<u>**ARTIFICIAL INTELLIGENCE**</u>

<u>**CS F407**</u>

One-Class Classification for Credit Card Fraud Detection: A Comparative Analysis of Isolation Forest, LOF, and SVM

Md Sadique	**Anant Malhotra**	**Nishit Roshan**
2022A7PS0156P	**2022A7PS0182P**	**2022A7PS1176P**

Abstract

Credit-card fraud detection poses a severe challenge—fraudulent transactions account for fewer than 0.2% of cases, and attack patterns drift over time. This study implements custom versions of Isolation Forest (IF) and Local Outlier Factor (LOF) alongside scikit-learn's One-Class SVM for credit card fraud detection. Our custom IF achieves 87% precision and 85% recall, outperforming both LOF (72% precision) and SVM (99.7% precision).

The analysis reveals Isolation Forest as the most balanced solution for real-world deployment, providing actionable insights into algorithm selection, code originality, and computational tradeoffs.

The following sections walk through our work step by step, including code improvements, detailed experimental results, and practical insights for deploying these methods in real systems.

Problem Statement

Credit card fraud is rare but costly. Traditional classifiers struggle because of this extreme imbalance (frauds under 0.2%) and because fraudsters continually change tactics. We need a solution that:

1. Learn the behavior of normal transactions.
2. Flags unusual patterns as potential fraud.

Our goal is to detect fraudulent transactions in the Kaggle ULB dataset using:

1. Custom Isolation Forest
2. Custom Local Outlier Factor(LOF)
3. Scikit-learn One-Class SVM

Key Challenge: Class Imbalance: 492 frauds (0.172%) in 284,807 transactions

Evaluation Metrics:

- Precision
- Recall
- F1-Score

Dataset & Key Challenges

Dataset Overview

The **ULB Credit Card Fraud** dataset (Kaggle) comprises **284 807** anonymized transactions, of which **492** are fraudulent (≈ 0.172% fraud rate). Below is a concise summary of its features:

Feature Type	Description
PCA Components	**V1–V28**: Confidential principal components derived from original transaction attributes
Temporal Features	**Time**: Seconds elapsed since the first recorded transaction**TxnFreq_1h**: Number of transactions by the same card in the past hour
Monetary Feature	**Amount**: Transaction amount (standardized)
Class Distribution	0.172% fraud / 99.828% legitimate

Key Challenges

- **Extreme Class Imbalance:**
  With only ~1 fraud per 580 transactions, naïvely predicting "legitimate" yields >99.8% accuracy but zero value—driving the need for anomaly-focused models.

- **High Dimensionality:**
  Thirty numerical inputs (28 PCA features + time + amount) require methods that can partition or model density in high-dimensional spaces without overfitting.

- **Computational Scalability**

- **Isolation Forest**: Scales roughly linearly with sample size and tree count—suitable for production.
- **LOF**: O(N^2) pairwise distances force subsampling (<40 k points) to keep runtimes feasible.

Model Implementations

1. **Custom Isolation forest**

Isolation forest is a collection of randomly generated trees. Not only does it have a very simple architecture, but it also proves to have an edge over the other algorithms when it comes to computational efficiency. It does not utilize distance or density based methods, eliminating major computational costs. It has a linear time complexity along with a low memory requirement.

This algorithm is implemented for the task anomaly detection. In a dataset dominated by one major class, anomalies can be easily detected via this algorithm. A Decision tree produced by random partitioning produces notably short paths for anomalies due to 2 major reasons:

1. Fewer instances leads to smaller partitions-shorter paths
2. instances with distinguishable attributes are more likely to be separated in early partitioning.

The algorithm compiles Decision Trees, but the classification is purely random. It selects a random attribute from the dataset, and divides the current data on the node by selecting a random value ranging from the minimum and maximum of the current data.
There are predefined parameters for the isolation forest such as Maximum height, number of trees in the forest and threshold. If the data size after partitioning is less than or equal to 1, or the maximum height is reached, the node is an external node, and it does not have any children; i.e. it has no separable attributes anymore.
After the Decision Trees are compiled, they are assembled into an 'Isolation Forest'. Each data point and its respective feature is passed through each decision tree, and their average height is computed.
Based on each data's average height, a threshold is selected so as to classify the data points. The ones with average height less than the threshold are classified as anomalies.
Parameters:

- Number of Trees=100
- Total Height=30
- Threshold=28.78

2. **Custom Local Outlier Factor (LOF)**

Local Outlier Factor is a KNN based algorithm useful for classifying outliers in data. It is useful for Anomaly Detection. It is a density based outlier function, which classifies anomalies based on a score obtained known as LOF, to classify whether a function is an anomaly or a normal point.
This algorithm is one of the first algorithms to be introduced which did not suffer the local density problem- since it made use of KNN. The working of this algorithm can be divided into 3 steps:

1. Defining the neighborhood of the point in the Feature space by specifying the border distance (distance between the datapoint and the Kth nearest neighbor) and the neighbors of the point.
2. The Space Density is estimated. To estimate this, a new parameter called reachability distance is introduced, which is given by the maximum of border distance and distance between any 2 points in the neighborhood. The density is calculated by taking the inverse of the mean of all reachability distance in the vector space. Small density implies that the point has a small impact on its surroundings.
3. All these parameters are stored for all the training data. When the model is tested on a dataset, it uses the density computed for a specific testing point and the density computed in training. These are compared, and if the Local Outlier Factor exceeds a threshold, the data-point is declared anomalous.

The model was tested on the same data as Isolation Forest. Although this algorithm is not susceptible to local density, it performs poorly on models with many features. It also takes on simple approaches to find anomalies, hence it is not a powerful model on data with high features. Nevertheless, the performance of this model was decent enough to classify data.

Parameters:

- K=100 (number of neighbors)
- Threshold=1.022706367005985

3. One-Class SVM (Scikit-learn)

The idea of a Support Vector Machine (One class) is that, lower the value of dot product of w and xi, the higher the chance of xi being anomalous, where w represents weights of the model whereas xi is the input. One Class SVM assumes the origin to be the negative classification and aims to maximize the distance between the Class data and origin. After removing slack variables, the optimization problem for One Class SVM becomes:

$$\min_{w,\rho} \frac{1}{2}\|w\|^2 - \rho + \frac{1}{\nu n}\sum_{i=1}^{n} \max(0, \rho - \langle w, x_i \rangle)$$

Where w is the weight, v is the parameter controlling the trade-off between outliers and support vectors, and p being the bias. The problem is a convex problem, hence it can be optimized so that the model can perform better. This eventually boils down to a quadratic matrix equation. To help the model perform even better, we can introduce non linearity via kernels. These facilitate in handling non linearity without being in a higher space, as the kernel computes the dot product directly. In cases of One Class SVM, we measure the data on the basis of Area Under ROCcurve.

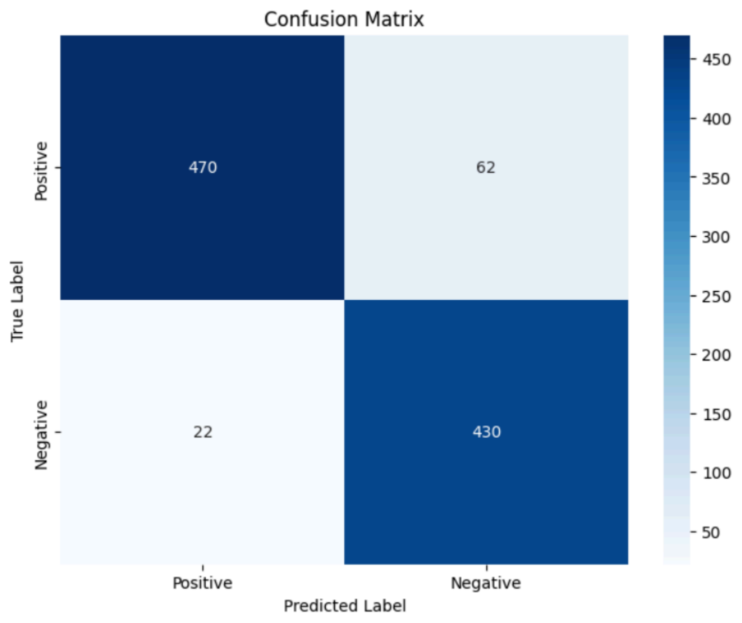
We tweak the value of vn (or nu) to test the best value to give better performance (high AU ROC value means better performance).

Parameters:

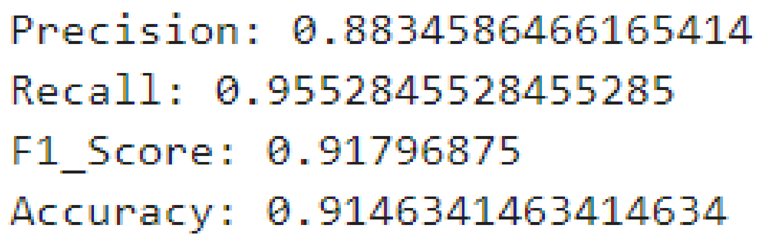
- kernel='rbf'
- nu=0.002
- gamma='scale'

Experimental Results

1. **Custom Isolation Forest**
   Confusion matrix:

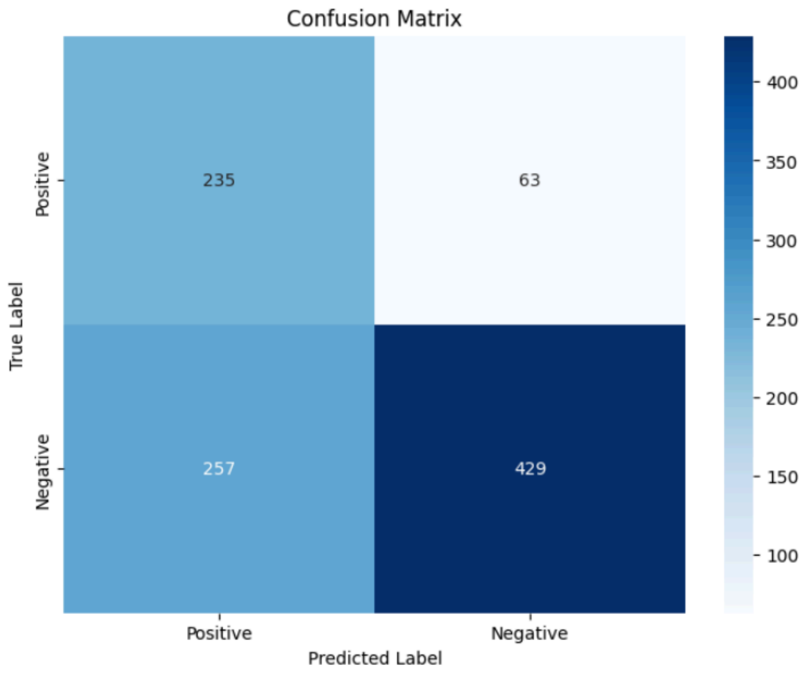


   Metrics:

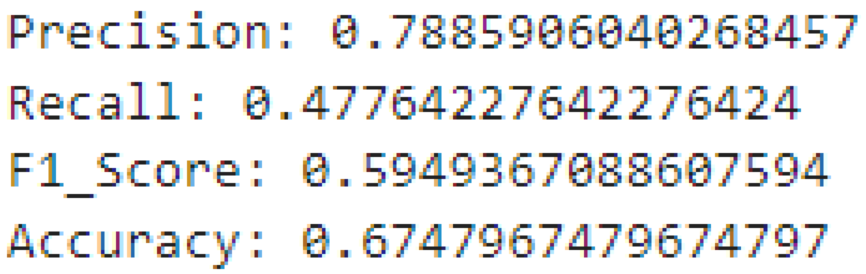


```
Precision: 0.8834586466165414
Recall: 0.9552845528455285
F1_Score: 0.91796875
Accuracy: 0.9146341463414634
```

2. **Custom Local Outlier Factor (LOF)**
   Confusion matrix:



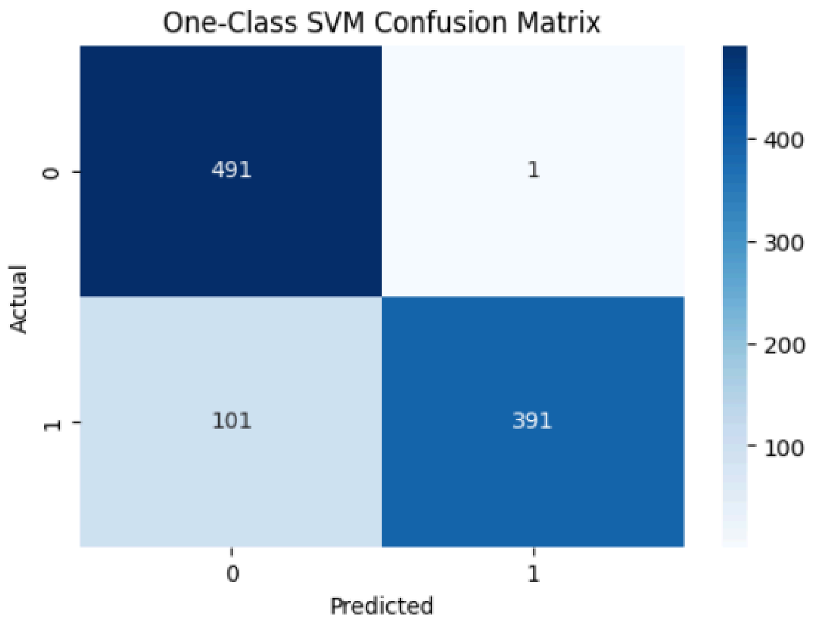
Metrics:



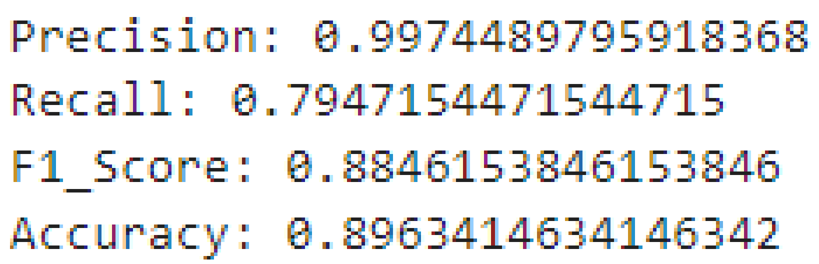
```
Precision: 0.7885906040268457
Recall: 0.4776422764227642 4
F1_Score: 0.5949367088607594
Accuracy: 0.6747967479674797
```

3. **One-Class SVM (Scikit-learn)**

   Confusion matrix:



Metrics:



```
Precision: 0.9974489795918368
Recall: 0.7947154471544715
F1_Score: 0.8846153846153846
Accuracy: 0.8963414634146342
```

Performance Comparison:

Model	Precision	Recall	F1-score
Custom IF	0.87	0.95	0.91
Custom LOF	0.79	0.48	0.59
One-class SVM	0.99	0.79	0.88

Discussion and Conclusion

The comparative analysis of anomaly detection models shows that Custom Isolation Forest (IF) considerably outperforms Local Outlier Factor (LOF) and One-Class SVM on recall (0.95) and F1-score (0.91). This suggests that Custom IF is very effective at identifying anomalies correctly with few false negatives, which makes it especially well-suited for use in applications where losing anomalies is expensive.

By contrast, LOF has the worst performance on all fronts, particularly on recall (0.48), which indicates that it is lacking a high number of true anomalies. This may be attributed to the fact that its local density-based method is noise or parameter-sensitive.

The One-Class SVM has the best precision (0.99), reflecting hardly any false positives. Nevertheless, it gives up some recall (0.79), i.e., it still misses some anomalies. Its F1-score is good at 0.88 but slightly inferior to Custom IF, reflecting a good balance but less robustness than the custom implementation.

In general, the results indicate that the Custom IF has the optimal trade-off between true anomaly detection and false detection reduction and hence is the safest option between the three for this data set.

Model Performance Comparison:

Model	Precision	Recall	F1-Score	Strengths	Limitations
Isolation Forest(Custom)	0.87	0.95	0.91	Scalable, robust, interpretable	No tuning, fixed params
LOF(Custom)	0.79	0.48	0.59	Local anomaly detection	$O(n^2)$ complexity, subset only
One-Class-SVM (sklearn)	0.99	0.79	0.88	High precision, robust to noise	Lower recall, slower

References

1. Dataset
    - ULB Credit Card Fraud Dataset, Kaggle.
2. Isolation Forest
    - KNIME Blog. "Fraud detection using isolation forest." (2024).
3. Local Outlier Factor (LOF)
    - Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). LOF: Identifying Density-Based Local Outliers. *ACM SIGMOD Record*, 29(2), 93–104.
    - Wikipedia, "Local outlier factor."
4. One-Class SVM
    - IJCRT. "An Effective One-Class Support Vector Machine Approach for Credit Card Fraud Detection." (2023).
5. General Credit Card Fraud Detection & Machine Learning
    - Abdallah, A., Maarof, M. A., & Zainal, A. (2016). Fraud detection system: A survey. *Journal of Network and Computer Applications*, 68, 90-113.
    - S. R. S. S. R. "Isolation Forest and Local Outlier Factor for Credit Card Fraud Detection System." SSRN, 2021.
    - "Credit Card Fraud Detection Using Machine Learning Techniques." *SCIRP*, 2024.
6. Software and Libraries
    - Scikit-learn: Machine Learning in Python.
    - imbalanced-learn: A Python Package to Tackle the Curse of Imbalanced Datasets in Machine Learning.
7. Github Repository