a request for prediction. For every similar instance it will map the new point by searching through the entire dataset. The most popular matric used for identifying the similarity is Euclidean distance. There are many other kind of measures such as hamming distance, Manhattan distance, Minkowski distance etc. The value of K found here is by regular trial and error method.

Results:

Case 1:

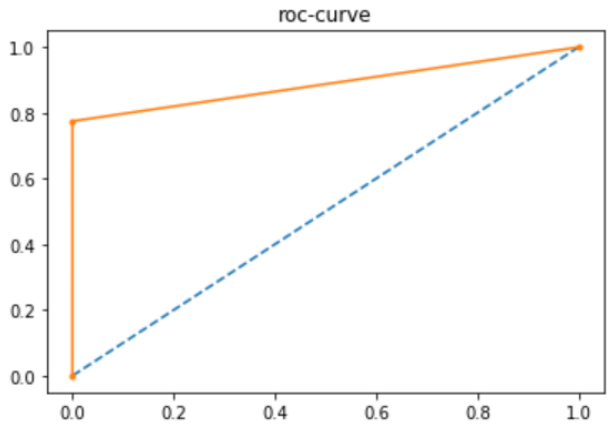
Without preprocessing - KNN

Clssification report:

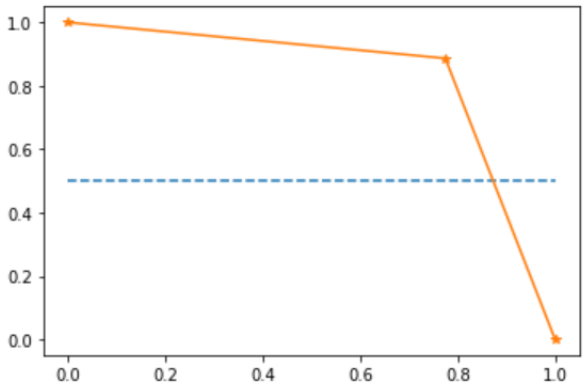
	precision	recall	f1-score	support
0	1.00	1.00	1.00	71091
1	0.79	0.77	0.78	111
accuracy			1.00	71202
macro avg	0.89	0.89	0.89	71202
weighted avg	1.00	1.00	1.00	71202

Confusion matrix: $\begin{bmatrix} 71080 & 11 \\ 25 & 86 \end{bmatrix}$

Roc-curve:



Precision-curve:



Without-preprocessing – NN

Classification report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	71091
1	0.89	0.77	0.83	111
accuracy			1.00	71202
macro avg	0.94	0.89	0.91	71202
weighted avg	1.00	1.00	1.00	71202

Confusion matrix: $\begin{bmatrix} 71068 & 23 \\ 25 & 86 \end{bmatrix}$

The roc-curve and the precision recall curve was exactly same as the curve which e got in KNN.

Case :2

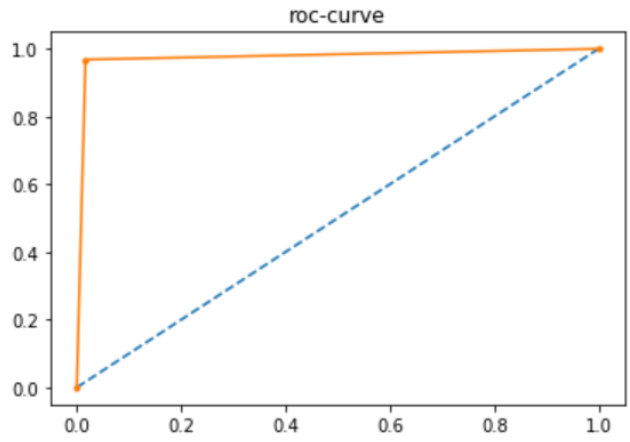
Preprocessed data(Under sampling): Neural Network

Cassification Report:

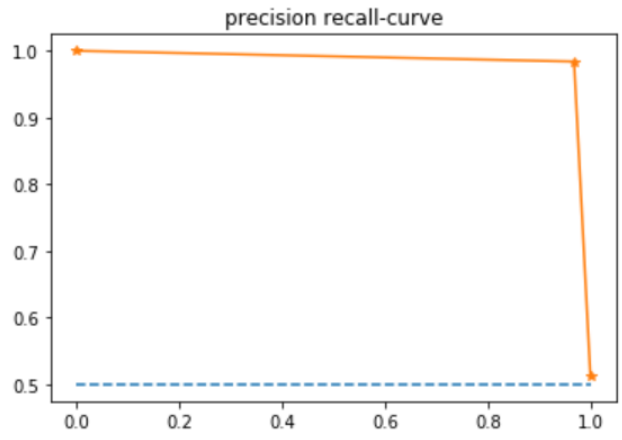
	precision	recall	f1-score	support
0	0.97	0.98	0.98	120
1	0.98	0.97	0.98	126
accuracy			0.98	246
macro avg	0.98	0.98	0.98	246
weighted avg	0.98	0.98	0.98	246

Confusion matrix: $\begin{bmatrix} 118 & 2 \\ 4 & 122 \end{bmatrix}$

Roc-curve:



Precision recall-curve:



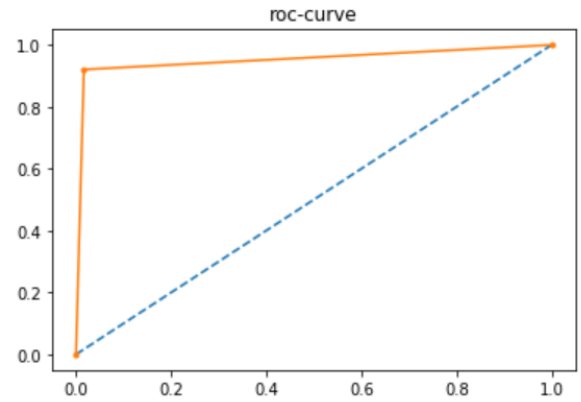
KNN Results:

Classification report:

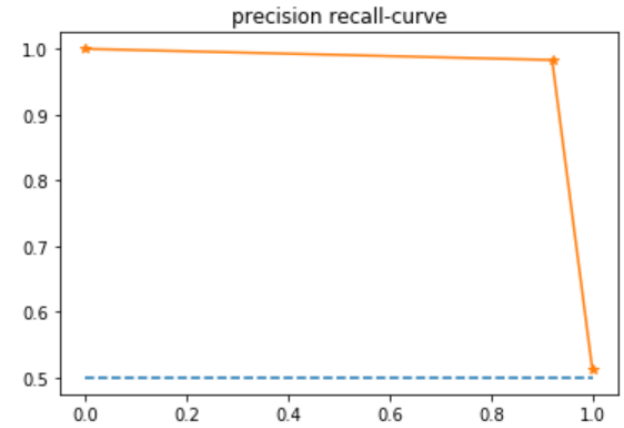
	precision	recall	f1-score	support
0	0.92	0.98	0.95	120
1	0.98	0.92	0.95	126
accuracy			0.95	246
macro avg	0.95	0.95	0.95	246
weighted avg	0.95	0.95	0.95	246

Confusion matrix: $\begin{bmatrix} 118 & 2 \\ 10 & 116 \end{bmatrix}$

Roc_curve:



Precision recall:



Case 3 :

Testing using Oversampled data (SMOTE): NN

Classification report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	70651
1	0.83	1.00	0.91	102
accuracy			1.00	70753
macro avg	0.91	1.00	0.95	70753
weighted avg	1.00	1.00	1.00	70753

Confusion matrix:

Confusion matrix: $\begin{bmatrix} 70575 & 62 \\ 0 & 70639 \end{bmatrix}$

KNN

	precision	recall	f1-score	support
0	1.00	1.00	1.00	70637
1	1.00	1.00	1.00	70639
accuracy			1.00	141276
macro avg	1.00	1.00	1.00	141276
weighted avg	1.00	1.00	1.00	141276

Confusion matrix: $\begin{bmatrix} 70447 & 190 \\ 0 & 70639 \end{bmatrix}$

Conclusion:

Among these results we cannot compare the models with only their accuracy. The metrics which we have used are precision, recall and f1score. With the use of under sampled data, neural network performed well irrespective of the size of the data. When it came to oversampling only the true negative values are getting detected in both the models. The model can be further improved by introducing shuffling to data frames and we can avoid the overfitting by doing the preprocessing techniques during the time of cross validations. Among the two model neural network is slightly higher in prediction accuracy.