Reproducibility in ML
Isolation Forest and Local Outlier Factor for Credit Card Fraud Detection System

by Nadia Hawarri, Ahmad Hendie and Malena Mendilaharzu

ABSTRACT

Due to the rise in credit card use, fraud identification became a crucial issue facing economic institutions. Henceforth, the development of techniques for the detection of fraud became an area of interest for many people. In *Isolation Forest (IF) and Local Outlier Factor (LOF) for Credit Card Fraud Detection System* [1], V. Vijayakumar, Nallam Sri Divya, P. Sarojini and K. Sonika explore the use of two algorithms (IF and LOF) for the identification of fraudulent transactions. The work attempted during this project was to reproduce their findings by replicating, step by step, the experiment they conducted. Therefore, we decided to work with the Kaggle's Credit Card Fraud Detection Dataset as it was the one used in their paper. Further, we decided to explore some alternative metric evaluations and propose improvements for their work. In Accordance with their findings, we discovered that Isolation Forest achieved a better performance than the one of the Local Outlier Factor algorithm.

Introduction

Credit card fraud is becoming a big issue, forcing cardholders to be more aware of checking for unauthorized transactions. However, with the increased number of credit card payments, keeping track and distinguishing between fraudulent and non-fraudulent transactions is becoming a challenge. Fortunately, automated techniques for the detection of fraud are constantly being developed and improved. In this paper, we study how two unsupervised Machine Learning algorithms, Isolation Forest and Local Outlier Factor, can be used for the detection of fraud. Isolation Forest consists of a variation of the Random Forest algorithm that works by isolating anomalies. Local Outlier Factor is a density-based outlier detection technique efficient at the recognition of local outliers. Other than for the detection of fraudulent transactions, both these algorithms can be used to detect other anomalies, such as, detecting when someone logs into your account from another country or even detecting abnormal cells in one's body for medical purposes.

After conducting numerous experiments and tuning the parameters, we were able to replicate the results found in the original paper and we showed that Isolation Forest seemed to be more efficient than Local Outlier Factor for the detection of outliers in the dataset used.

Dataset

For this project, the same dataset as in the original paper, Kaggle's Credit Card Fraud Detection Dataset, was used to conduct our experiments. This dataset consists of the transactions made by credit cards in September 2013 by European cardholders. Credit card purchases are defined by tracking the purchase activity into two types: fraudulent and non-fraudulent purchases. More specifically, the dataset presents the transactions that occurred in two days, where out of 284,807 total transactions, 492 were fraudulent. As expected, and as we can see in the figure below, the dataset is extremely unbalanced. More specifically, the fraudulent transactions account for only 0.172% of the total transactions.

Figure 1.1: Class distribution

The Credit Card Fraud Detection Dataset only contains numerical input variables consisting of the features V1,…,V28 which consist of the principal components obtained with PCA. Due to confidentiality issues, more background information about these features is not provided. The only features that are not transformed are Time and Amount. Although no further background information is provided, a correlation plot (Pearson) is extremely helpful in showing how the features interact with one another.

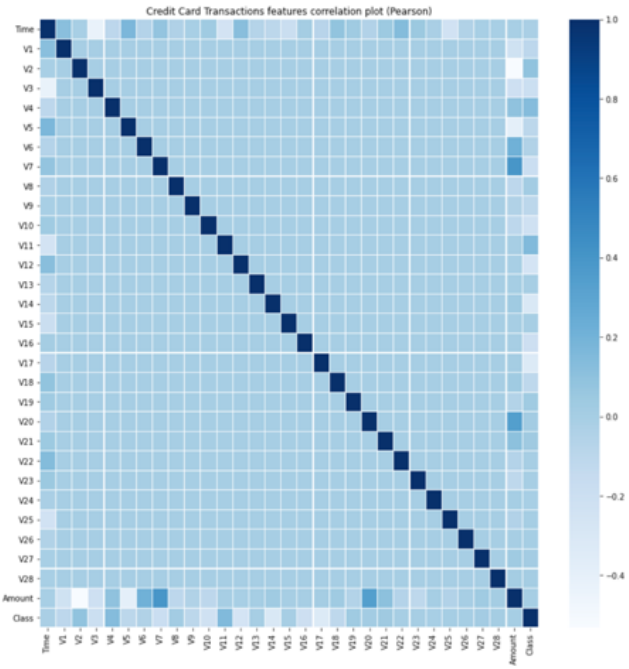

Figure 1.2: Correlation Plot

We began by preprocessing the data to check that no lines with missing values had to be removed before training the model.

Model training

The two algorithms (Isolation Forest and Local Outlier Factor) presented in the original paper were used to train the model.

Isolation Forest

Isolation Forest is an unsupervised learning technique typically used for the detection of outliers. As opposed to the most common procedures that work by profiling normal points, Isolation Forest works by isolating anomalies (points that are very different from the rest of the points). This algorithm is very efficient in practice as it has both a low memory demand and converges in linear complexity time.

Isolation Forest actually consists of a variation of the Random Forest algorithm. Instead of dividing the data until reaching homogeneous partitions, it involves building isolation trees until making partitions such that each datapoint is isolated. The idea behind this is that a regular point is much harder to isolate than an anomalous point.

As it can be seen in the figures below, 4 random partitions were needed to isolate the anomalous point (Figure 2.1) whereas 13 random partitions where needed to isolate the regular point (Figure 2.2). This means that the number of partitions can tell us whether a point is anomalous or not.

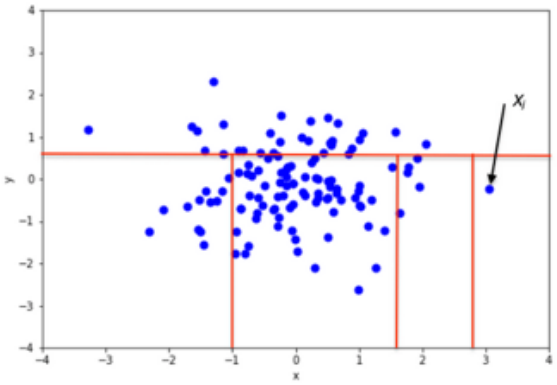

Figure 2.1: Partitions for an anomalous point

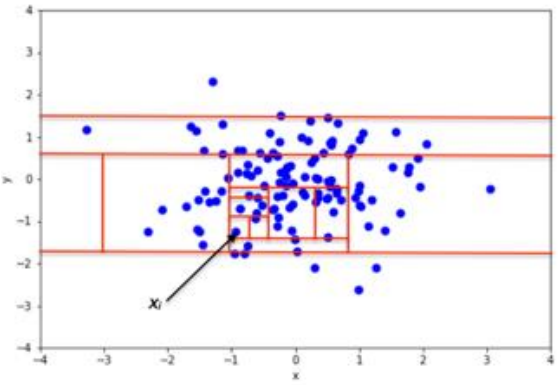

Figure 2.1: Partitions for a regular point

Every time a sample is taken from the dataset, the algorithm builds an isolation tree until the point is isolated. The isolation tree is built by randomly selecting a feature and randomly partitioning along the range. Once we have built a set of isolation trees, the process of prediction involves computing the anomaly score for any new point that we are given. Say we have a new point x and that a sample size m was chosen during the training process, then we can calculate the anomaly score S(x,m) using the following formula:

$$S(x,m) = -2 \cdot \frac{E(h(x))}{c(m)}$$

where h(x) represents the average search height for x from the isolation trees and c(m) is the average path length to find any general node. This formula tells us what the depth of finding this particular point x across all the isolation trees is compared to the depth of finding an average point. If the expected value of finding the depth of x is much smaller than the rest of the points, then it means that x is an anomalous point (fewer partitions needed to isolate hence smaller depth). The prominent parameters for this algorithm are the number of estimators (number of isolation trees), the sampling size m and the contamination value.

Local Outlier Factor

The Local Outlier Factor is a density-based outlier detection technique which contrarily to the distance-based outlier detection techniques, is efficient at spotting both global and local outliers. This algorithm computes the local density deviation of a given data point with respect to its neighbors. More specifically, every point is assigned a LOF score and based on a certain threshold it decides whether a point is an outlier or not. If the LOF score is greater than threshold then it implies that the point is an outlier (has a substantially lower density than its neighbors). Otherwise, the point is characterized as a regular point.

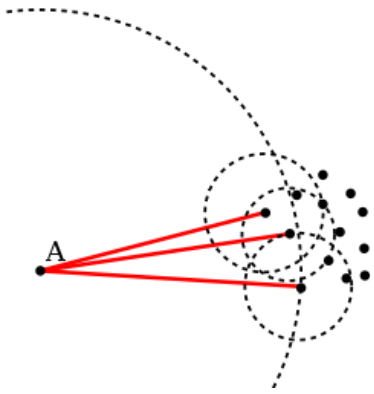

Figure 3.1: Neighborhood density

Results

The original paper did not provide the exact values for each parameter which made the replication of results harder. We first tried running the algorithms with their default values and then we tuned the hyperparameters to obtain more similar results.

The results obtained with the default values were the following:

For IF, the model achieved:

Accuracy_ score = 0.9612791775511924

Roc_AUC_score = 0.9093084587815061

Classification Report:

	precision	recall	f1-score	support
0	1.00	0.96	0.98	71069
1	0.04	0.86	0.08	133
accuracy			0.96	71202
macro avg	0.52	0.91	0.53	71202
weighted avg	1.00	0.96	0.98	71202

For LOF, the model achieved:

Accuracy_ score = 0.9547203730232297

Roc_AUC_score = 0.6583669560991082

Classification Report:

	precision	recall	f1-score	support
0	1.00	0.96	0.98	71069
1	0.02	0.36	0.03	133
accuracy			0.95	71202
macro avg	0.51	0.66	0.50	71202
weighted avg	1.00	0.95	0.98	71202

Hyperparameter Tuning for similar results

We then proceeded to tune the hyperparameters to achieved results more similar to the ones in the paper.

Three hyperparameters had to be tuned for the Isolation Forest algorithm: the number of estimators, the sampling size and the contamination value. Their model achieved an accuracy score of 0.997296 when trained with the Isolation Forest algorithm. By setting the number of estimators equal to 50, the sampling size to 20 and the contamination to 0.001 our model obtained an accuracy score of 0.997457.

Accuracy_ score = 0.9974579365748153

Classification report

	precision	recall	f1-score	support
0	1.00	1.00	1.00	71069
1	0.18	0.10	0.13	133
accuracy			1.00	71202
macro avg	0.59	0.55	0.56	71202
weighted avg	1.00	1.00	1.00	71202

For the Local Outlier Factor algorithm, two parameters had to be tuned: the number of neighbors and the contamination score. When using this algorithm, they obtained an accuracy score of 0.996243. By setting the number of neighbors to 20 and the contamination score to 0.001 we achieved an accuracy score equal to 0.996860.

Accuracy_ score = 0.9968680655037779

Classification report

	precision	recall	f1-score	support
0	1.00	1.00	1.00	71069
1	0.00	0.00	0.00	133
accuracy			1.00	71202
macro avg	0.50	0.50	0.50	71202
weighted avg	1.00	1.00	1.00	71202

Improvement and correction for their work

Note that in the paper the performance of both algorithms was compared to one another by comparing their accuracy score. Indeed “accuracy and its complement error rate are the most frequently used metrics for estimating the performance of learning systems in classification problems” [2]. The reason behind this is that it the accuracy score is easy to calculate and interpret. However, when working with unbalanced datasets (as the one in this paper), accuracy becomes an unreliable measure of model performance. Achieving a 90 or even 99 percent classification accuracy, may be trivial on imbalanced classification problems. As it was shown only 0.172% of the dataset consisted of fraudulent transactions

so even if our model classified everything as non-fraudulent, the accuracy score would still be approximately 99 percent. However, it is clear that in this case the model would not be efficient at distinguishing fraudulent and non-fraudulent.

For this reason, contrarily to the work done in the paper, we decided to compare the performance of the algorithms by using the AUC score instead. Therefore, although we obtained the same results as them, the logic behind picking Isolation Forest as the best algorithm in this case was not to say that its accuracy score was better (0.96>0.95) but rather than its AUC score was higher (0.91>0.66).

Conclusions and discussions

This project allowed to show how important reproducibility is to ensure that both the results of a paper are correct and that the process that led to derive them is valid. It allows to reduce the risk of errors, thereby by increasing the reliability of an experiment.

The paper chosen did not follow all the reproducibility criteria but, through hyperparameter tuning, we were still able to reproduce their results. Finally, we were able to derive their same conclusion that Isolation Forest is a better algorithm than Local Outlier Factor for the detection of anomalies. Even further, we provided better evidence to deduce this statement. For future work, it would be interesting to train a model using Neural Networks- in conjunction with sampling techniques to address the class imbalance- for the detection of fraud and compare its performance to the one of the Isolation Forest algorithm.

Sources

[1] V. Vijayakumar, Nallam Sri Divya, P. Sarojini, K. Sonika. Isolation Forest and Local Outlier Factor for Credit Card Fraud Detection System. International Journal of Engineering and Advanced Technology (IJEAT). ISSN: 2249 – 8958, Volume-9 Issue-4, April 2020

[2] Cornell University. A Survey of Predictive Modelling under Imbalanced Distributions (2015). arXiv:1505.01658 [cs.LG]

[3] Dataset: Kaggle's Credit Card Fraud Detection Dataset

Collaboration

The three of us worked together over Zoom to generate the code used and write the report.