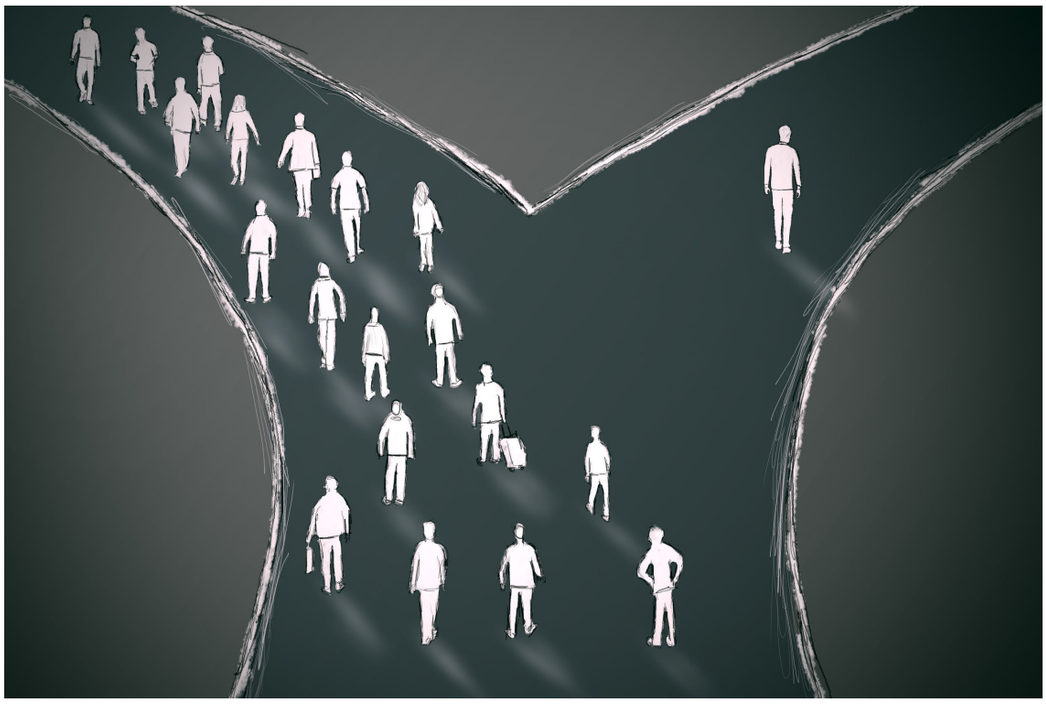
Stanley Ngunyi Gathagu

Unsupervised Learning: Outlier Detection

Using Unsupervised Machine Learning To Detect Anomalies



1 Introduction

In this analysis we aim to detect outliers by using contextual and behavioural attributes of the sample. Outlier detection techniques are often referred to as *Anomaly Detection* techniques where the goal is to identify samples that exhibit an inconsistent behaviour given their contextual attributes (features). Stated differently, outliers are extreme and unexpected data points that deviate from other observations in the data, they may indicate experimental errors, variability in a measurement or a novelty (*point outliers, contextual outliers, or collective outliers* respectively).

There are different approaches popular in the research and business communities for tackling the problem of outlier detection. Namely, *Rule* based techniques, *Machine Learning* based and *Deep Learning* based techniques. However, Rule based systems have one major disadvantage over ML and DL based anomaly detection algorithms. Namely, the rules and signals used in the approach need to be adjusted on frequent bases which make this method not sustainable. This is especially problematic for one specific application of anomaly detection, which is the fraud identification using anomaly detection. Unlike Rule based methods, ML and DL anomaly detection algorithms don't require manual and frequent adjustments for accurate performance.

1.1 Challenge Goal

In this challenge, we will use Multivariate and Unsupervised Machine Learning Algorithm, *Isolation Forest*, for detecting outliers in our unlabelled data. Moreover, we will combine Isolation Forest with PCA technique for reducing the dimension of our data while keeping as much as possible information and for visualizing outliers in 2D and in 3D.

It's known that most outlier detection systems are effective in 99% cases. For the remaining cases the model either inaccurately labels the observation as outlier while it is not an outlier (False Positive) and in other cases the model inaccurately labels the observation as not an outlier while it is actually an outlier (False Negative). Which of these two type of mistakes is more dangerous depends on the use-case and what are the consequences of making these mistakes. For example in Fraud Detection high FP might significantly affect the customer satisfaction hence one might want to use more complex techniques such as Deep Learning techniques (RNN with LSTMs) to avoid making many FP.

1.2 Paper Structure

The remaining part of this short explanatory paper is structured as follows: Section 2 will discuss the data, Section 3 will present the motivation behind the chosen methodology, Section 4 will present the results and the conclusions and Section 5 will present the Evaluation of Unsupervised Outlier Detection and how Cross-Scoring approach is used to measure the performance of the chosen outlier detection model.

2 Data

In this challenge we will use unlabeled data consisting of 78764 samples among which some of them are outliers. In this data, each sample is described by a total of 41 features: 37 contextual attributes and 4 behavioral attributes. By definition, outlier is a sample that exhibits an inconsistent *behavior* given it's contextual attributes.

Before moving forward with the data visualization and transformation, the following steps are performed: *1: Checking whether there are missing data points present in the data* (there are no missing values present in the data), *2: determining the datatype of each variable in the data set* (following variables are categorical-string variables {b2, c3, c9, c15, c18, c19, c20, c26, c29} and the remaining variables are numeric variable), *3: identifing the range of data values.*

2.1 Data Preprocessing

There are two main data transformations we conduct in this use-case:

- Normalization of the numerical variables
- Converting *String* categorical variables to *Integer* categorical variables

Since, the majority of the numerical variables have varying and large numerical ranges, which can influence the model especially when using techniques such as PCA which we intend to use for dimensionality reduction in this use-case, normalization is essential. More specifically, the PCA calculates a new projection of the data and the new axis are based on the standard deviation of the variables. Hence, a variable with a high standard deviation will have a higher weight for the calculation of axis than a variable with a low standard deviation. If we normalize the data, all variables have the same standard deviation, thus all variables have the same weight and the PCA calculates the relevant axis. Therefore, it is necessary to normalize data before performing PCA. To normalize the numerical variables in our data we use the *Pythons* library called *preprocessing.*

Furthermore, we convert all categorical variables in the data from Sting values to Integer values such that those variables can also be used in PCA and in the Outlier Detection algorithm.

2.2 Data Visualization

In order to select set of possible Machine Learning Algorithms for detecting outliers in the data, it is crucial to look at the distribution of the features since some of the algorithms rely heavily on the assumption that the features follow Normal distribution. For the visualization purposes we only look at the behavioural variables {b1, b2, b3, b4}. Note that for visualization purposes, the variable "b2" (categorical variable) has already been transformed to categorical variable with integer values. Figure 1 corresponds to the pair-wise scatter plot of 4 behavioural attributes in the data. In the plot, we have manually circled the points which might be considered as outliers.
Figure 2 plots the distributions of the earlier mentioned 4 behavioural variables after transformation (normalization) but we can see that even after normalization the distributions of all these variables

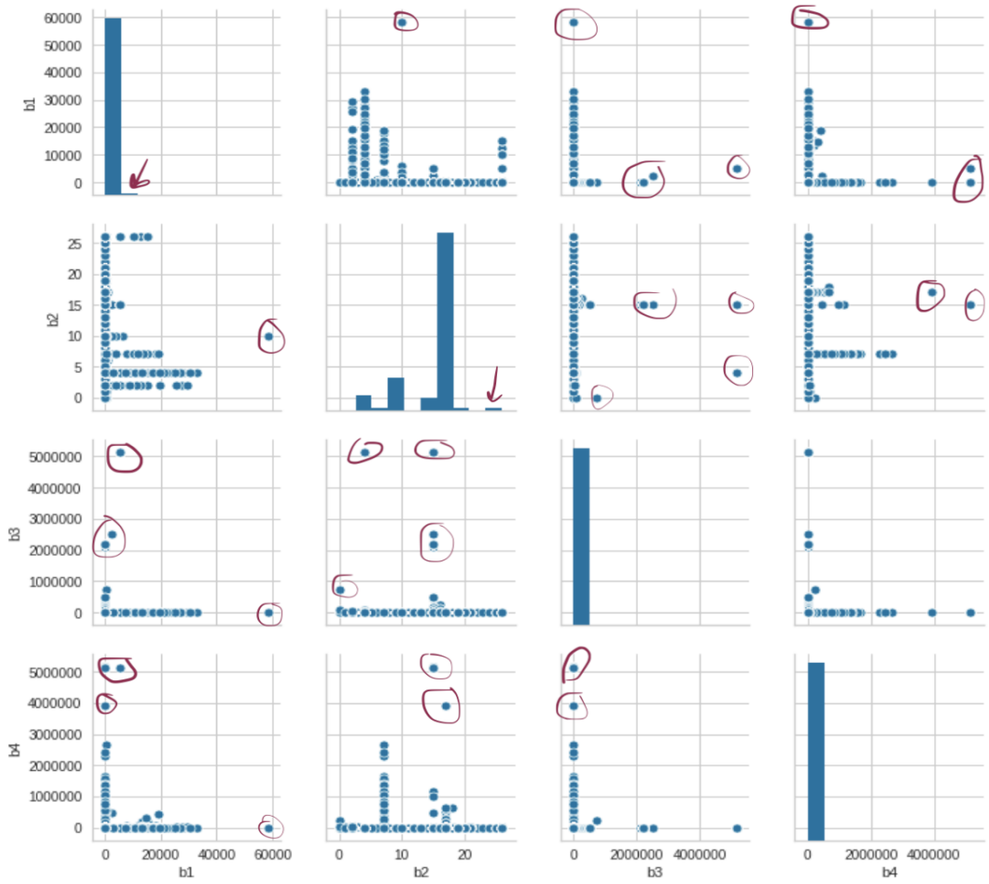


Figure 1: Pair-Wise Scatter Plot: Behavioural Attributes

are not symmetric and centered around their mean suggesting that even after normalization the behavioural features don't follow Normal distribution. In the top left corner of this figure, we can see the *Kurtosis* corresponding to each of these four variables, which tells us about the degree of *peakedness* of a distribution. We see that in all 4 cases the peak of the distribution doesn't lie in the centre and lies either in the far left part of the distribution or in the right part which once again verifies that these variables don't follow Normal distribution. In fact, "b1" and "b3" are right skewed whereas "b2" and "b4" are left skewed.

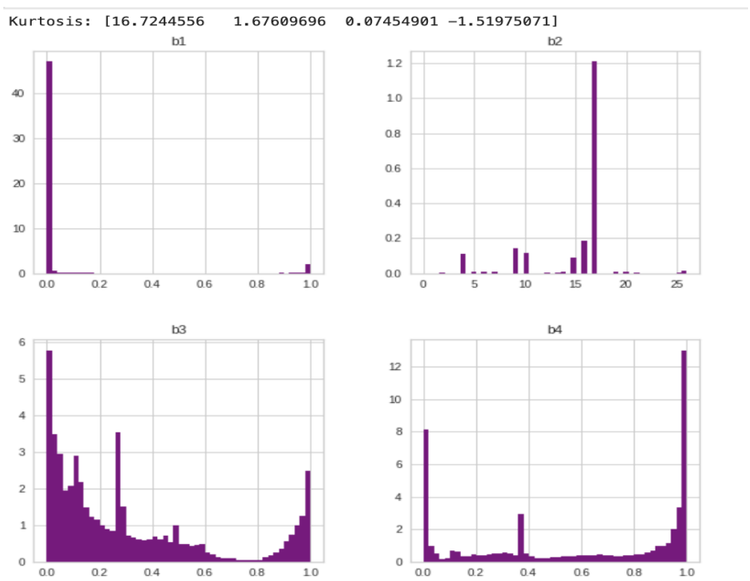


Figure 2: Probability Distribution Functions of the Transformed Behavioural Variables

3 Methodology

In this section we will discuss various aspects considered when choosing an outlier detection technique. Namely, following factors have been consider when choosing an anomaly detection technique for this use-case:

- Univariate vs Multivariate Outlier Detection
- Unsupervised vs Supervised Learning
- Data Distribution (Normal vs not Normal)
- Global and/or Local Outlier Detection
- Computational Speed and Space Requirement

3.1 Selecting Outlier Detection Techniques

Univariate vs Multivariate Outlier Detection

Multivariate Outliers occur when the values of various features, taken together seem anomalous even though the individual features do not take unusual values. Multivariate Anomalies occur when the values of various features, taken together seem anomalous even though the individual features do not take unusual values.

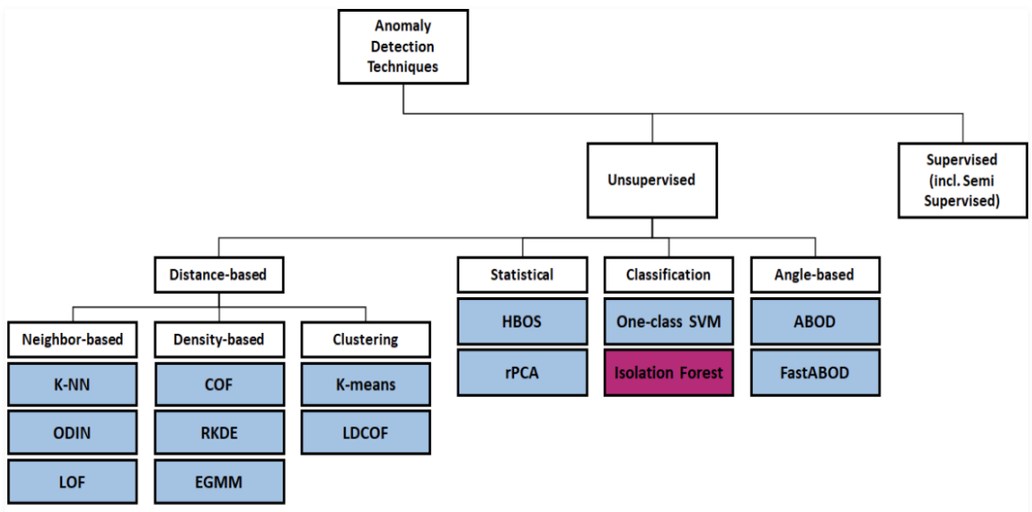
Unsupervised vs Supervised Outlier Detection

The main difference between the two types of Machine Learning Methods is that Supervised Learning Algorithms require labelled data or a *ground truth* which basically means having a prior knowledge of what the output values for the samples should be. Therefore, the goal of the Supervised Learning is to learn a function that, given a sample of data and desired outputs, best approximates the relationship between input and output observable in the data. On the other hand, Unsupervised Learning algorithms don't require labeled data and their goal is to infer the natural structure present within a set of data points and learn from the features in the data to produce an output. *Since, our data doesn't contain labels for the samples indicating whether they are actually outliers or not, we will be dealing with* ***Unsupervised Learning****.*

Unsupervised Outlier Detection Techniques

Unsupervised anomaly detection algorithms can roughly be categorized into the following categories as illustrated in the figure below: Nearest-neighbor based techniques, Density based techniques,(2) Clustering based methods, Statistical, Classification and Angle-based algorithms. Since, all these different unsupervised approaches work differently and serve different purposes, the choice of the technique is highly dependent on the use-case. For example, the **LOF**'s or **Local Outlier Factor** main purpose is to identify *local anomalies* where LOF is an unsupervised ML algorithm that uses the density of data points in the distribution as a key factor to detect outliers. Another such

example approach is **K-NN** or **K-Nearest Neighbour** approach for outlier detection. The main goal behind K-NN approach, unlike the LOF, is to identify *global anomalies*. Some very popular techniques such as **Z-score** anomaly detection, heavily depend on the underlying distribution of data which is not applicable to our case since the our data is not normally distributed, as we saw in the previous section.



Some examples of outlier detection techniques that identify both global and local anomalies and don't assume that the features in the data follow Normal distribution are: Isolation Forest, One-class SVM, K-Means, rPCA, DBScan and others. *Since, we aim to investigate the behaviour of samples while considering the behavioural variables in isolation and in combination, we will be dealing with* ***multivariate*** *outlier detection.*

Inclusion-exclusion Principle

One of the most important assumptions for an unsupervised anomaly detection algorithm is that the dataset used for the learning purpose is assumed to have all non-anomalous training examples or very small fraction of anomalous examples. So, the goal of the anomaly detection algorithm is to learn the patterns of a normal activity in the data so that when an anomalous activity occurs, we can flag it through the inclusion-exclusion principle. *In this use-case, we will assume that the data contains* ***1****%* ***outliers***

One of the main limitation to standard, distance-based methods is their inefficiency in dealing with high dimensional datasets: The main reason for that is, in a high dimensional space every point is equally sparse, so using a distance-based measure of separation is highly ineffective. Since, our data is high dimensional we will be better of to not rely on Distance-based Outlier detection algorithms. Moreover, since, our goal is to identify both global and local anomalies in a multivariate setting, our candidate unsupervised learning algorithms for this use-case are: **Isolation Forest**, **One-class SVM**, **K-Means**, **rPCA**, **DBScan**.

3.2 Isolation Forest

Because of time limitation, we will only pick one algorithm out of the previously selected algorithm which is the Isolation Forest Unsupervised outlier detection technique. Isolation Forest is based on the Decision Tree algorithm and it isolates the outliers by randomly selecting a feature from the given set of features and then randomly selects a split value between the max and min values of that feature. This random partitioning of feature space produces shorter paths in the trees for the anomalous points, thus distinguishes them from the rest of the data. This processes is illustrated in the Figures 3 and 4.

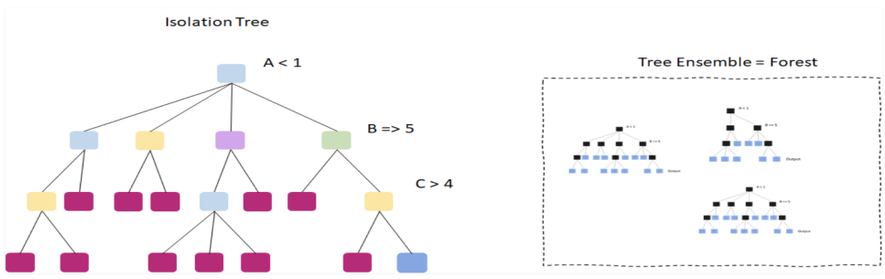


Figure 3: Isolation Forest: Tree Building Process

Unlike the most anomaly detection techniques, where the first step is to construct a profile of what's "normal", and then report anything that can't be considered as "normal" as anomalous, in the Isolation Forest algorithm does not first define "normal" behavior, and it does not calculate point-based distances. Instead, Isolation Forest isolates anomalies in the data points instead of profiling "normal" data points.

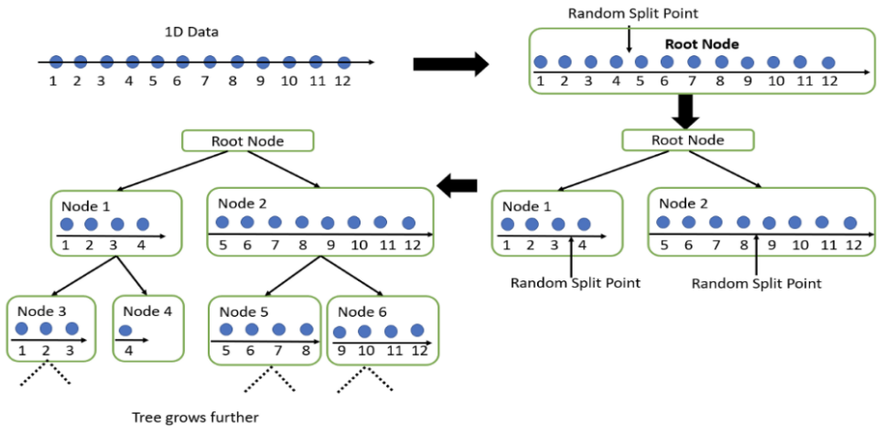


Figure 4: Isolation Forest: Tree Building Process

The steps to compute an Isolation Tree can be described as follows :

- 1: Select a feature at random from data (f)
- 2: Select a random value from the feature f and use this random value as a threshold (t)
- 3: Store data points where f $<$ t are stored in Node 1 and where f $\geq$ t go in Node 2
- 4: Repeat Steps 1–3 for Node 1 and Node 2
- 5: Terminate when the tree is fully grown or when termination criterion is met

As anomalies data points mostly have a lot shorter tree paths than the normal data points, trees in the isolation forest does not need to have a large depth that results also in *low memory requirement* which is another comparative advantage of this algorithm. Moreover, this algorithm works very well with a small data set as well as with large datasets. Finally, Isolation Forests are known to work well for high dimensional data.

3.3 PCA: Dimensionality Reduction

It was mentioned in the Data section that there are 37 contextual variables that describe the samples which need to be used to train the model. Unfortunately, high-dimensional data also affects the detection performance of Isolation Forest, but the performance can be vastly improved by adding a features selection approach to the process to reduce the dimensionality of the sample space. Hence, we will use PCA (Principal Component Analysis) technique to perform feature selection for the contextual attributes in order to reduce the number of features in the model while keeping as much as possible variation in the data as possible. Since, we our goal is to use contextual features as a way to learn about the sample and then find out whether that samples behaviour is "normal" or not, we will only reduce the contextual variables dimensions and we will combine these Principle Components, which are linear combination of the many descriptive features with 4 behavioural attributes to train the model and identify the outliers.

The figure 5 illustrates the percentage variation in data explained by each Principal Component. The Elbow rule suggests that the number of optimal PCs is equal to the x-axis value in this graph where the "elbow" occurs which is equal to 3 but one could also use 4 as the optimal number of PCs. Moreover, one can compute that all these four PCs together explain around 98% of variance in the contextual data which is very high. This suggests that if we use these 4 PCs instead of the 37 contextual attributes we will still keep the 98% of the information in that data.

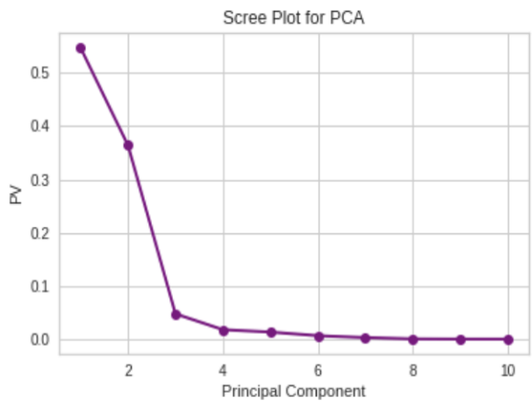


Figure 5: Scree Plot for PCA: Elbow Rule

4 Results

We use the 4 behavioral variables combined with 4 Principal Components which are linear combinations of 37 contextual variables to train the Isolation Forest ensemble model. The model has identified 788 outliers out of total 78764 samples.

For the visualisation purposes, we run also another set of PCA this time with all variables included and picked the first 2 and 3 PCs that explain the 98% and 99% variation in the data, respectively in order to visualize the outliers as illustrated in the figures 6, 7, 8, and 9.

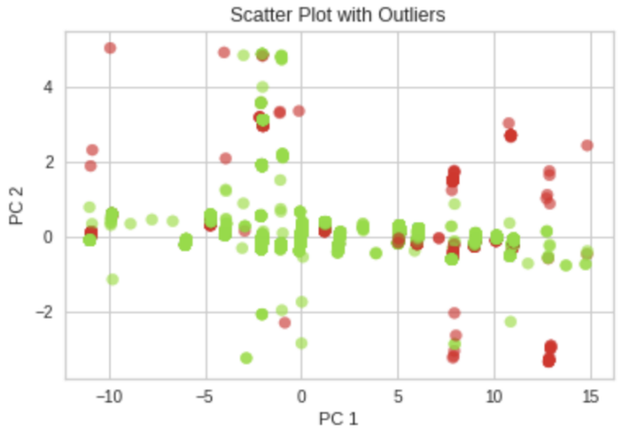


Figure 6: Isolation Forest Outlier Detection 2D: PC1 vs PC2

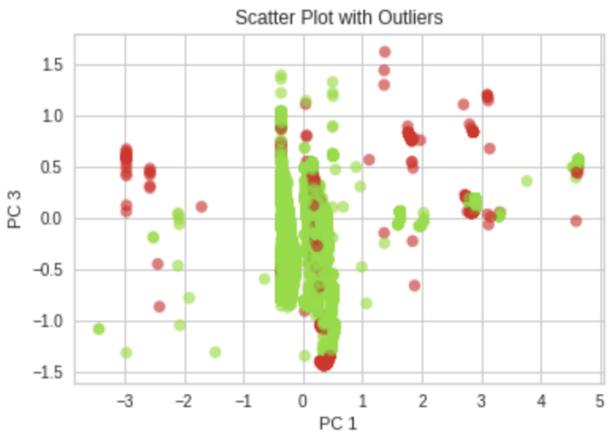


Figure 7: Isolation Forest Outlier Detection 2D: PC1 vs PC3

Figure 8: Isolation Forest Outlier Detection 2D: PC2 vs PC3

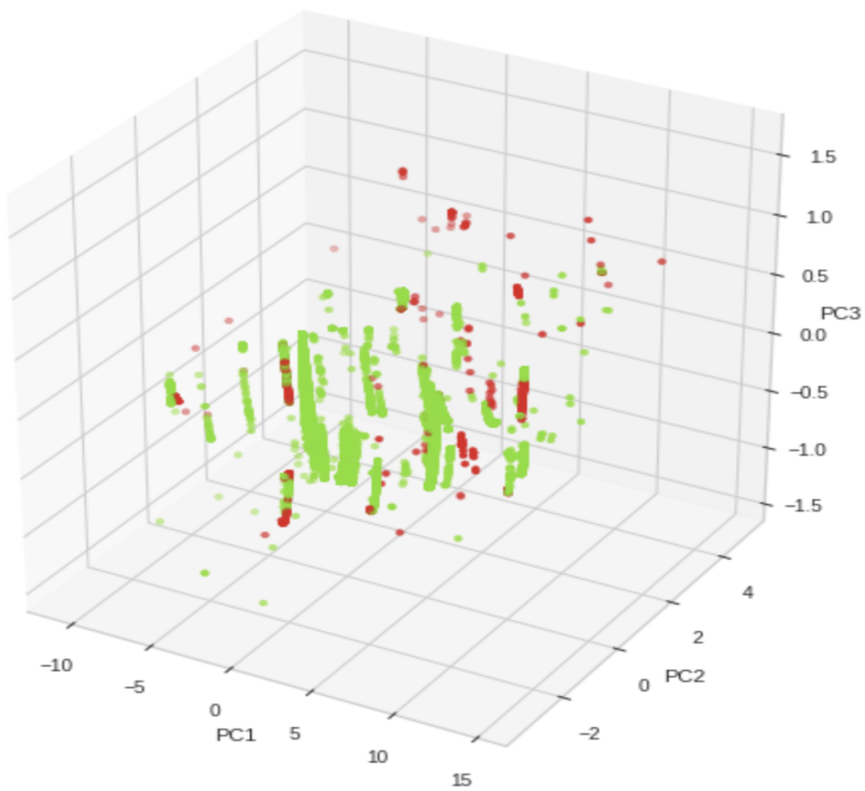
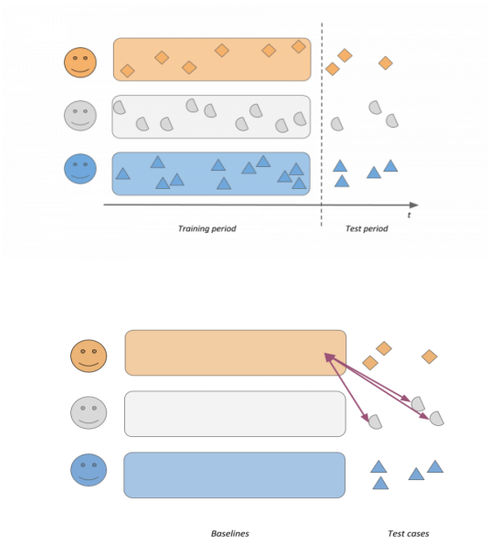


Figure 9: Isolation Forest Outlier Detection 3D: PC1 vs PC2 vs PC3

5 Evaluation of Outlier Detection Techniques

Since the number of occurrence of anomalies is relatively small compared to normal data points, we cann't use a usual model accuracy as an evaluation metric because for a model that predicts everything as non-anomalous, the accuracy will be greater than 99.9% and we would not have identified any outlier.

In case of supervised learning, the evaluation of an outlier detection model is much more convenient than in case of unsupervised learning model. In case of labelled data, metrics such as False Positive, False Negative, Precision, Recall and F1 score can be used to evaluate the model. However, in case of not labelled data, those metrics can no longer be used. Though evaluation of Unsupervised ML outlier detection techniques is not an easy task but is a possible one since there are different approaches for this. One popular approach for evaluating unsupervised ML outlier detection technique is the **Cross-scoring** introduced by Goix N. (2016). The concept behind this approach is that when measuring how well an algorithm learned the normal behavior of a user, we test its knowledge against activities made by other users as well as activities made by this target user. Stated differently, to see if we would recognize sample A as outlier, we check what would



happen if sample B replaced A but followed its own (sample B-typical) behaviour, but this time using A's place. If sample B's behaviour is identified as outlier compared to A's baseline then this implies that the outlier detection algorithm has the capability of finding potentially not normal points. Once the model is trained with this artificially created labelled data, we can then calculated common classifying model accuracy metrics such as Recall, Precision, F1 score which can be calculated as follows:

$$\textbf{Precision} = \frac{tp}{tp + fp}$$
$$\textbf{Recall} = \frac{tp}{tp + fn}$$

By using 20% randomly selected data where 1% is switched we get the following results:

From the results we can see that Model was able to correctly find 68% of all outliers and it was able to correctly label all non-anomalous points as "normal" points. From the calculations we can

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

```
True Positive Rate:68%
False Positive Rate:0%
True Negative Rate:100%
False Negative Rate:32%
Precision:0.6730769230769231
Recall:0.6774193548387096
F1 Score:0.67524115755627
```

also see the model's Precision is 67% and the Recall is 68%. These metrics can be improved by hyper parameter tuning for example.

6 References

Aggarwal, C.C. and Yu, P.S. *Outlier detection for high dimensional data.* ACM Sigmod Record, 2001.

Goix, N. *How to Evaluate the Quality of Unsupervised Anomaly Detection Algorithms?* ICML2016 Anomaly Detection Workshop, New York, NY, USA, 2016.

Hodge, V.J. and Austin, J. *A survey of outlier detection methodologies.* Artificial Intelligence Review, 2004.

Fei T. L. and Kai M. T. and Zhi-hua Z. *Isolation Forest.* In ICDM '08: Proceedings of the Eighth IEEE International Conference on Data Mining. IEEE Computer Society. 413-422, 2008.