Comparing the Performance of Anomaly Detection Algorithms

Bhadri Naarayanan P[1]
[1]UG Scholar,
Department of Information Technology,
Sri Venkateswara College of Engineering,
Sriperambudur, Tamil Nadu, India

Clement Franklin D C[2]
[2]UG Scholar,
Department of Information Technology,
Sri Venkateswara College of Engineering,
Sriperambudur, Tamil Nadu, India

Gouvtham N[3]
[3] UG Scholar,
Department of Information Technology,
Sri Venkateswara College of Engineering,
Sriperambudur, Tamil Nadu, India

Ms. Sharon Femi P[4]
[4]Assistant Professor,
Department of Information Technology,
Sri Venkateswara College of Engineering,
Sriperambudur, Tamil Nadu, India

***Abstract* - An Anomaly is a data point which differs in characteristics from other data points in the dataset. The detection of anomaly plays an important role in machine learning. But most of the algorithms provide anomaly detection only with limited generalization capacity. In this paper, In this paper, we compare the efficiency of anomaly detection methods which has better robustness. The three outlier detection algorithms used are Local Outlier Factor, Isolation Forest and Autoencoders. Based on the accuracy, recall, precision, F1 score of the algorithms, the comparison graph is constructed for the three datasets and the efficient algorithm is determined.**

Key Words: Anomaly, Machine learning, Outliers

I. INTRODUCTION

Anomaly detection is a research area in data mining. It is also called as outlier detection, novelty detection or deviation detection. According to the standard definition of Hawkins [11], “An outlier is an observation that deviates so much from other observations as to arouse suspicion when comparing to the results generated by other different mechanism”. The main applications of outlier detection are to identify credit card frauds, network intrusion, identification of patients who has abnormal symptoms due to a particular disease etc. Anomalies are a set of data that have different characteristics from other set of data in the data sample. Hence anomaly detection has become a prominent area of research in data mining. Though, the quantity of outliers are very less compared to the normal data, the detection of these points have become important.

The different types of outliers are Point outliers, Contextual outliers and Collective outliers [12]. Though, the individual data instance in a collective outlier may not be outliers by themselves, yet their occurrence together as a collection is anomalous. The outlier detection methods can be Statistical-based, Distance based, Density based and cluster based methods. In general, a single outlier detection algorithm may not be best suited for all the different data scenarios of the real-world datasets. Outlier detection can be done for univariate or multivariate data. But, the real world data is heterogeneous, where the data point can have both

This paper is about the large-scale anomaly detection using anomaly detection algorithms. The objective of this paper is to determine the efficient algorithm on the occurrence of anomalies on a large scale. There are currently numerous algorithms for determining the anomalies. But the present algorithms work on loss of information. There are algorithms which work on a mixture of data but are not that much efficient. The algorithm proposed will provide a final score which enables us to determine the occurrence of anomalies in a more efficient manner.

At present there are many classifier detection algorithms that are used in research and development sector. The algorithms for outlier detection are very less, the algorithms that will be proposed will produce one of the efficient algorithms in the three algorithms for three different datasets.

The rest of the paper is organized as follows. Section II discusses the related research carried out in this area. Section III presents the proposed system architecture. Section IV elaborates the detailed description of the proposed system and the experimental results are discussed in Section V. Section VI evaluates the performance of different algorithms. Finally, Section VII concludes the paper by highlighting the research contributions and future plans to extend this work further.

II. LITERATURE REVIEW

Chandola et al gives a survey of the anomaly detection techniques [1]. It gives ideas for different ways in which the problem of anomaly detection has been formulated in literature, and have attempted to provide an overview of the huge literature on various techniques.

Xie et al gives a survey of Fast Tensor Factorization for Accurate Internet Anomaly Detection [2]. The paper provides the initiative to investigate the potential and methodologies of performing tensor factorization for more accurate Internet anomaly detection. It gives a model to the traffic data as a three-way tensor and formulates the anomaly detection problem as a robust tensor recovery problem with the constraints on the rank of the tensor and the cardinality of the anomaly set.

Huang et al gives the survey of Novel Outlier Cluster Detection Algorithm [3]. This paper provided the idea to novel an outlier cluster detection algorithm called ROCF based on the concept of mutual neighbour graph and on the idea that the size of outlier clusters is usually much smaller than the normal clusters. The formal analysis and experiments show that this method can achieve good performance in outlier detection.

Agarwal et al gives a survey of outlier ensembling[4]. In some cases, ensemble analysis techniques have been implicitly used by many outlier analysis algorithms, but the approach is often buried deep into the algorithm and not formally recognized as a general-purpose meta-algorithm. The idea of various methods which are used in the literature for outlier ensembles and the general principles by which such analysis can be made are more effective.

Zhang et al has proposed a model containing open source software for Multi-class Imbalance learning [5]. It provides users with different categories of multi-class imbalance learning algorithms, including the latest advances in the field.

Rayana et al has proposed a new ensemble approach for outlier detection in multi-dimensional point data [6], which provides improved accuracy by reducing error through both bias and variance. In this paper, a sequential ensemble approach called CARE that employs a two-phase aggregation of the intermediate results in each iteration to reach the final outcome is proposed. Through extensive experiments on sixteen real-world datasets mainly from the UCI machine learning repository, it is shown that CARE performs significantly better than or at least similar to the individual baselines.

Agarwal et al gives a survey of Theoretical Foundations for Outlier Ensembles [7]. It provides the investigation of the theoretical underpinnings of outlier ensemble analysis. It also discusses the impact of the combination function and discuss the specific trade-offs of the average and maximization functions. These insights are used to propose new combination functions that are robust in many settings.

Min et al gives a survey of Random Effects Logistic Regression Model for Anomaly Detection [8]. This paper proposes a random effects logistic regression model to predict anomaly detection. The research is based on a sample of 49,427 random observations for 42 variables of the KDD-cup 1999 (Data Mining and Knowledge Discovery competition) data set that contains ‘normal’ and ‘anomaly’ connections. The proposed model has a classification accuracy of 98.94% for the training data set, while that for the validation data set is 98.68%.

Paulheim et al gives a survey of decomposition of Outlier detection into a set of Supervised Learning [9]. It shows that for numerical datasets, the approach can be used in conjunction with arbitrary regression learning algorithms, that it reliably yields good results using M5’ (regression trees) or isotonic regression as base learners, and that its results are invariant to the adding of irrelevant noise attributes.

Campus et al gives a survey of Unsupervised Outlier Detection [10]. Little is known regarding the strengths and weaknesses of different standard outlier detection models, and the impact of parameter choices for these algorithms. Based on the overall performance of the outlier detection methods, we provide a characterization of the datasets themselves, and discuss their suitability as outlier detection benchmark sets. We also examine the most commonly-used measures for comparing the performance of different methods, and suggest adaptations that are more suitable for the evaluation of outlier detection results.

III. PROPOSED SYSTEM ARCHITECTURE

In the proposed system architecture as seen in Fig -1, the datasets are identified. The identified dataset is then used for preprocessing of the data which is then used to find the correlation of the dataset. This dataset is used for finding the anomalies using the LOF, Auto encoder and Isolation forest algorithms. The dataset is trained and tested in 7:3 ratio (i.e. 70% of the data is used for training the algorithm and 30% of the data used for testing). The anomaly detection algorithms is applied to the random data samples and the accuracy will be generated. These algorithms are applied to the raw data and preprocessed data. Finally, the two results of the will be used to compare along with their accuracy scores, recall score, precision and the F1 score.

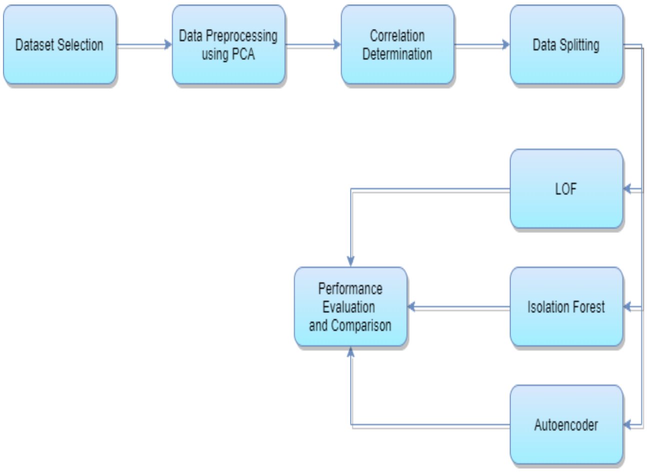

Dataset Selection
Data Preprocessing using PCA
Correlation Determination
Data Splitting
LOF
Performance Evaluation and Comparison
Isolation Forest
Autoencoder

Fig -1: Proposed System Architecture

IV. MODULES DESCRPTION

A. Dataset Description

The dataset is downloaded from two repositories, namely the Kaggle and the UCI repositories [15]. The three datasets identified are the breast cancer dataset, corona virus dataset, heart disease dataset. The corona virus dataset contains anomalies with respect to the corona virus pandemic worldwide. The breast cancer dataset contains the records of women who are likely to possess the disease. The heart disease dataset contains parameters like age, pressure sugar etc. which will lead to the anomaly factor of a person getting the disease or not.

B. Data Preprocessing

Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues. Data preprocessing prepares raw data for further processing. Data preprocessing is used database-driven applications such as customer relationship management and rule-based applications (like neural networks). In the data, the values may either be higher or lower as the data is uneven. Thus, the data is normalized by fitting them between the range of 0 and 1 so the data will be ready for comparison using min-max normalization.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (1)$$

C. Principal Component Analysis

Principal components analysis is essentially just a coordinate transformation. The original data are plotted on an X-axis and a Y-axis. For two-dimensional data, PCA seeks to rotate these two axes so that the new axis X' lies along the direction of maximum variation in the data. PCA requires that the axes be perpendicular, so in two dimensions the choice of X' will determine Y'. You obtain the transformed data by reading the x and y values off this new set of axes, X' and Y'. For more than two dimensions, the first axis is in the direction of most variation; the second, in direction of the next-most variation; and so on. They are related to Eigen values and Eigen vectors of the covariance matrix you just calculated.

D. Correlation Determination

There may be complex and unknown relationships between the variables in your dataset. It is essential to find out and quantify the degree to which variables in your dataset are dependent upon each other. The performance of some algorithms can deteriorate if two or more variables are tightly related, an interest will also arise in the correlation between input variables with the output variable in order provide insight into which variables may or may not be relevant as input for developing a model. The correlation metric is

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[n\Sigma x^2 - (\Sigma x)^2][n\Sigma y^2 - (\Sigma y)^2]}} \quad (2)$$

where r is the correlation coefficient, x and y are the variables.

E. Data Splitting

The data used to build the final model usually comes from multiple data samples. In particular, three data sets are commonly used in different stages of the creation of the model. The model is initially fit on a training data sample, that is a set of examples used to fit the parameters of the model. In practice, the training data sample often consist of pairs of an input vector and the corresponding output vector, which is commonly denoted as the target. The current model is run with the training data sample and produces a result, which is then compared with the target, for each input vector in the training data sample. The test data sample is a data sample used to provide an unbiased evaluation of a final model fit on the training data sample. If the data in the test data sample has never been used in training, the test data sample is also called a holdout data sample.

F. Anomaly Detection Algorithms

a) Local Outlier Factor (LOF)

The Local Outlier Factor [13] is based on a concept of a neighbourhood density, where locality is given by way of k nearest neighbours, whose distance is used to estimate the density. By evaluating the local density of an object to the nearby densities of its neighbours, one can identify areas of comparable density, and factors that have a substantially decrease density than their neighbours. These are regarded to be outliers. The neighbourhood density is estimated by means of the traditional distance at which a point can be reached from its neighbours. The definition of reachability distance used in LOF is an extra measure to produce greater steady outcomes inside clusters. With this k defined, the k-distance can be introduced which is the distance of a point to its kth neighbour. If k was 3, the k-distance would be the distance of a point. The k-distance is now used to calculate the reachability distance. t to the third closest point.

$$reach - dist(a, b) = maxk - distance(b), dist(a, b) \quad (3)$$

The advantage of LOF is that, due to the local approach, LOF is able to identify outliers in a data set that would not be outliers in another area of the data set.

b) Isolation Forest

Isolation Forest [14] explicitly identifies anomalies instead of profiling normal data points. Isolation Forest, like any tree ensemble method, is built on the basis of decision trees. In these trees, partitions are created by first randomly selecting a feature and then selecting a random split value between the minimum and maximum value of the selected feature. In principle, outliers are less frequent than regular observations and are different from them in terms of values. They lie further away from the regular observations in the feature space. That is why by using such random partitioning they should be identified closer to the root of the tree. The shorter average path length, i.e., the number of edges an observation must pass in the tree going from the root to the terminal node, with fewer splits necessary. A normal point requires more partitions to be identified than an abnormal point.

c) Autoencoder

Autoencoder is an unsupervised artificial neural network that learns how to efficiently compress and encode data then learns how to reconstruct the data back from the reduced encoded representation to a representation that is as close to the original input as possible. Autoencoder, by design, reduces data dimensions by learning how to ignore the noise in the data. Autoencoders consists of 4 main parts: Encoder, Bottleneck, Decoder, Reconstruction Loss. The training then involves using back propagation in order to minimize the network's reconstruction loss.

d) Performance Evaluation

- Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

- Recall is the ratio of correctly predicted positive observations to all the observations in actual class.

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

- F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account.

$$F1Score = 2 * \frac{(Recall * Precision)}{(Recall + Precision)} \tag{6}$$

- Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations.

$$Accuracy = \frac{CorrectlyPredictedObservation}{TotalObservations} \tag{7}$$

V. EXPERIMENTAL RESULTS

The proposed method is implemented in Python 3. The correlation of the datasets is found and is displayed as a heatmap. The correlation heatmap generated from the datasets is shown in Fig – 2.

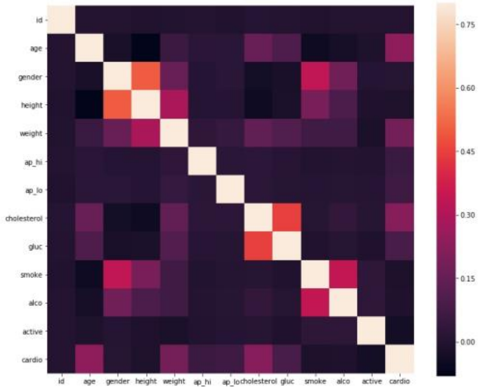
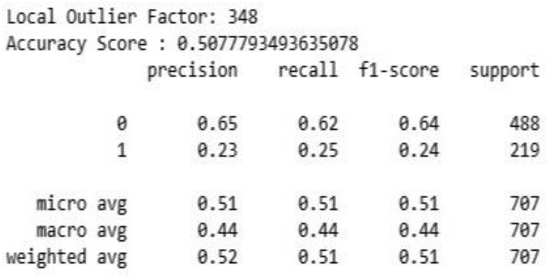


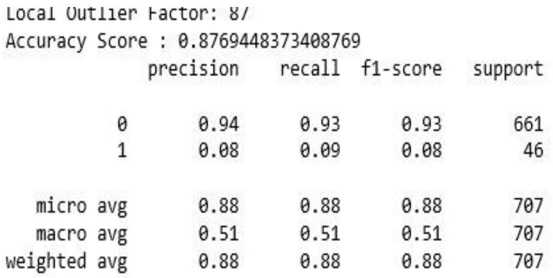
Fig -2: Correlation Matrix for Cardio Dataset

The output containing accuracy, precision, recall, F1 score for LOF algorithm implementation of the three datasets are shown in Figs – 3, 4 and 5.



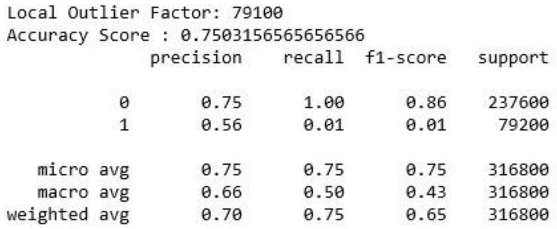
```
Local Outlier Factor: 348
Accuracy Score : 0.5077793493635078
              precision    recall  f1-score   support

           0       0.65      0.62      0.64       488
           1       0.23      0.25      0.24       219

   micro avg       0.51      0.51      0.51       707
   macro avg       0.44      0.44      0.44       707
weighted avg       0.52      0.51      0.51       707
```

Fig -3: LOF Output for Breast Cancer



```
Local Outlier Factor: 87
Accuracy Score : 0.8769448373408769
              precision    recall  f1-score   support

           0       0.94      0.93      0.93       661
           1       0.08      0.09      0.08        46

   micro avg       0.88      0.88      0.88       707
   macro avg       0.51      0.51      0.51       707
weighted avg       0.88      0.88      0.88       707
```

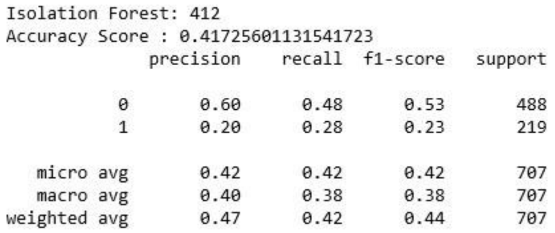
Fig -4: LOF Output for Cardio Disease



```
Local Outlier Factor: 79100
Accuracy Score : 0.7503156565656566
              precision    recall  f1-score   support

           0       0.75      1.00      0.86    237600
           1       0.56      0.01      0.01     79200

   micro avg       0.75      0.75      0.75    316800
   macro avg       0.66      0.50      0.43    316800
weighted avg       0.70      0.75      0.65    316800
```

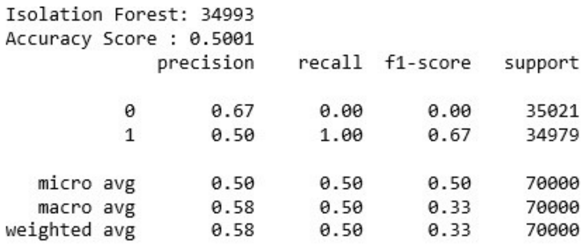
Fig -5: LOF Output for Corona Virus

The output containing accuracy, precision, recall, F1 score for Isolation Forest algorithm implementation of the three datasets are shown in Figs – 6, 7, 8.



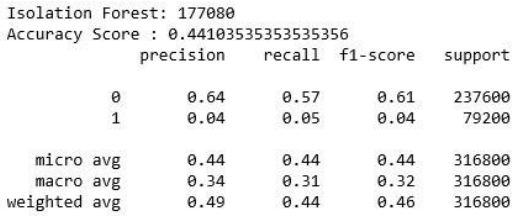
```
Isolation Forest: 412
Accuracy Score : 0.41725601131541723
              precision    recall  f1-score   support

           0       0.60      0.48      0.53       488
           1       0.20      0.28      0.23       219

   micro avg       0.42      0.42      0.42       707
   macro avg       0.40      0.38      0.38       707
weighted avg       0.47      0.42      0.44       707
```

Fig -6: Isolation Forest Output for Breast Cancer



```
Isolation Forest: 34993
Accuracy Score : 0.5001
              precision    recall  f1-score   support

           0       0.67      0.00      0.00     35021
           1       0.50      1.00      0.67     34979

   micro avg       0.50      0.50      0.50     70000
   macro avg       0.58      0.50      0.33     70000
weighted avg       0.58      0.50      0.33     70000
```

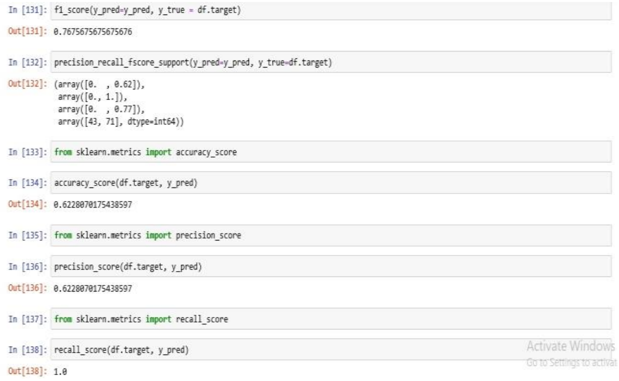
Fig -7: Isolation Forest Output for Cardio Disease



```
Isolation Forest: 177080
Accuracy Score : 0.44103535353535356
              precision    recall  f1-score   support

           0       0.64      0.57      0.61    237600
           1       0.04      0.05      0.04     79200

   micro avg       0.44      0.44      0.44    316800
   macro avg       0.34      0.31      0.32    316800
weighted avg       0.49      0.44      0.46    316800
```

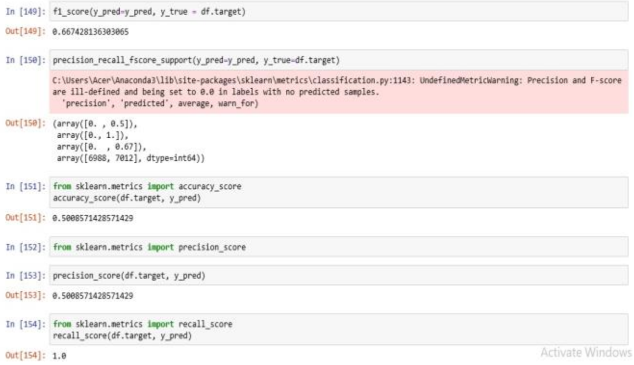
Fig -8: Isolation Forest Output for Corona Virus

The output containing accuracy, precision, recall, F1 score for Autoencoder algorithm implementation of the three datasets are shown in Figs – 9, 10 and 11.



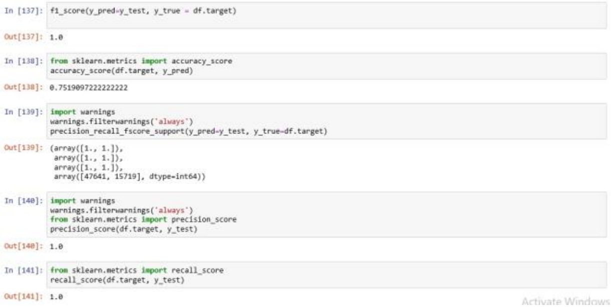
```
In [131]: f1_score(y_pred=y_pred, y_true = df.target)
Out[131]: 0.7675675675675676

In [132]: precision_recall_fscore_support(y_pred=y_pred, y_true=df.target)
Out[132]: (array([0.  , 0.62]),
           array([0., 1.]),
           array([0.  , 0.77]),
           array([43, 71], dtype=int64))

In [133]: from sklearn.metrics import accuracy_score

In [134]: accuracy_score(df.target, y_pred)
Out[134]: 0.6228070175438597

In [135]: from sklearn.metrics import precision_score

In [136]: precision_score(df.target, y_pred)
Out[136]: 0.6228070175438597

In [137]: from sklearn.metrics import recall_score

In [138]: recall_score(df.target, y_pred)
Out[138]: 1.0
```

Fig -9: Autoencoder Output for Breast Cancer



```
In [149]: f1_score(y_pred=y_pred, y_true = df.target)
Out[149]: 0.667428136303065

In [150]: precision_recall_fscore_support(y_pred=y_pred, y_true=df.target)
C:\Users\Acer\Anaconda3\lib\site-packages\sklearn\metrics\classification.py:1143: UndefinedMetricWarning: Precision and F-score
are ill-defined and being set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
Out[150]: (array([0. , 0.5]),
           array([0., 1.]),
           array([0.  , 0.67]),
           array([6988, 7012], dtype=int64))

In [151]: from sklearn.metrics import accuracy_score
          accuracy_score(df.target, y_pred)
Out[151]: 0.5008571428571429

In [152]: from sklearn.metrics import precision_score

In [153]: precision_score(df.target, y_pred)
Out[153]: 0.5008571428571429

In [154]: from sklearn.metrics import recall_score
          recall_score(df.target, y_pred)
Out[154]: 1.0
```

Fig -10: Autoencoder Output for Cardio Disease



```
In [137]: f1_score(y_pred=y_test, y_true = df.target)

Out[137]: 1.0

In [138]: from sklearn.metrics import accuracy_score
          accuracy_score(df.target, y_pred)
Out[138]: 0.7519097222222222

In [139]: import warnings
          warnings.filterwarnings('always')
          precision_recall_fscore_support(y_pred=y_test, y_true=df.target)
Out[139]: (array([1., 1.]),
           array([1., 1.]),
           array([1., 1.]),
           array([47641, 15719], dtype=int64))

In [140]: import warnings
          warnings.filterwarnings('always')
          from sklearn.metrics import precision_score
          precision_score(df.target, y_test)
Out[140]: 1.0

In [141]: from sklearn.metrics import recall_score
          recall_score(df.target, y_test)
Out[141]: 1.0
```

Fig -11: Autoencoder Output for Corona Virus

VI. PERFORMANCE EVALUATION

The three algorithms are compared in the form of a bar graph as seen in Figs – 12, 13 and 14.

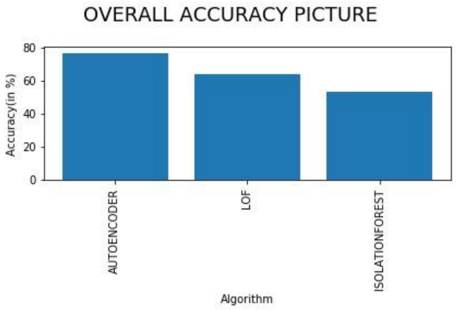


Fig -12: Algorithm Comparison for Breast Cancer

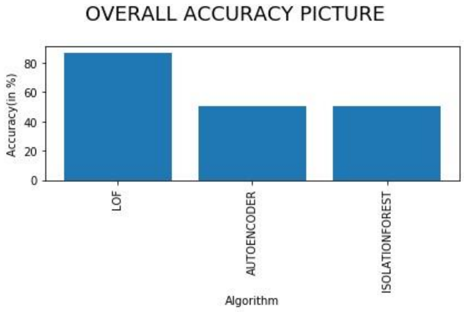


Fig -13: Algorithm Comparison for Cardio Dataset

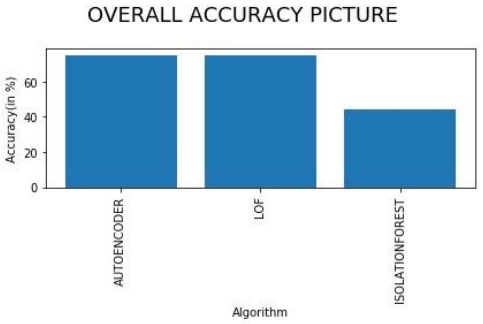


Fig -14: Algorithm Comparison for Corona Dataset

VII. CONCLUSION

The anomaly has been detected for the identified datasets and the correlation matrix has been determined. The three algorithms are implemented i.e.. the LOF, Isolation Forest and the Autoencoder algorithms are implemented for the three different datasets namely Corona disease, Breast cancer, and Heart disease dataset the accuracies are compared and the efficient algorithm is found. In the future, the double level ensemble strategy will be used to combine algorithms and produce a better accuracy rate which in turn produces a better anomaly detection algorithm that detects outliers in datasets.

REFERENCES

[1] V. Bhatt, KG Sharma, A Ram An enhanced approach for LOF in data mining, 2013.

[2] K. Xie, X. Li, X. Wang, G. Xie, J. Wen, J. Cao, D. Zhang, Fast tensor factorization for accurate internet anomaly detection, IEEE/ACM Trans. Netw. 25 (2017) 3794–3807.

[3] J. Huang, Q. Zhu, L. Yang, D.D. Cheng, Q. Wu, A novel outlier cluster detection algorithm without top-n parameter, Knowl. Based Syst. 121 (2017) 32–40.

[4] L Sun, S Versteeg, S Boztas, A Rao , Detecting anomalous user behavior using an extended isolation forest algorithm (2016)

[5] C. Zhang, J. Bi, S. Xu, E. Ramentol, G. Fan, B. Qiao, H. Fujita, Multiimbalance: An open-source software for multi-class imbalance learning, Knowl. Based Syst. (2019)

[6] S. Rayana, W. Zhong, L. Akoglu, Sequential ensemble learning for outlier detection: A bias-variance perspective, in: Proc. 16th IEEE Int. Conf. Data Mining (ICDM), 2016, pp. 1167–1172.

[7] C.C. Aggarwal, S. Sathe, Theoretical foundations and algorithms for outlier ensembles, SIGKDD Explor. Newsl. 17 (1) (2015) 24–47 [8] C. Zhang, C. Liu, X. Zhang, G. Almpanidis, An up-to-date comparison of state-of-the-art classification algorithms, Expert Syst. Appl. 82 (2017) 128–150.

[8] H. Paulheim, R. Meusel, A decomposition of the outlier detection problem into a set of supervised learning problems, Mach. Learn. 10 (2–3) (2015) 509–531.

[9] S. Rayana, L. Akoglu, An ensemble approach for event detection and characterization in dynamic graphs, in: Proc. ACM SIGKDD Workshop on Outlier Detection and Description (ODD), 2014

[10] Y Ma, P Zhang, Y Cao, L Guo, Parallel auto-encoder for efficient outlier detection, (2013)

[11] D.M. Hawkins. Identification of Outliers. Chapman and Hall (1980).

[12] V. Hodge and J. Austin. “A survey of outlier detection methodologies.” Artificial Intelligence Review, 22(2):85–126 (2004).

[13] M. M. Brueing, H. P. Kriegal, R. T. Ng and J. Sander, “LOF: Identifying density-based local outliers”, ACM SIGMOD Record, vol. 29, No. 2, pp. 93-104 (2000).

[14] Liu, Fei Tony, Ting, Kai Ming and Zhou, Zhi-Hua. “Isolation forest.” ICDM‘08. Eighth IEEE International Conference on Data Mining (2008).

[15] UCI Machine Learning Repository [http://archive.ics.uci.edu/ml].