Benchmarking anomaly detection techniques in the fraud detection domain*

*Note: Sub-titles are not captured in Xplore and should not be used

1st Leonardo Brito
Engineering and Data Science
Instituto Superior Técnico
Lisbon, Portugal
leonardo.amado.brito@tecnico.ulisboa.pt

Company Supervisor: Jacopo Bono
Data Scientist
Feedzai
Porto, Portugal
jacopo.bono@feedzai.com

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

In the realm of digital transactions, the significance of detecting fraud at its earliest is paramount. Supervised learning models are often the preferred choice, renowned for their accuracy, but they hinge on the availability of labeled data. This dependency can introduce operational delays, leaving a window where transactions might be exposed to unchecked fraudulent activities.

To address this gap, there's a growing interest in leveraging unsupervised learning. Without the need for labeled data, unsupervised anomaly detection techniques offer a promising avenue, especially when one considers that fraud typically surfaces as statistical anomalies amidst legitimate transactions.

Drawing from open-source datasets, this report sets out to critically evaluate a range of unsupervised anomaly detection methods in the context of fraud detection. Our approach is methodical: starting with a detailed data exploration, we then benchmark against a supervised baseline, and finally, analyze the performance of both classical and deep learning unsupervised models. The overarching aim is to ascertain the potential of unsupervised techniques in offering interim protection against fraud during periods when supervised models are not yet feasible.

II. Exploratory Data Analysis

A. European cardholders's transactions

This dataset represents credit card transactions made by European cardholders in September 2013 over a span of two days.

Out of the 284,807 transactions recorded, 492 were fraudulent, making up a mere 0.172

Identify applicable funding agency here. If none, delete this.

The dataset consists mainly of numerical variables derived from a Principal Component Analysis (PCA) transformation. To maintain confidentiality, the original features and further details about the data are not disclosed. The variables V1, V2, through V28 are the principal components derived from the PCA. The only exceptions that haven't undergone PCA are 'Time' and 'Amount'. The 'Time' feature indicates the seconds that have passed between each transaction and the first one in the dataset. On the other hand, 'Amount' denotes the transaction value, which can be useful for context-sensitive learning approaches. The 'Class' feature is the target variable, where a value of 1 indicates a fraudulent transaction, and 0 signifies a legitimate one.

There is no null or missing values of the dataset.

TABLE I
Summary Statistics for Fraud and Valid Transactions

Statistic	Fraud	Valid
Count	492	284315
Mean	122.21	88.29
Standard Deviation	256.68	250.11
Minimum	0.0	0.0
25% Quantile	1.0	5.65
50% Quantile (Median)	9.25	22.0
75% Quantile	105.89	77.05
Maximum	2125.87	25691.16

B. Analysis of Transaction Amounts for Fraudulent and Valid Transactions

The table presents a summary of transaction amounts for both fraudulent and valid categories. A notable observation is the disparity in the count of transactions: while there are only 492 fraudulent transactions, there are a staggering 284,315 valid ones. This vast difference highlights the class imbalance inherent in the dataset.

When examining the transaction amounts, fraudulent transactions have a mean value of approximately $122.21, which is slightly higher than the valid transactions' mean of $88.29. Despite this, the maximum fraudulent transaction ($2125.87)

is significantly lower than the highest valid transaction, which reaches a substantial $25,691.16.

The spread of transaction amounts, as represented by the standard deviation, is roughly similar for both classes, hovering around the $250 mark. The median value for valid transactions ($22.00) is more than double that of fraudulent transactions ($9.25), indicating that half of the fraudulent transactions are below this amount.

While fraudulent transactions are comparatively rarer, their amounts can vary significantly, sometimes even surpassing the typical amounts seen in valid transactions. It underscores the importance of leveraging advanced techniques to detect such anomalies amidst the vast sea of valid transactions.

1) Imbalanced Dataset: An imbalanced (or unbalanced) dataset refers to a situation where the number of observations belonging to one class is significantly lower than those belonging to the other classes. In the context of a binary classification, which consists of two classes, an imbalanced dataset typically has a disproportionate ratio of observations in one class compared to the other. This problem can lead to:

- **Performance Deception:** Traditional metrics like accuracy can be misleading. A naive classifier predicting only the majority class would still achieve a very high accuracy due to the imbalance.
- **Model Bias:** Many machine learning models might exhibit a bias towards the majority class, often ignoring the minority class.
- **Decreased Predictive Power:** The minority class, in this case, the class of higher interest might not be predicted well, leading to a higher number of false negatives.

Given that our dataset has a class with 99.83% representation, it's highly imbalanced. I will present later on this report some considerations and how to deal with this problem.

subsubsectionCorrelation Matrix The fact the dataset is unbalanced can lead to misleading results when using correlation metrics:

- In highly imbalanced datasets, even small patterns in the minority class can result in seemingly strong correlations. This can lead to the model finding relationships that aren't generalizable.
- Correlation measures in an imbalanced dataset might be dominated by the majority class, potentially overshadowing meaningful relationships present in the minority class.
- Also, because the minority class has fewer data points, there's a higher risk of not detecting a relationship (false negative) when one might exist.

Due to the problem i presented, Traditional correlation metrics may not be suitable for imbalanced datasets. In this binary classification problem with a highly imbalanced target, a high Pearson correlation coefficient with a predictor might be misleading.

C. Correlation Matrix

The Pearson correlation coefficient, often denoted as r, measures the linear relationship between two continuous variables. The value of r lies between -1 and 1, where:

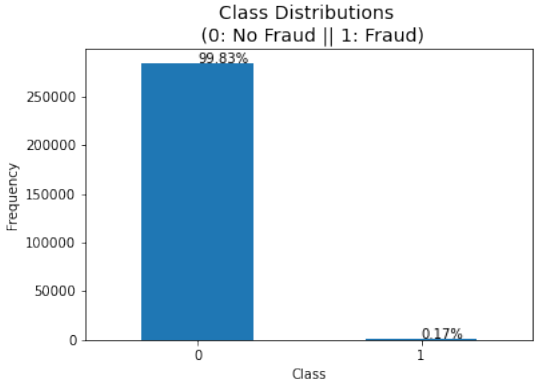

Fig. 1. Example of a figure caption.

- $r = 1$: Indicates a perfect positive linear relationship.
- $r = -1$: Indicates a perfect negative linear relationship.
- $r = 0$: Suggests no linear relationship.

The mathematical formula for the Pearson correlation coefficient between two variables X and Y is:

$$r = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2 \sum(Y_i - \bar{Y})^2}}$$

Where:

- X_i and Y_i are the individual data points.
- $\bar{X}$ and $\bar{Y}$ are the means of X and Y respectively.

The numerator of the formula calculates the covariance between X and Y. It sums the product of the differences of each data point from their respective means. The denominator, on the other hand, normalizes this covariance by the product of the standard deviations of X and Y, ensuring the correlation coefficient remains between -1 and 1.

D. Variance

article amsmath

Given that your data is the result of a PCA transformation, several unique characteristics and considerations come into play:

1) **Orthogonality:** The principal components resulting from PCA are orthogonal (uncorrelated). Thus, the correlation matrix of these components should be a diagonal matrix with ones on the diagonal (or very close to this in practice due to numerical precision).
2) **Variance Explained:** One of the key aspects of PCA is the amount of variance explained by each principal component. The first few components typically capture the majority of the variance in the dataset, while the latter components capture less and less variance.

Given these considerations, the variation analysis that makes the most sense for PCA-transformed data includes:

1) **Variance Explained:**
   - **Scree Plot:** A plot showing the fraction of total variance explained by each principal component.

This helps in determining how many components to retain for further analysis.

PCA is a dimensionality reduction technique that seeks to identify axes in data that maximize variance. The method involves computing the eigenvalues and eigenvectors of the dataset's covariance matrix. The eigenvectors, termed principal components, determine the direction of the new feature space, while the eigenvalues define their magnitude, i.e., the variance in those directions.

1) **Eigenvalues and Variance**: The eigenvalue associated with each principal component signifies the variance along that component. In PCA, these components are ordered by descending eigenvalues. This means the first principal component encapsulates the largest variance in the dataset.
2) **Orthogonality**: Every principal component is orthogonal to every other, implying they are uncorrelated. Hence, each subsequent component captures the direction of maximum variance not represented by the preceding components.
3) **Maximizing Variance**: The first principal component (often denoted as $PC1$) represents the direction in the original feature space capturing the utmost variance. The second principal component (often $PC2$) is orthogonal to $PC1$ and represents the second-highest variance, and so forth.

Consequently, when PCA-transformed features are acquired (like 'V1', 'V2', 'V3', etc.), they are inherently ordered by the variance they represent from the original dataset, with 'V1' representing the most and the final component the least.