Evaluation of the Performance of Isolation Forests for anomaly detection

Author

Barath Raj Asokan

MSc Data Analytics and Artificial Intelligence

61922

Abstract

Most of the existing anomaly detection algorithms construct a profile of normal instances and then try to identify those instances that do not abide to the normal profile as anomalies.

This thesis aims to evaluate the performance of a different model – Isolation Forests that explicitly isolates anomalies instead of profiling normal points. The Isolation Forest algorithm will be compared with a density-based anomaly detection algorithm – Local Outlier Factor (LOF) and a Distance based anomaly detection algorithm – Mahalanobis Distance and their performance will be evaluated using the precision of the detected anomalies in the data sets.

Two Data sets will be used to compare the results, one will be a synthetic data set which is self-generated (labelled data) and the other will be a natural data set (unlabeled) from Kaggle.

Acknowledgement

I would like to express my sincere gratitude to Prof. Dr. Christophe Croux for allowing me to undertake this thesis.

I would also like to thank EDHEC Business School for providing the necessary tools and resources required for the completion of the thesis.

List of Figures

Table of Contents

1. Introduction

In Machine Learning, the detection of anomalies/outliers has always been of keen interest in a myriad of fields like Healthcare, Finance, Government agencies etc. Anomaly detection is the identification of extreme cases which have extreme values different from the rest of the data set. These observations are called anomalies/outliers and needs to be identified to be separated from the normal observations. There could be many reasons for the anomalies:

1. Variability of the given data set.
2. Man made errors during the data collecting process.
3. Some new/rare occurrence outside of the norm.

These are few of the examples where anomalies exist.

Managing outliers is a tedious process and is challenging because it is not easy to determine if the issue is related to errors in the data collection process or other reasons, In some cases the issue would simply be because the abstracted data from the source is by itself an anomaly. Depending on the need the outliers need to either be removed or kept (after identification). If the occurrence of the outliers is due to the source, then removing them would lead to a lot of bias in the data and lead to loss of information.

Outlier Detection techniques and algorithms are extremely important when it comes to Data Mining because they can skew the distribution making it difficult for training machine learning algorithms. There are various algorithms and models to detect outliers, in this thesis we will be comparing one such algorithm – Isolation Forest which has a unique detection technique with more common algorithms like Local Outlier Factor (LOF) and Mahalanobis Distance.

We will use two data sets to evaluate the performance of the aforementioned algorithms. One will be a synthetic dataset which is self-generated, this will help to accurately measure the precision or accuracy of the algorithms as the data will be labelled which make it easier to compare. The second will be a Natural dataset which was from a real source which will help to ascertain the performance of the algorithms in real life situations. With this the analysis should provide concrete evidence and results of the effectiveness of the employed algorithms.

2. Types of Outliers

Outliers are generally those instances that have a different behavior or are isolated and far from the normal instance, they are usually few in number and can be isolated if the application calls for it. *(Divya D. and S. S. Babu,2016.)*

There are generally three types of categories in which all the outliers fall into:

2.1 Type 1: Global Outliers

A data point is a global outlier if it has its value far outside the entirety of the other data points.

It usually has a very high/very low value compared to the rest of the data points.

The graph below is an example of a Global outlier (point outlier)

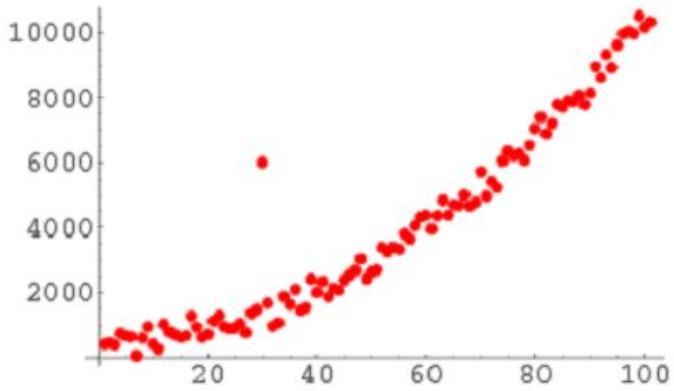

Fig 1: Example of a Global outlier.

2.2 Type 2: Contextual Outlier

If a data point has a significantly different value in a specific context or condition, then it is a Contextual outlier. So, it may not be an outlier if considered in a different context or condition.

Contextual outliers are difficult to spot without prior knowledge about the given data. If the domain knowledge is not sufficient then the contextual outlier would be a valid data point.

The graph below is an example of a contextual outlier:

If the values were not known to be temperatures in summer, then the said anomaly point would be considered valid (i.e., in a different context).

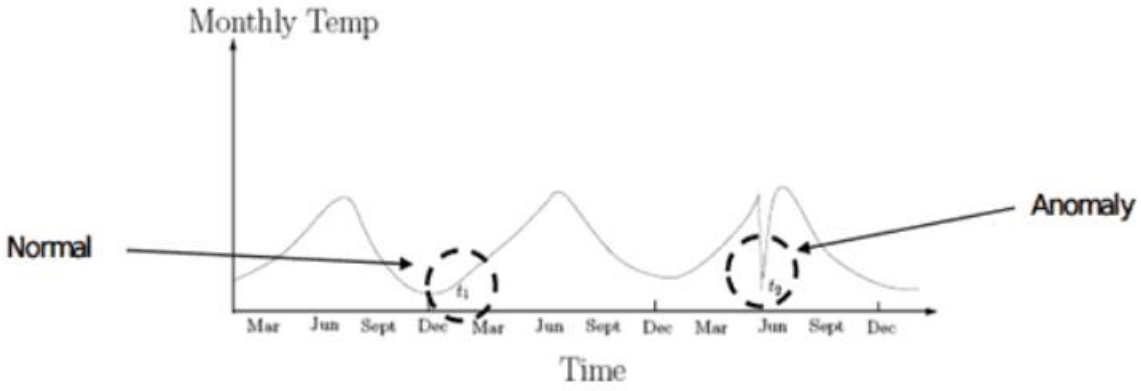

Fig 2: Example of a Contextual outlier – t1 and t2 are outliers only if the context is known.

2.3 Type 3: Collective outliers

If a collection of data points has extreme values when compared to the rest of the data points, then they are collective outliers. It is to be noted that the values of the individual data points are not by itself anomalous in either a given context/condition or a global sense.

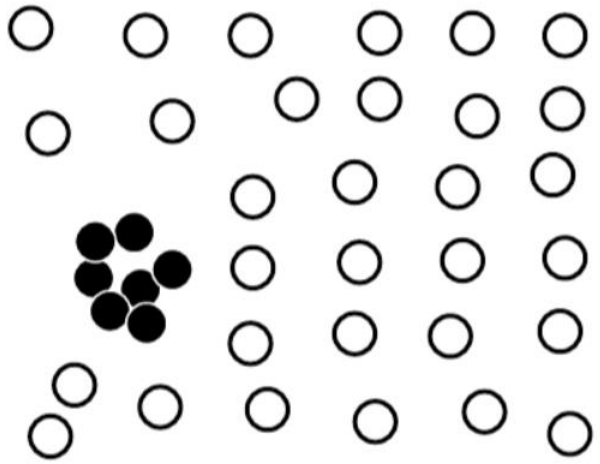

Fig 3: Example of a collective outlier.

3. Anomaly Detection Techniques

There are three Anomaly Detection Techniques which are widely used:

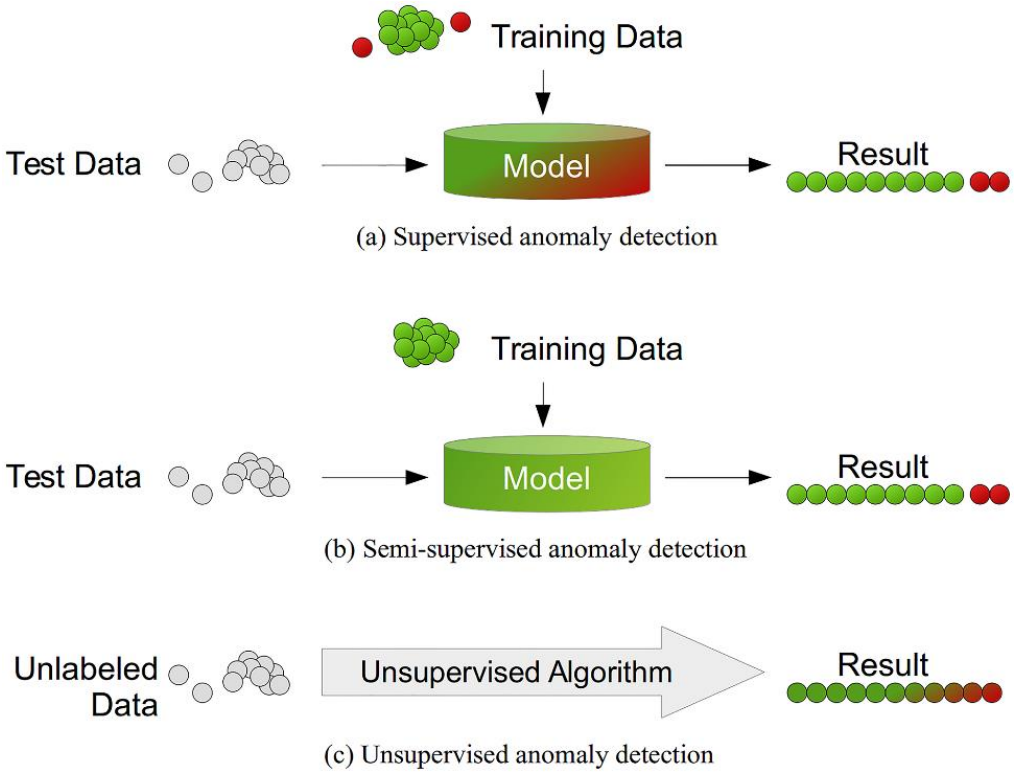

Fig 4:3 types of Anomaly Detection Techniques.

3.1 Supervised Anomaly Detection

Here the Training dataset is pre labelled with anomalies and the nominal data points. The supervised anomaly setting is the ideal setting where the outliers are known before the modelling process. Therefore, there are sets of points treated as anomalous, but are not yet identified for the model to train on. *(Jonathan Johnson. September 16,2020)*

Popular Supervised Anomaly Detection Algorithms:

1. Support Vector Machines
2. Decision Trees
3. K-Nearest Neighbours

3.2 Semi-Supervised Anomaly Detection

In this setting all the data points are assumed to be nominal, but it has anomalies present in the data set. This is a less ideal scenario where the data is cleaned and complete, but all the data points are assumed to be nominal data points. So, here the modeler takes up the responsibility to detect the anomalies in the data set.

3.3 Unsupervised Anomaly Detection

In this scenario the training data is unlabeled and consists of both nominal and anomaly points. This is the hardest setting where the model has to identify the nominal behavior from the anomalous behavior using the given data. There is not ground truth available to check if the labeled data is right or wrong. In Unsupervised anomaly detection the first task is to create order in the data set by creating clusters and then determine the groups that don't belong.

Popular Unsupervised Anomaly Detection Algorithms:

1. K-Means
2. LOC (Local Outiler Factor)
3. One Class support vector machine

Most of the existing anomaly detection models that are currently used construct a profile of normal instances, then identify those instances that do not conform to those normal instances as anomalies.

3.4 Types of Anomaly Detection Algorithms:

There are mainly two types of Algorithms widely used, Distance Based Algorithm and Density Based Algorithm. These are collectively called Proximity Based Algorithms.

3.4.1 Distance Based Methods:

Distance based algorithms take the takes into consideration the neighborhood of a data point, which is evaluated by a given radius. This point is then considered an outlier if the neighborhood does not have enough other points (which could be set by a threshold for example). *(F. Angiulli, S. Basta and C. Pizzuti, 2006)*

Formally as an equation if we want to consider point o,

let **r (r>0)** be a distance threshold and $\boldsymbol{\pi}$ **(0<$\boldsymbol{\pi}$<1)** be a fraction threshold.

An object **o** is DB(r,π) Distance Based:

$$\frac{\|o' | dist(o, o') \leq r\|}{\|D\|} \leq \pi$$

Where **dist** is the distance measured (usually Euclidean). **o'** is the point with which the distance of the point **o** is being measured with. This approach takes **o(n^2)** time, so it is extremely time intensive if the data set is huge and of high dimensionality.

3.4.2 Density Based Methods:

Density Based Algorithms takes into consideration the density of a point with respect to the density of its neighbors. The point is considered an outlier if its density is much lower (which could be set with a threshold) than the density of its neighbors. *(Bo Tang, Haibo He,2017)*

The Assumption here is that the density around the normal point is the same or similar to the density of its neighbors but the density of an outlier is significantly different from that of its nearest neighbors.

To simplify this method, we'll look at the formulae involved:

$$N_k(o) = [o' | o' \in D, dist(o, o') \leq dist_k(o)]$$

N_k(o) is the number of neighbors in the neighborhood set by **k** of point **o**,

dist(o,o') is the distance of **o'** from the point in consideration **o**.

dist_k(o) is the distance from the point **o** and its k-nearest neighbors, the neighborhood of **o** contains all the points **(o')** of the distance to **o** which is not greater than **dist_k(o)** the kth distance of **o**.

The density is calculated using the average distance from the points in **N_k(o)** to **o**. If the point **o'** is too close to the point **o** there would be statistical fluctuations so to avoid that we use a smoothing constant and switch to reachability distance,

we calculate the reachability distance of a point **o'** to the point in question **o**,

$$reachdist_k(o, o') = max[dist_k(o), dist(o, o')]$$

k is a user specified parameter with is needed for the smoothing effect. **k** specifies the minimum number of neighbors to determine the density of a point. (Reachability distance is not symmetric)

Local reachability density **lrd_k(o)** is then calculated by,

$$lrd_k(o) = \frac{\|N_k(o)\|}{\sum_{o' \in N_k(o)} reachdist_k(o, o')}$$

Now the Local reachability density of the point **o** is compared with that of its neighbors to measure the degree to with it could be considered an outlier.

Traditional Statistical Anomaly detection models like Local Outlier Factor (LOF), K-NN, one class SVM etc., have a high demand on the memory and time taken because of either Distance calculation (for Distance-based algorithm) or of Density Calculation (for Density based algorithms). The effect of these indicators is even higher when we consider multi-dimensional data. Also, another important drawback of these algorithms is that they are prone to outlier swamping and outlier masking.

Outlier swamping happens when the outliers are closer to the normal instances, so they share some commonality with them, and Outlier masking happens when the outliers are clustered together so they become hard to detect.

3.5 Isolation Forest (iForest) – Anomaly Detection

Isolation Forest is an unsupervised machine learning algorithm that uses Decision Trees to identify anomalies. It is neither a Distance based algorithm, nor a Density based Algorithm.

The Isolation Forest algorithm uses two basic properties of all anomalies:

1. Anomalies are few in number and are in minority compared to the nominal values.
2. Anomalies have very different base values than the nominal values.

Isolation Forests constructs a tree for each data point, and because of the susceptibility of the anomalies to isolation because of their typical behavior, they are isolated closer to the root of the tree. The normal points would be isolated towards the deep end of the tree. (*F. T. Liu, K. M. Ting and Z. Zhou, 2008.)*

The proposed algorithm builds an ensemble of trees for each data point and the anomalies are those points with the shortest average path length.

This algorithm needs only two variables:

1. The total number of trees to be built.
2. The size of the sampling to be used.

Breakdown of the steps involved in the Isolation forests:

The following example has two dimensions Q1 and Q2.

Step 1: Sampling the data set for training.

1. The first step is to sample the data set for training a decision tree.
2. The sampling can depend on the data set (essentially for a noisy data set a higher proportion of data set is preferred)

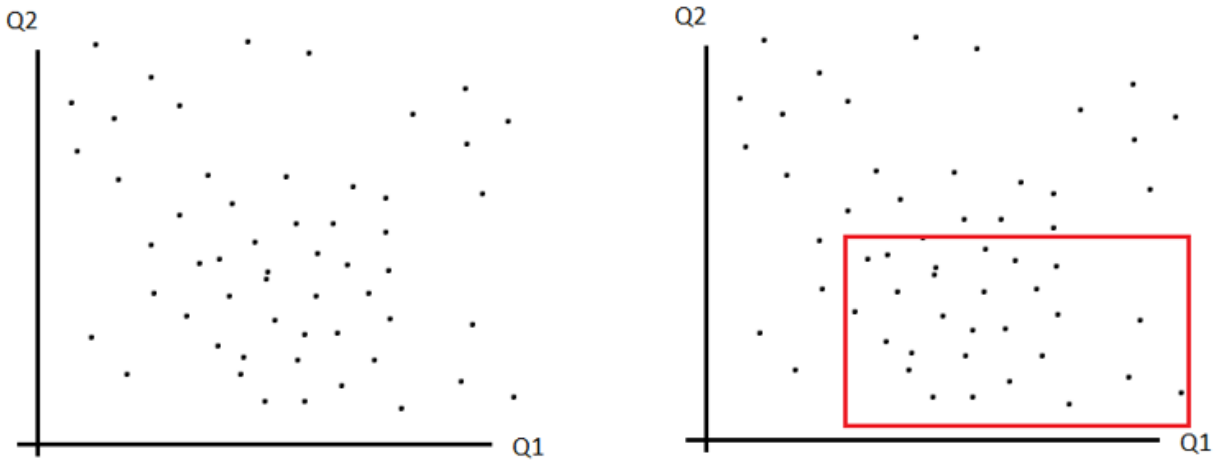

Fig 5: Sub-sampling the dataset.

Step 2: Form the Decision trees with the selected sample.

1. Here we for the tree with the selected sample of data points
2. The next step is to have a random choice of a feature (Q1 or Q2, or a random choice of a Q1/Q2 value between its max or min value (Q1'))

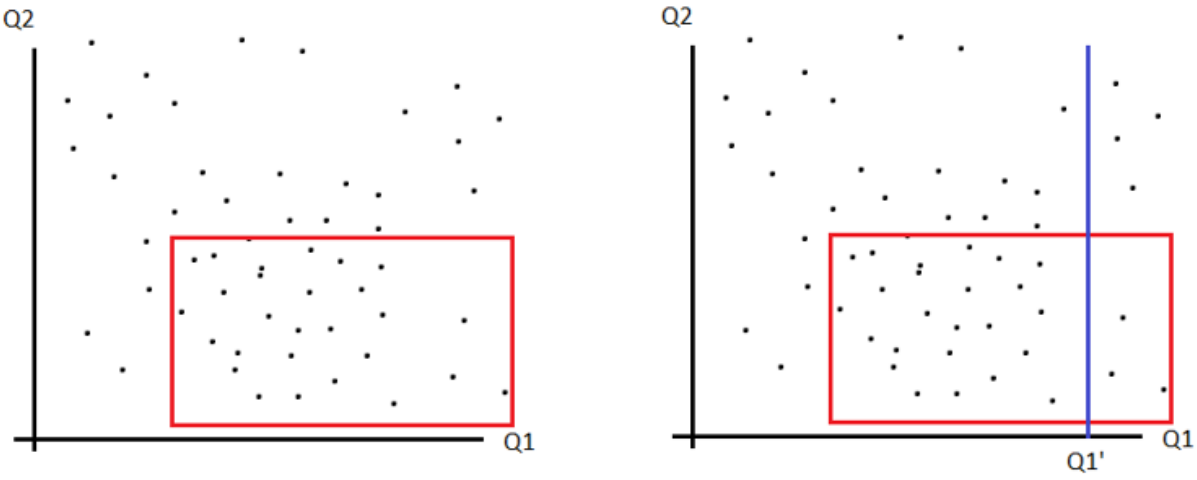

Fig 6: Building the Decision trees.

Step 3: Repeat the above step iteratively.

1. Perform the above step for the two sub-sampled data formed based on the binary split from step 2.
2. The Anomalous points which are few and different will be isolated quicker like the point in the lower right corner.
3. Essentially it takes less path length for the anomalies to get isolated.
4. Perform the same steps iteratively to create a forest of trees.

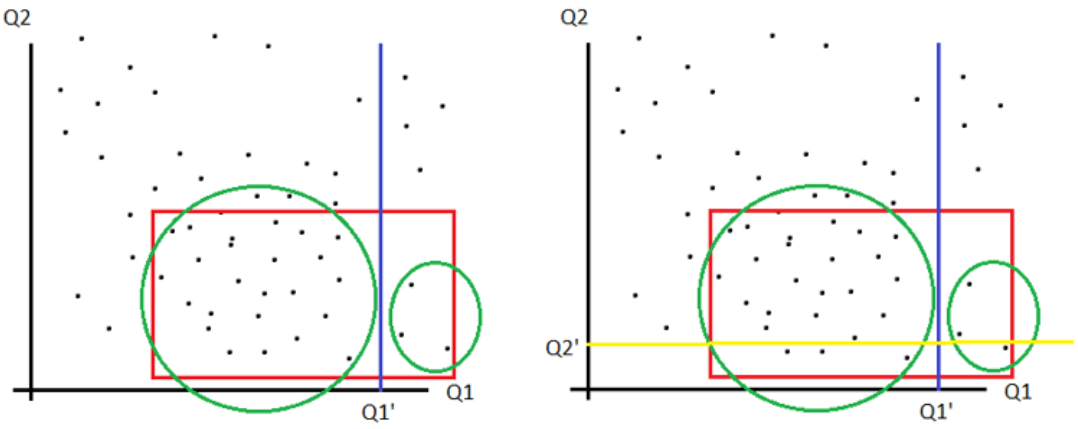

Fig 7: Building more decision trees to create a forest of trees (until specified).

Step 4: Calculate the Anomaly Score.

1. Pass all the data points into the trained forest model for each tree that was formed.
2. Calculation of the Anomaly Score (S):

$$Anomaly\ Score\ (S) = 2^{\frac{-E(h(k,m,N)}{c(n)}}$$

$$,where\ c(n) = 2(\ln(n-1) + 0.5772156649) - 2\left(\frac{n-1}{n}\right)$$

$$,where\ n\ is\ a\ number\ of\ data\ points\ in\ a\ chosen\ sample$$

$$,where\ E(h(k,m,N)) = \frac{\sum_{i=1}^{N} \begin{cases} if\ k == 1, \sum_{j=1}^{M} 1 \\ else,\ \sum_{j=1}^{M} 1 + c(k) \end{cases}}{N}$$

$$,where\ N\ is\ a\ total\ number\ of\ trees$$

$$,where\ M\ is\ a\ total\ number\ of\ binary\ splits$$

$$,where\ k\ is\ a\ total\ number\ of\ data\ points\ in\ the\ final\ node\ (exit\ node)$$

   - If S is closer to 1 then it is an anomaly
   - If S is lower than 0.5 then it is a normal point
3. Calculate the Anomaly Score for each of the trees and the final Anomaly Score is the average of its values across all the trees.
4. An Anomaly gets a score closer to 1. *(Mensi A., Bicego M., 2019)*

The Major Advantages of using Isolation Forests:

1. The isolation of the anomalies of the Isolation Forests allows them to build partial sub samples to an extreme extent which is not possible in any other algorithm. Since the normal points lie at the end of a tree a large part of the tree need not be constructed when the anomalies are detected closer to the root of the tree. This small sample size also reduces the effect of outlier swamping and masking.
2. Isolation forests does not depend on distance calculation nor density calculation to detect anomalies. This helps to reduce the computational cost by a substantial amount which makes an impact with huge data sets and high dimensionality.

3.6 LOF (Local Outlier Factor):

The LOF of a point gives the density of that point in comparison to the density of its neighbors (the neighbors are taken using the K-NN algorithm, so the choice of K is up to the modeler). If the density of the said point is lower than the densities of its neighbors, then the point is far away from dense areas, hence it is an outlier. *(Zhangyu.C, Chengming.Z, et al., 2019)*

$$LOF_k(\boldsymbol{o}) = \frac{\sum_{\boldsymbol{o}' \in N_k(\boldsymbol{o})} \frac{lrd_k(\boldsymbol{o}')}{lrd_k(\boldsymbol{o})}}{\|N_k(\boldsymbol{o})\|} = \sum_{\boldsymbol{o}' \in N_k(\boldsymbol{o})} lrd_k(\boldsymbol{o}') \cdot \sum_{\boldsymbol{o}' \in N_k(\boldsymbol{o})} reachdist_k(\boldsymbol{o}' \leftarrow \boldsymbol{o}).$$

LOF(o) is the local outlier factor value,

N_k(o) is the neighbors within the neighborhood k of the point **o**,

reachdist_k(o' to o) is the reachability distance from the point **o** to **o'**,

lrd_k(o) is the local reachability density of the point **o**,

lrd_k(o') is the local reachability density of the point **o'**.

It calculates the average of the ratio of the Local reachability density of the point **o** and those of its k-nearest neighbors. So, the lower the local reachability density of **o** and the higher the local reachability density of its k-nearest neighbors, the higher the LOF value will be.

It is a score that tells how likely a point is an outlier.

If LOF > 1 then outlier, If LOF closer to 1 then not an outlier.

For example, in the following figure, the neighbors of the point on the top right – which are the four points to the bottom left – have a much higher density than the considered point, hence it is an outlier.

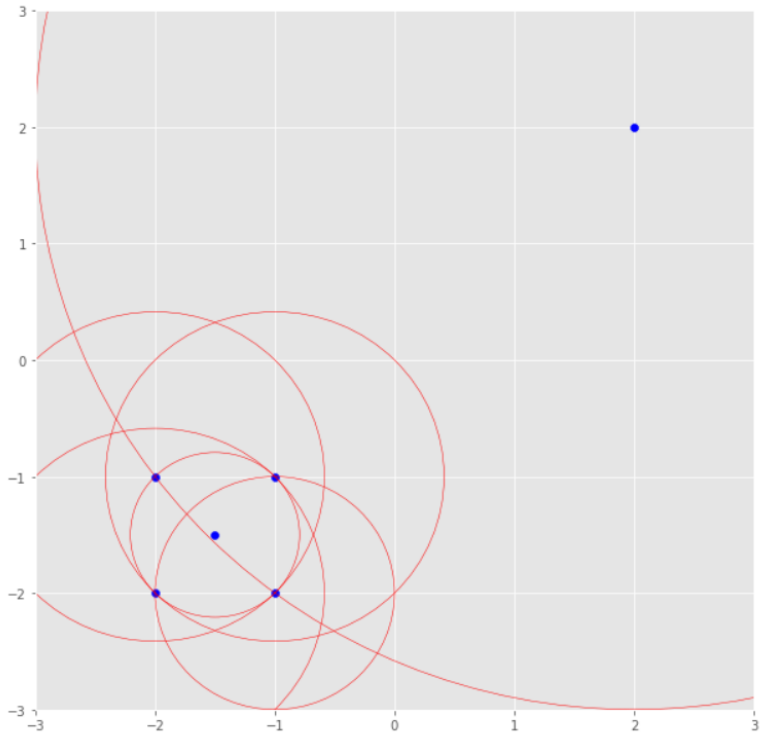

Fig 8: Example of the LOF algorithm (the point on the top right corner is an anomaly).

3.7 Mahalanobis Distance:

The Mahalanobis Distance measures how far each point is from the center of the data, if the Distance is large then it is regarded as an outlier.

Mahalanobis also converts normal distributed data to standard normal distribution with only uncorrelated variables, this is done by taking the inverse covariances of the distribution. It works really well with multivariate data because of this reason.

The Formulae associated with Mahalanobis Distance:

$$D^2 = (x - m)^T \cdot C^{-1} \cdot (x - m)$$

D^2 is the square of the Mahalanobis Distance,

x is the vector of the observations,

m is the vector of the mean values of the independent variables,

C^(-1) is the inverse of the covariance matrix of the independent variables.

The following figure is an example of the Mahalanobis Distance algorithm, all those points that lie outside the outlier threshold (the green ellipse) is an outlier.

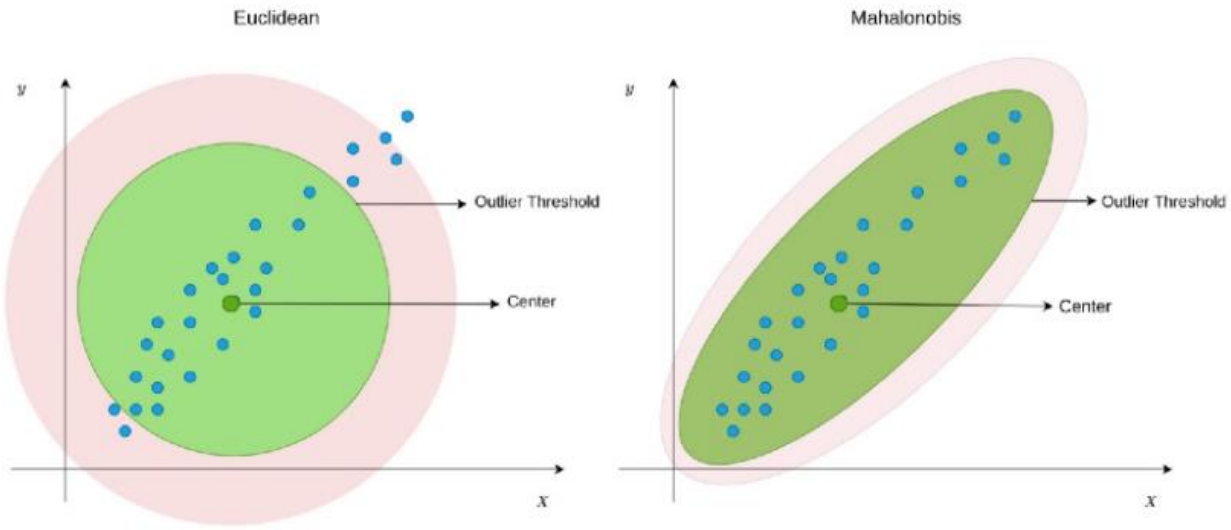

Fig 9: Example of the Mahalanobis distance method – any point outside the green ellipse is an outlier.

4. Analysis Synthetic Dataset

Two data sets will be used to compare the performance of Isolation forests algorithm with Local Outlier Factor (LOF) algorithm and Mahalanobis algorithm.

The first data set is a Synthetic data set which has self-generated anomalies. This would make it easier to measure the performance as we know exactly where the anomalies are present. This data set has random values between 0 and 4 and are in two different clusters, one cluster is from (0,1) and the other cluster is from (2,4).

These clusters were created to try and induce the masking and swamping effects of the outliers.

This is the Training set with the normal value points.

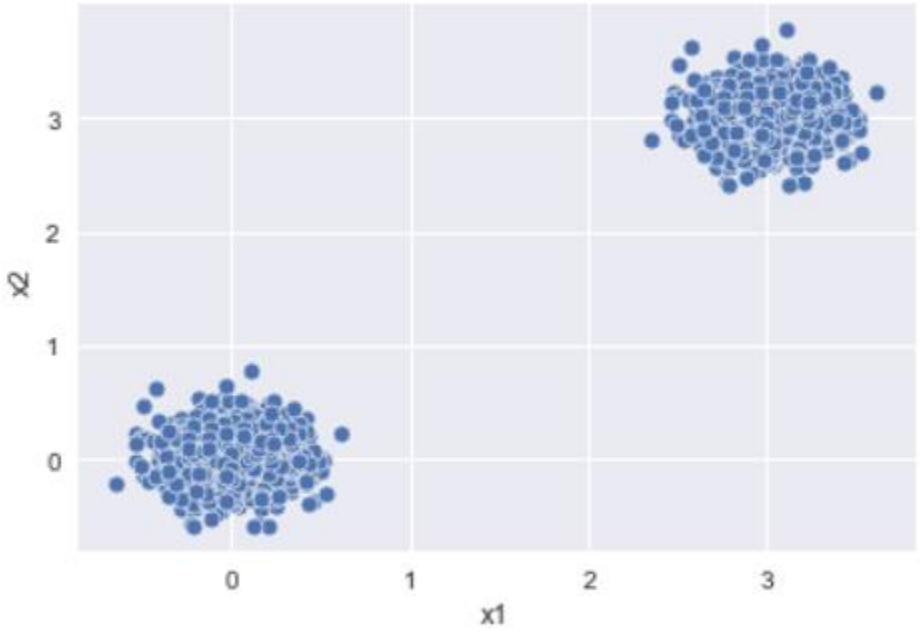

Fig 10: Self-generated Synthetic dataset with two clusters.

Now we will add anomalies to this data set which will also be in a random fashion. As we can see from the graph below, the red dots that lie within the clusters are swamped outliers which would be difficult to isolate. There are also small clusters of outliers which induces a masking outlier effect.

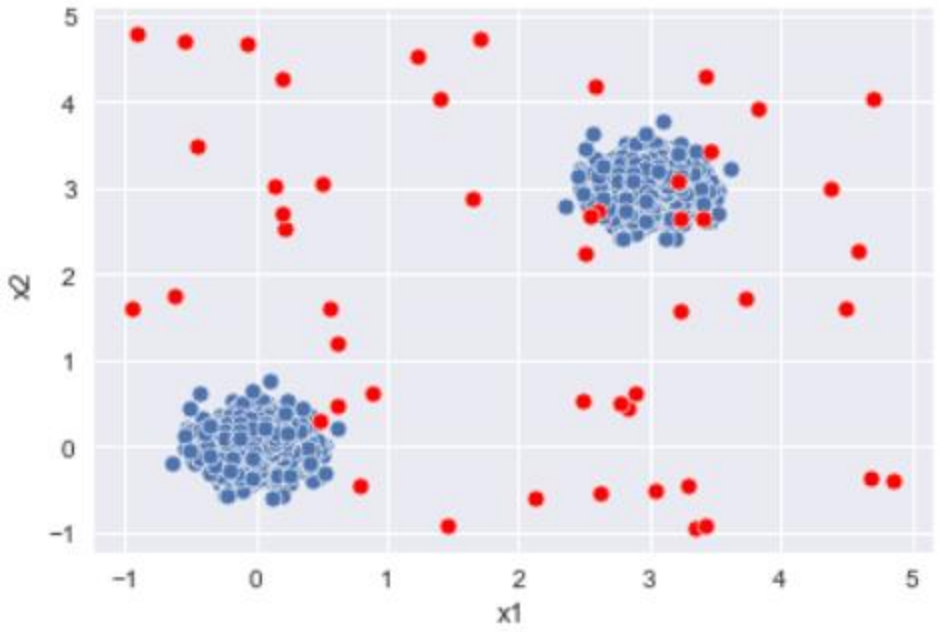

Fig 11: Synthetic Dataset with generated random anomalies (red dots represent the anomalies).

4.1 Result

	Algorithm	Precision/Accuracy
0	iForests	0.98
1	LOF	0.92
2	Mahalanobis Dis	0.96

Fig 13: Table of the algorithms and their respective Precision.

iForests correctly identified **98%** of the outliers in this data set. As it was a 2-Dimensional data set Mahalanobis distance was very effective in the anomaly detection with **96%** accuracy followed by LOF with a **92%** accuracy.

For the Duration of the execution of the algorithm,

```
Duration Isolation Forest: 0:00:00.449440

Duration LOF: 0:00:00.098602

Duration Mahalanobis: 0:00:01.395421
```

Isolation forests have clearly overcome both the swamping and the masking outlier effects with high accuracy in the anomaly isolation. We can see that Local Outlier Factor was the fastest to compute with 0.09 secs as the number of neighbors was set to 20 (It is highly dependent on the number of neighbors). However Local Outlier Factor is prone to the masking and swamping effects as it is density based (the density of the masked outlier would be high hence making it difficult to be detected). Mahalanobis Distance isolates the anomalies efficiently as well with a 2 standard deviation threshold.

5. Analysis Natural Data Set

The second data set is the NAB Data Corpus dataset from Kaggle which is a time series data set for research in anomaly detection methods. Data is ordered, time stamped for ease and has single value metrics.

It is a real-world dataset from the AWS (Amazon Web Services) server metrics which includes twitter volume and advertisement clicking metrics.

Name of the data set: Real AWS Cloudwatch

It contains the time series of the AWS server metric collected by the Amazon Cloudwatch service which includes the CPU Utilization.

It is an unlabeled data set, so it will not be possible to evaluate the precision or accuracy for each algorithm. But here we could see how many anomalous points are being detected by each algorithm in a given time scale, so both speed and the number of isolations can be evaluated.

Following is an overview of the time series dataset:

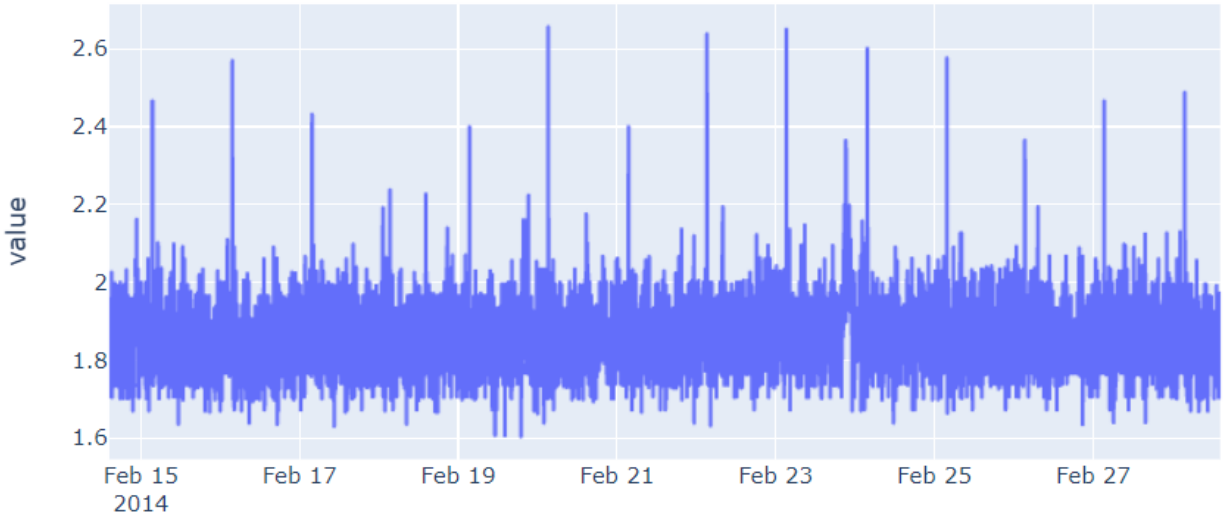

Fig 12: NAB Time series data set with CPU Utilization.

5.1 Result

The Red dots represent the anomalies identified by the respective algorithms.

5.1.1 Isolation Forest:

Isolation Forests isolates all those points at the extremes that are closer to the root of the tree, so it has really good performance which can be further enhanced by hyperparameter tuning.

We can also observe anomalies which are closely clustered together have been isolated efficiently so it overcomes the masking outlier problem. The duration taken was 0.4 secs due to partial building of trees from random sub-sampling, so it scales really well with large amounts of data.

Total number of anomalies detected: 41.

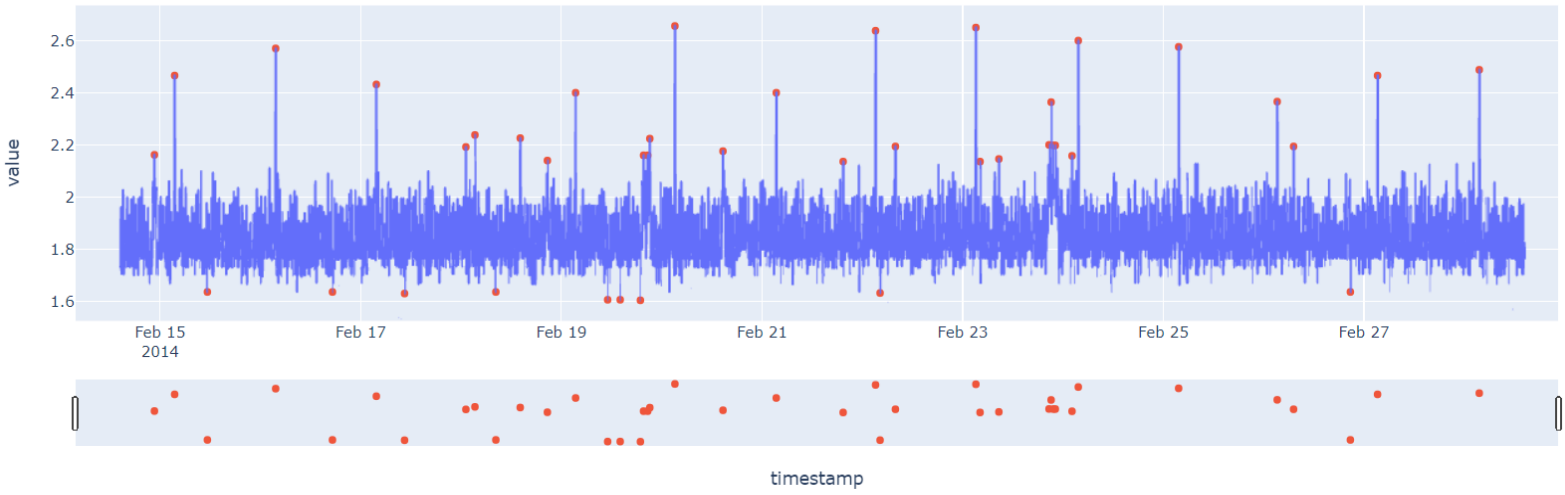

Fig 14: The anomalies detected by the Isolation Forest Algorithm.

Duration of the execution:

```
Duration Isolation Forest: 0:00:00.447666
```

5.1.2 Local Outlier Factor:

The LOF algorithms isolates some points and has a decent performance. It does not do well with the masking and swamping outlier effects that exist in the dataset and hence the lower number of detected anomalies. This also leads to a lower duration of execution time with 0.02 secs. The performance can indeed be improved with hyperparameter tuning and setting the ideal number of neighbors for each point which requires more insight into the given data.

Total number of anomalies detected: 33.

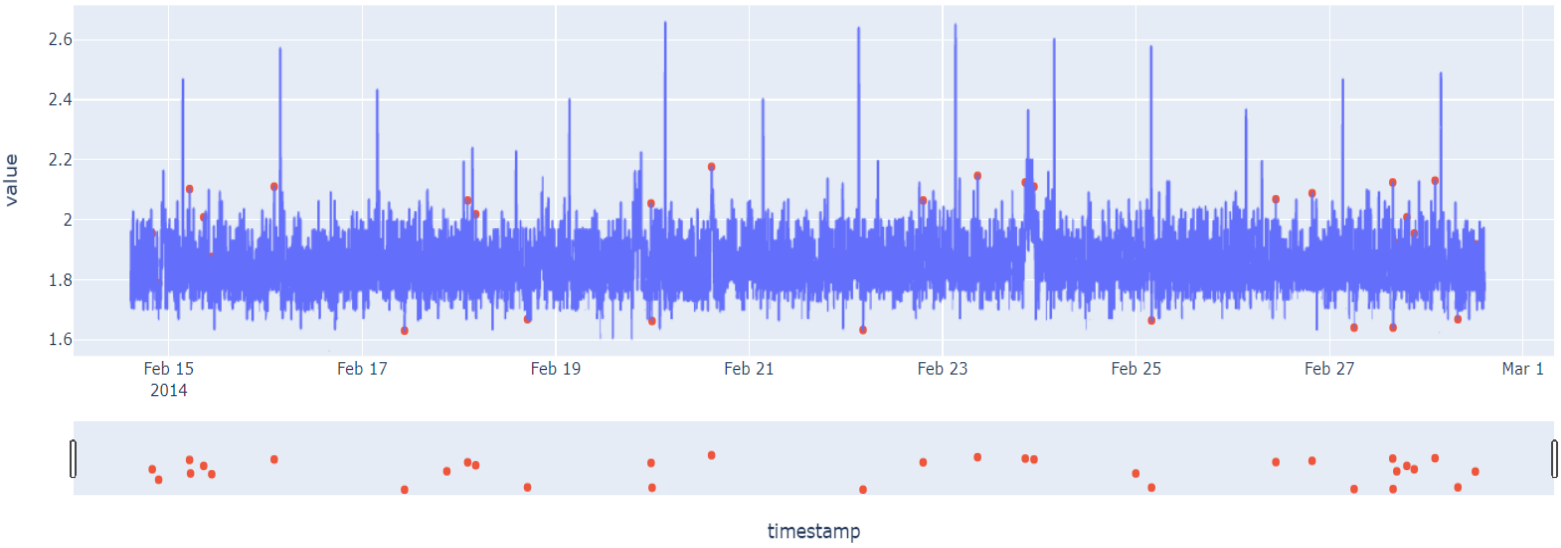

Fig 15: The anomalies detected by the LOF Algorithm.

Duration of the execution:

```
Duration LOF: 0:00:00.021947
```

5.1.3 Mahalanobis Distance:

Mahalanobis Distance algorithm identifies all the peaks in one direction above a given threshold which was set to 2 standard deviations (so a Confidence Interval of 95% was chosen). It does however overcome the swamping and masking outlier effects, this could be due to the presence of inverse covariance that is present in the calculation which avoids correlated points from the algorithm. The duration of the execution is nominal with 0.13 secs.

Total number of anomalies detected: 141.

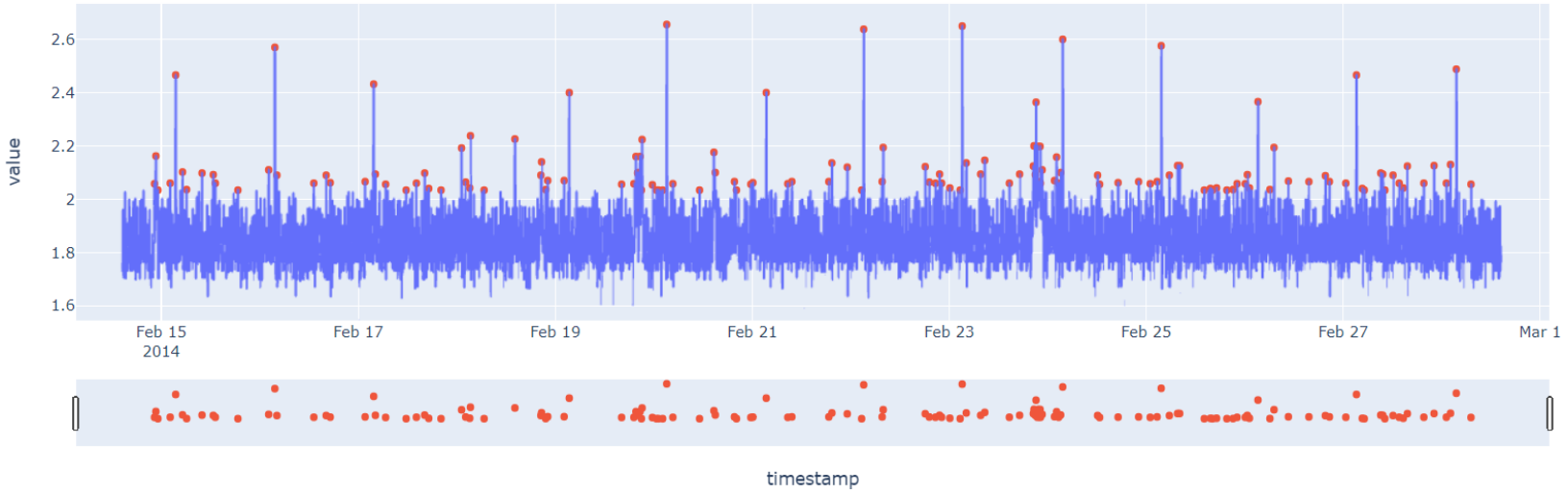

Fig 16: The anomalies detected by the Mahalanobis Distance Algorithm.

Duration of Execution:

```
Duration Mahalanobis: 0:00:00.135976
```

5.1.4 A Comparative graph:

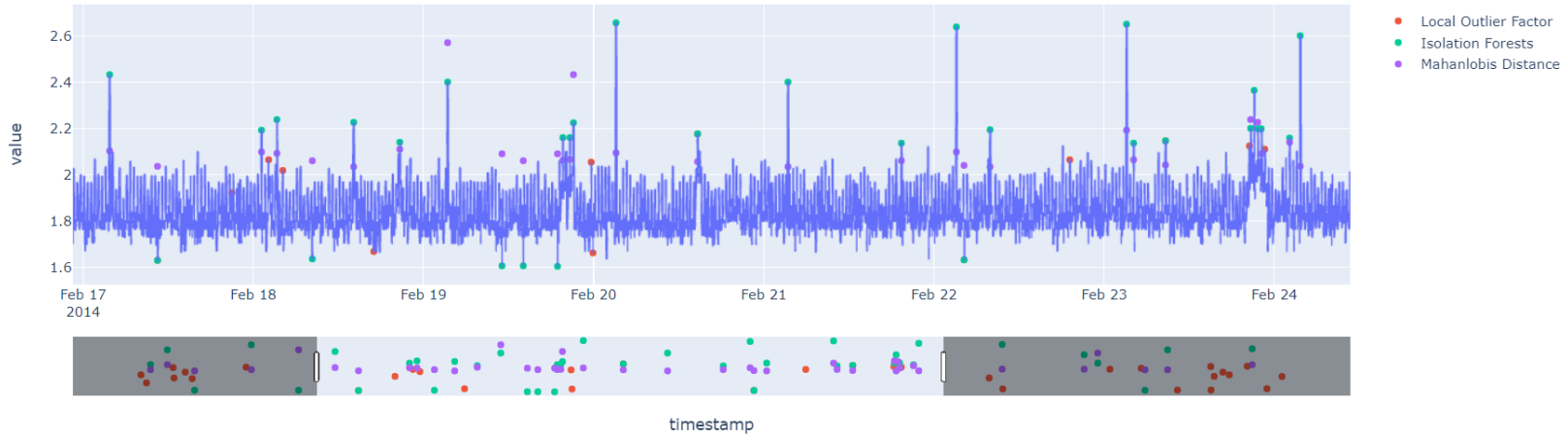

Fig 17: A comparative graph which has all the anomalies detected by the respective algorithms.

This Graph illustrates all the anomalies detected by all three algorithms within a specified timeframe.

Some observations include:

1. Local Outlier Factor has very few outliers detected (33) because of masking and swamping outlier effects.
2. Mahalanobis Distance does not consider both the ends of the peaks and identifies a lot of normal points as anomalies.
3. Isolation Forest performs the best and identifies the anomalous points in both the direction with high accuracy.

We can see that Isolation forests has identified only the far away point as anomalies unlike the Mahalanobis Distance or the LOF with incorrect and fewer isolation (in the case of LOF).

From the observed results it is clear that isolation forests far outperform both the Distance-based algorithm – Mahalanobis Distance and the Density-based algorithm – Local Outlier Factor in both the synthetic data set and the natural Kaggel data set.

6. Conclusion

This Thesis evaluates the performance of the Isolation forests algorithm in comparison with a Density based model – Local Outlier Factor (LOF) and a Distance based model – Mahalanobis Distance. The concept of typical outliers and of their different types have been discussed in the literature of this thesis. All different types of Anomaly Detection methods were also seen in brief to understand the working of the algorithms being compared and to which category they belong.

The working of Isolation Forests was studied and its high efficiency in detecting anomalies was shown with the help of two datasets – a synthetic self-generated data set and a natural dataset from Kaggle. By taking advantage of the anomalies' nature of 'few' and 'different', the isolation forest algorithm isolates those points which are close to the root of the tree as anomalies and those points farther away from the root of the tree as normal points. This allows for quick and efficient isolation of the anomalies by using partial models (because of sub-sampling). It also overcomes the effects of swamped and masked outliers which other algorithms struggle to deal with. It also scales really well with huge amounts of data and high dimensionality.

All these capabilities make Isolation Forests highly desirable for real life applications which consists of large databases.

7. Bibliography

1. Divya D. and S. S. Babu, "Methods to detect different types of outliers,", International Conference on Data Mining and Advanced Computing (SAPIENCE), 2016, pp. 23-28, doi: 10.1109/SAPIENCE.2016.7684114.
2. Jonathan Johnson. September 16, 2020. Anomaly Detection with Machine Learning. Bmc. https://www.bmc.com/blogs/machine-learning-anomaly-detection.
3. F. Angiulli, S. Basta and C. Pizzuti, "Distance-based detection and prediction of outliers," in IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 2, pp. 145-160, Feb. 2006, doi: 10.1109/TKDE.2006.29.
4. Bo Tang, Haibo He, A local density-based approach for outlier detection, Neurocomputing, Volume 241, 2017, Pages 171-180, ISSN 0925-2312.
5. F. T. Liu, K. M. Ting and Z. Zhou, "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining, 2008, pp. 413-422, doi: 10.1109/ICDM.2008.17.
6. Mensi A., Bicego M. (2019) A Novel Anomaly Score for Isolation Forests. In: Ricci E., Rota Bulò S., Snoek C., Lanz O., Messelodi S., Sebe N. (eds) Image Analysis and Processing – ICIAP 2019. ICIAP 2019. Lecture Notes in Computer Science, vol 11751. Springer, Cham. https://doi.org/10.1007/978-3-030-30642-7_14.
7. Zhangyu.C, Chengming.Z, et al., Outlier detection using isolation forest and local outlier factor, RACS '19: Proceedings of the Conference on Research in Adaptive and Convergent Systems, 2019, pp. 161-168, doi. 10.1145/3338840.3355641.
8. Sahanand Hariri, Matias C.K, Isolation Forest for Anamoly Detection, June 21, 2018, LSST Workshop, NCSA, UIUC.