**Indonesian Journal of Electrical Engineering and Computer Science**
Vol. 24, No. 2, November 2021, pp. 1141~1148
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v24.i2.pp1141-1148

# A simple, effective distance and density based outlier detection algorithm

**Sajidha S. A.[1], Udai Agarwal[2], Pruthviraj R. P.[3], Sparsh Agarwal[4], Nisha V. M.[5], Amit Kumar Tyagi[6]**
[1,2,3,4,5,6]School of Computer Science and Engineering, Vellore Institute of Technology, Chennai, India
[6]Centre for Advanced Data Science, Vellore Institute of Technology, Chennai, India

**Article Info**

***Article history:***

Received Dec 11, 2020
Revised Aug 19, 2021
Accepted Sep 9, 2021

***Keywords:***

Anti-neighbours
Area under curve
Density
Distances
Outliers

**ABSTRACT**

Outliers are eccentric data points with anomalous nature. Clustering with outliers has received a lot of attention in the data processing community. But, they inordinately affect the quality of the results obtained in case of popular clustering algorithms during the process of finding an optimal solution. In this work, we propose a novel method to classify the data points with grouping characteristics as either an outlier or not. We use both distance and density of a particular data point with respect to the rest of the data points for this process. Distances are used to find the points at the extremities while the densities are used to identify the data points at the sparsest spaces. Further, every data model has to take into account the aspect of generalization in order to work robustly even in out of the box situations. Hence, our approach provides a generalization aspect to the model. The accuracy of the proposed work is measured using area under curve (AUC) was found the highest for cardioto data set -AUC value-0.90 and second highest AUC value was obtained for Spambase data set -0.52 and several other datasets are used to demonstrate the usage of the model proposed.

***Corresponding Author:***

Sajidha S. A.
School of Computer Science and Engineering
Vellore Institute of Technology
Vandalur-Kelambakkam Road, Chennai-600127, India
Email: amitkrtyagi025@gmail.com

## 1. INTRODUCTION

An outlier is an observation that is strikingly far from the cluster centers. It is an eccentric value relative to the data. Normally, computed values such as average or least square lines can be dramatically affected by such values [1]. Hence, methods to detect outliers and to moderate their effects are needed. The importance of density and distance of data points while identifying the initial seed points for $k$-means for numerical data, $k$-modes for categorical data and mixed datasets using modified $k$-means algorithm, is elucidated in the work [2]-[4] in which the initial seed points were effectively identified. One of the major drawbacks of the partition based clustering algorithm is that they cannot detect the presence of outliers. Focusing on removing the outliers from the dataset further optimizing the clustering methodologies to give accurate results. The authors of [1] showed how the bivariate data represented in the form of box plots, when generalized, gives out the bag plots which in turn can be used to skim out the outliers. This aspect of the binary property was further mentioned in [5] as elucidated. In [6], the outlier detection using in-degree number (ODIN) algorithm, $k$-means and 'Outlyingness factor' have been used to remove one or more data points (outliers) and get non-overlapping clusters. In [7] proposed the approach of finding the clusters followed by the local outliers and finally looking for the global outliers. In [8], local kernel density estimation is done. In [9] explain the Gaussian uniform mixture model (GuMM) methodology which fits a cluster to identify outliers that do not fall into any of the normal clusters.

Another work in [10] proposed angle-based outlier detection (ABOD) algorithm assigns an angle-based outlier factor (ABOF) to each point in the database and returns a sorted list of points based on ABOF values. FastABOD [10] approximates ABOF based on a sample of the database. The work in [11], the distances of all the data points from their closest $k$ nearest neighbors are measured. If the sum of all the distances is greater than the threshold value $t$, then the point is considered an outlier. In the work discussed in [12] use a density-based approach for finding the outliers in the local outlier probability (LoOP) algorithm.

The work in [13] focuses on finding outliers in high dimensional data, using different subspaces of the original space (a subset of complete feature set). The work in [14] proposes a novel subspace search method that selects high contrast subspaces (HiCS) for density-based outlier ranking. The outlier scores are based on local outlier factor (LOF). The work in [15] introduced the correlation outlier probabilities (COP) method that generalizes this idea by looking for the arbitrarily oriented subspaces of highest variance and further provides an error vector for each identified outlier as a form of explanation. In the works discussed in [16], [17] studied distance based techniques used for class imbalance data.

In the work [18] compares ten different methodologies and their performances over nine real-time datasets. The methodologies like LOF, ODIN and $k$NN. are compared. In [19] the authors combine the statistical methods of mean and standard deviation (MSD) with the K-means clustering algorithm while detecting the outliers. In [20] uses cluster bound to find the suspected outlier instance. If the average distance of the suspected outlier is greater than the average distance of the neighborhood points then it can be considered as an eligible outlier.

In [21] the authors have ensemble 3 famous clustering algorithms for outlier detection from which it can seen that ensemble method outperforms individual algorithms. In [22] the authors use Neighbor entropy local outlier factor (NELOF) to reduce the time taken to scan the data set as compared to LOF. In [23] the authors have proposed a two stage thresholding method which overcomes the biasing problems in statistical methods. In our work we introduce a feasible novel simple outlier detection algorithm that uses both distance and density to identify the outliers and does not use any user defined parameters. The density of each point is estimated only once and is used in the process of detecting the outliers. Also, the proposed algorithm ensures in identifying the same set of outliers every time the algorithm is executed, which is essential in the real life problems.

## 2. RESEARCH METHOD

### 2.1. Proposed outlier detection technique

We propose a distance and density-based outlier detection methodology that can be used for identifying global and contextual outliers in data. This process is essential for various prediction and classification purposes as these outliers can cause a major deviation in the results leading to false outcomes. The proposed algorithm is independent of user input and hence provides a consistent output every time. The results are generated in a fixed number of iterations.

#### 2.1.1. Algorithm 1

**Step 1**. Initialize $L$ = Total number of data points

**Step 2**. Find the distance matrix Euclidean distance (ED) for the given points using ED.

For each data point $P_j, i := 1\ to\ L$

For each data point $P_j, j := 1\ to\ L$

Calculate $ED(i,j) = Euclidean_{Distance}(P_i, P_j)$

**Step 3.** Calculate the row sum for each row in $ED$ matrix (for finding the extremities in the dataset) and store it in $RS$

For each $i := 1\ to\ L$

$$RS(i) = \left[\sum_{j=1}^{L} ED(i,j), i\right] \quad (1)$$

Sort RS in descending order w.r.t row sum

St**ep 4**. Calculate $\bar{x}$ (i.e. threshold radius for finding the count of anti-neighboring points) calculate

$$\bar{x} = \frac{2}{|L|(|L|-1)} \sum_{i=0}^{L-1} \sum_{j=i+1}^{L} ED(i,j) \quad (2)$$

**Step 5**. Calculate the count of anti-neighbors for each data point as $CTS$ (for finding the data points in the sparsest regions)

```
For each data point $i := 1\ to\ L$
Initialize count = 0
For each $j := 1\ to\ L$
If $ED(i,j) > \bar{x}$
Increment count by 1
```

$$CTS(i) = [count, i] \tag{3}$$

Sort $CTS$ in descending order w.r.t count

**Step 6**. Find gap values in $RS$ and $CTS$ as $GD\_RS$ and $GD\_CT$ (for finding the variation in adjacent values)

For each data point $i := 1\ to\ L-1$

$$GD_{RS(i)} = [RS(i,0) - RS(i+1,0), i] \tag{4}$$

$$GD_{CT(i)} = [CTS(i,0) - CTS(i+1,0), i] \tag{5}$$

Sort $GD\_RS$ in descending order w.r.t gaps in row sums
Sort $GD\_CT$ in descending order w.r.t gaps in counts

**Step 7.** Identifying various possible outlier sets with respect to different α values
Repeat the following steps for $\alpha \coloneqq 0.01\ to\ 0.09$ (by increments of 0.01)

**Step 7.1**. Initialize $n = 1$ (where n is the number of gaps to be considered in further calculations)

```
If $\alpha \times L > 1$
$n = int\ (\alpha \times L)$   {Taking the lower bound integer of the   product}
```

**Step 7.2**. Taking the mean of first $n$ indices in $GD\_RS$ and $GD\_CT$ as $M\_RS$ and $M\_CT$. (Indices stored with the gaps in $GD\_RS$ and $GD\_CT$ arrays are equal to the number of points considered as outliers if that gap is taken as the differentiating bound for normal and outlier data points, since the indices are the initial position where the gaps occurred in the sorted $RS$ and $CTS$ arrays.)

$$M_{RS} = \frac{(\sum_{i=1}^{n} GS\_RS(i,1))}{n} \tag{6}$$

$$M_{CT} = \frac{(\sum_{i=1}^{n} GS\_CT(i,1))}{n} \tag{7}$$

**Step 7.3.** Index closest to $M\_RS$ in $GD\_RS$ ($idx\_RS$) is taken as the differentiation bound in $GD\_RS$. Index closest to $M\_CT$ in $GD\_CT$ ($idx\_CT$) is taken as the differentiating bound in $GD\_CT$.

```
Initialize $idx\_RS$ , $idx\_CT = 0$
Initialize $min\_RS = GD\_RS(0,1)$
Initialize $min\_CT = GD\_CT(0,1)$
For each gap $i := 0\ to\ n$
If $(M\_RS - GD\_RS(i,1) < \min\ \_RS)$
```

$$min\_RS = M\_RS - GD\_RS(i,1) \tag{8}$$

$$idx\_RS = GD\_RS(i,1) + 1$$

if $(M\_CT - GD\_CT(i,1) < \min\ \_CT)$

$$min\_CT = M\_CT - GD\_CT(i,1) \tag{9}$$

$$idx\_CT = GD\_CT(i,1) + 1$$

**Step 7.4**. Data points corresponding to indices 0 to idx_RS in and data points corresponding to indices 0 to $idx\_CT$ in $CT$ array are considered as $outliers\_CT$. Initialize $outliers\_RS$ as an empty array of size $idx\_RS$.

For each $i \coloneqq 0\ to\ idx\_RS$

$$outliers\_RS(i) = RS(i,1) \tag{10}$$

Initialize $outliers\_CT$ as an empty array
For each $i \coloneqq 0\ to\ idx\_CT$

$$outliers\_CT(i) = CT(i,1) \tag{11}$$

**Step 7.5**. The final list of outliers are generated by taking the union of $outliers\_RS$ and $outliers\_CT$.

$$Outliers = (outliers\_RS) \cup (outliers\_CT) \tag{12}$$

**Step 8.** Return to case with maximum AUC (i.e. Accuracy metric used in the study) along with corresponding α value.

### 2.1.2. Description of the algorithm

In the algorithm 1 mentioned above, we consider both distances (i.e. in the form of row sum) and density (i.e. in the form of counts) (i.e. Steps 3, 4, 5). These calculations are done only after the computation of a distance matrix ED that takes $O(n^2)$ time (i.e. Step 2). One of the main benefits of using the proposed algorithm is the consistency of the results. There is an unchanged 10-epoch process that is executed once for each dataset. Hence, a reasonable time is taken by the proposed model to parse over the datasets.

### 2.1.3. Illustration of the proposed outlier detection algorithm

A dataset is generated synthetically to illustrate the process is given in Table 1. The dataset consists of 15 data points with 2 attributes ($X$ $and$ $Y$). Attribute Label 0 identifies a data point as an outlier. The plot of the data points has been shown in Figure 1. Additionally, 5 data points $\{(0.6, 0.3), (0.8,0.1), (0.1), (1, 0.4), (0.04)\}$ have been added as outliers in the dataset. The plot with additional points has been shown in Figure 2. Sorted row sum values along with corresponding object indices are shown in Table 2.

Table 1. Data points considered for illustration

| Object Index | X | Y | Label |
|---|---|---|---|
| 0 | 0.388 | 0.432 | 0 |
| 1 | 0.111 | 0.101 | 0 |
| 2 | 0.305 | 0.084 | 0 |
| 3 | 0.527 | 0.644 | 0 |
| 4 | 0.555 | 0.576 | 0 |
| 5 | 0.198 | 0.067 | 0 |
| 6 | 0.833 | 0.898 | 0 |
| 7 | 0.583 | 0.779 | 0 |
| 8 | 0.805 | 0.813 | 0 |
| 9 | 0.055 | 0.050 | 0 |
| 10 | 0.416 | 0.694 | 0 |
| 11 | 0.194 | 0.423 | 0 |
| 12 | 0.166 | 0.084 | 0 |
| 13 | 0.5 | 0.661 | 0 |
| 14 | 0.527 | 0.559 | 0 |
| 15 | 0.6 | 0.3 | 0 |
| 16 | 0.8 | 0.1 | 0 |
| 17 | 0.0 | 1.0 | 0 |
| 18 | 1.0 | 0.4 | 0 |
| 19 | 0.0 | 0.4 | 0 |

Table 2. Row sums of all data points sorted in descending order

| Index | Row Sum | Object Index |
|---|---|---|
| 0 | 22.840 | 17 |
| 1 | 21.186 | 18 |
| 2 | 20.560 | 16 |
| 3 | 20.178 | 19 |
| 4 | 18.649 | 15 |
| 5 | 15.556 | 6 |
| 6 | 14.545 | 9 |
| 7 | 14.402 | 8 |
| 8 | 13.488 | 1 |
| 9 | 13.227 | 5 |
| 10 | 13.197 | 12 |
| 11 | 12.831 | 2 |
| 12 | 12.323 | 7 |
| 13 | 11.687 | 11 |
| 14 | 11.359 | 10 |
| 15 | 10.968 | 13 |
| 16 | 10.922 | 3 |
| 17 | 10.818 | 4 |
| 18 | 10.742 | 0 |
| 19 | 10.681 | 14 |

To find the threshold for considering a point as an outlier, gaps between adjacent values of row sums in Table 2 are calculated and sorted in descending order are shown in Table 3. The mean of first $n$ Density Index as shown in Table 3, for each value of α=0.1 to 0.9 is computed and finding the density index closest to the mean gives the index value considered (for Table 3 the identified index is 4 i.e. the first value). The data points before this index (i.e. index 4 in Table 2) are considered as outliers.

Find the count of anti-neighboring points for each point using radial distance ($\bar{x}$) as explained in Section 3 (i.e. in (1)). These values are sorted in descending order as shown in Table 4, we get the data points that are furthest from forming a cluster. To find a threshold similar to differentiate outliers from normal points the gaps between adjacent values of counts is computed and sorted in descending order as shown in Table 4. Sorted count gaps along with the index (i.e. position) where they occurred in Table 4 are shown in Table 5.

Identifying the appropriate index from the sorted gaps to calculate the values of $(n)$ (i.e. number of gaps to be considered from the starting of sorted gaps array) is done as mentioned in Step 7.1 For Table 5 the identified index is 4 i.e. the first value. Therefore, the data points from index $0$ $to$ $4$ inclusive as shown in

Table 4, are identified as outliers (when using counts only). Thus, the final labels of data points from 0 to 14 are 0 and from 15 to 19 are found as 1 which are the final outliers. This is shown in Figure 3.

Union of outliers identified with row sums and that with the counts is taken as the outlier set. Outlier sets are generated for each value of α. AUC scores for the dataset w.r.t each outlier set is calculated and the outlier set generating the best (i.e., highest) AUC value is returned as the final set of outliers. Table 6 shows the data points marked in orange are identified as outliers.

Table 3. Gaps corresponding to row sum index

| Index | Row Sum | Object Index |
|---|---|---|
| 3.092 | 4 | 3.092 |
| 1.633 | 0 | 1.633 |
| 1.529 | 3 | 1.529 |
| 1.011 | 5 | 1.011 |
| 0.914 | 7 | 0.914 |
| 0,635 | 12 | 0,635 |
| 0.626 | 1 | 0.626 |
| 0.508 | 11 | 0.508 |
| 0.391 | 14 | 0.391 |
| 0.381 | 2 | 0.381 |
| 0.366 | 10 | 0.366 |
| 0.327 | 13 | 0.327 |
| 0.260 | 8 | 0.260 |
| 0.142 | 6 | 0.142 |
| 0.104 | 16 | 0.104 |
| 0.076 | 7 | 0.076 |
| 0.061 | 18 | 0.061 |
| 0.045 | 15 | 0.045 |
| 0.029 | 9 | 0.029 |

Table 4. Counts related to the data points in sorted order

| Count Index | Counts | Object Index |
|---|---|---|
| 0 | 18 | 17 |
| 1 | 17 | 16 |
| 2 | 17 | 18 |
| 3 | 17 | 18 |
| 4 | 16 | 15 |
| 5 | 11 | 6 |
| 6 | 11 | 9 |
| 7 | 10 | 7 |
| 8 | 10 | 8 |
| 9 | 8 | 1 |
| 10 | 8 | 2 |
| 11 | 8 | 5 |
| 12 | 8 | 12 |
| 13 | 6 | 3 |
| 14 | 6 | 10 |
| 15 | 6 | 11 |
| 16 | 6 | 13 |
| 17 | 5 | 0 |
| 18 | 5 | 4 |
| 19 | 5 | 14 |

Table 5. Gaps corresponding to count values

| Count Gaps | Count Index |
|---|---|
| 5 | 4 |
| 2 | 8 |
| 2 | 12 |
| 1 | 0 |
| 1 | 3 |
| 1 | 6 |
| 1 | 16 |
| 0 | 1 |
| 0 | 2 |
| 0 | 5 |
| 0 | 7 |
| 0 | 9 |
| 0 | 10 |
| 0 | 11 |
| 0 | 13 |
| 0 | 14 |
| 0 | 15 |
| 0 | 17 |
| 0 | 18 |

Table 6. Final labels of data points

| Object index | X | Y | Label |
|---|---|---|---|
| 0 | 0.388 | 0.432 | 0 |
| 1 | 0.111 | 0.101 | 0 |
| 2 | 0.305 | 0.084 | 0 |
| 3 | 0.527 | 0.644 | 0 |
| 4 | 0.555 | 0.576 | 0 |
| 5 | 0.198 | 0.067 | 0 |
| 6 | 0.833 | 0.898 | 0 |
| 7 | 0.583 | 0.779 | 0 |
| 8 | 0.805 | 0.813 | 0 |
| 9 | 0.055 | 0.050 | 0 |
| 10 | 0.416 | 0.694 | 0 |
| 11 | 0.194 | 0.423 | 0 |
| 12 | 0.166 | 0.084 | 0 |
| 13 | 0.5 | 0.661 | 0 |
| 14 | 0.527 | 0.559 | 0 |
| 15 | 0.6 | 0.3 | 1 |
| 16 | 0.8 | 0.1 | 1 |
| 17 | 0.0 | 1.0 | 1 |
| 18 | 1.0 | 0.4 | 1 |
| 19 | 0.0 | 0.4 | 1 |

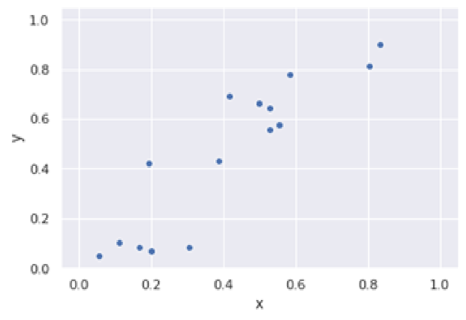

Figure 1. Synthetic dataset of 15 data points

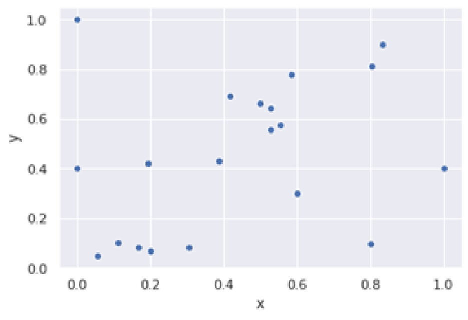

Figure 2. Dataset after adding outliers

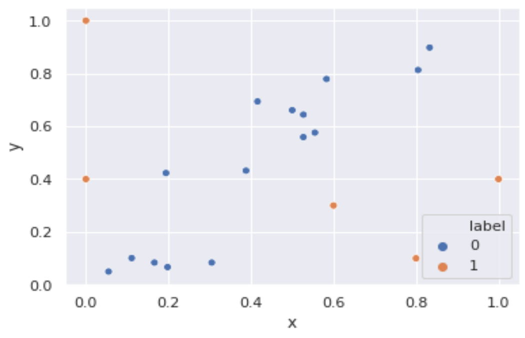

Figure 3. Synthetic dataset with identified outliers [(0(blue) non-outliers, 1(orange)-outliers]

## 3. RESULTS AND DISCUSSION

Four datasets were used from the UCI repository [24] and ELKI library [25] for the evaluation of the proposed methodology. Table 7 provides the details of the datasets, where $N$ and $O$ refer to the total number of data points and outliers, respectively. AUC is an output metric for classification problems using different threshold settings. For a dataset, the true positive rate and the false positive rate values of the respective outputs are calculated (i.e. using (13) and (14)). Further, they are plotted as an ROC curve. Higher the AUC values, better the methodology.

$$True\ Positive\ Rate = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{13}$$

$$False\ Positive\ Rate = \frac{False\ Positive}{False\ Positive + True\ Negative} \tag{14}$$

Table 7. Real time datasets used in the experiment

| Dataset | No. of data points N (Actual Outliers-O) | No. of Attributes |
|---|---|---|
| Arrhythmia | 450 (206) | 259 |
| Spambase | 4601 (1813) | 51 |
| Cardioto | 2126 (471) | 21 |
| Stamps | 340 (31) | 9 |

We have compared our proposed algorithm with other outlier detection algorithms, namely FastABOD [10], $k$NN [11], kNNW [11], ODIN [6], LOF [5], LoOP [12], COP [15], SOD [13], GuMM [9] and HiCS [14]. From the AUC results of Cardioto dataset shown in Table 8 and Figure 4 it can be observed that the proposed outlier detection algorithm outperforms FastABOD [10], kNN [11], kNNW [11], ODIN [6], LOF [5], LoOP [12], COP [15], SOD [13], GuMM [9] and HiCS [14] algorithms. The proposed outlier detection algorithm based on the AUC value for Spambase dataset outperforms ODIN [6], COP [15], GUMM [9], performs same as LOF [5] and slightly less compared to the rest of the algorithms. The AUC Values as shown in Table 8, also show that our proposed outlier detection algorithm performed better than GUMM [9] and slightly lesser than other methods for Arrhythmia dataset.

Table 8. Comparison of AUC for 4 real-time datasets (proposed algorithm results are higher than the bolded values)

| Algorithms | Arrhythmia | Spambase | Cardioto | Stamps |
|---|---|---|---|---|
| FastABOD | 0.74 | 0.01 | 0.55 | 0.01 |
| KNN [11] | 0.75 | 0.64 | 0.62 | 0.93 |
| KNNW [11] | 0.75 | 0.63 | 0.59 | 0.90 |
| ODIN [6] | 0.71 | 0.51 | 0.58 | 0.81 |
| LOF [5] | 0.74 | 0.52 | 0.59 | 0.95 |
| LoOP [12] | 0.73 | 0.54 | 0.57 | 0.67 |
| COP [15] | 0.70 | 0.50 | 0.57 | 0.69 |
| SOD [13] | 0.73 | 0.55 | 0.52 | 0.62 |
| GUMM [9] | 0.47 | 0.50 | 0.51 | 0.87 |
| HiCS [14] | 0.70 | 0.55 | 0.60 | 0.95 |
| Proposed | 0.67 | 0.52 | 0.90 | 0.04 |

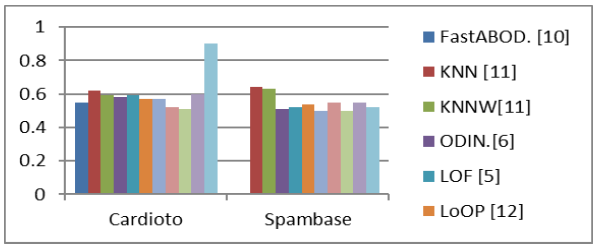

Figure 4. AUC comparison for cardioto and Spambase data set

Table 9 shows the actual outlier to normal data point ratios with the time taken for executing the algorithm for the above-mentioned datasets. From this we observe that the ratio for Cardioto, Spambase, and Arrhythmia is significantly higher as compared to the Stamps dataset. Also, Cardioto, Arrhythmia, and Spambase dataset attributes contain numerical data. This can be a contributing factor for determining the performance of an algorithm. The Spambase dataset describes the word and char frequency in an email; such digital frequencies give very few group characteristics [18] which can also contribute to the results of our proposed outlier detection algorithm.

From Table 8 the AUC values for the Stamps dataset is lesser compared to the other algorithms. As this dataset also have very few grouping characteristics [18] this also contributes to the considerably lower results from our proposed algorithm. From Table 9, we observe that the outlier to normal data points ratio is 0.1 in case of the Stamps dataset. The comparatively lower AUC scores of the proposed algorithm for the Stamps dataset can also be attributed to the low ratio of outliers to normal data points**.** Since the proposed algorithm uses distance and density for identifying outliers from normal data points, better results can be achieved with datasets having group characteristics.

One of the benefits of using the proposed algorithm is the consistency of the results. Most of the methodologies used for comparison, in Table 8 utilize user-defined variables during the process of outlier detection. Thus, it can be unequivocally proved that even though in some cases the proposed methodology does not give good AUC values, it only takes a definite number of runs without any dependency on a user-defined parameter. There is an unchanged $_{10}$-epoch process that is executed once for each dataset. Hence, a reasonable time is taken by the proposed model to parse over the datasets as shown in Table 9. It is set so to give the best differentiation between outliers and normal data points and avoiding false-positive outliers. Also, the final set of outliers is consistent in nature as our proposed algorithm does not use any user defined parameters.

Table 9. Real-time datasets and results with time of execution in milliseconds

| Dataset | No. of Data points (Total) | Actual Outliers | (Actual Outliers)/(Total - Actual Outliers) | Time (ms) |
|---|---|---|---|---|
| Arrhythmia | 450 | 206 | 0.844 | 0.1 |
| Spambase | 4601 | 1813 | 0.65 | 4.42 |
| Cardioto | 2126 | 471 | 0.284 | 0.92 |
| Stamps | 340 | 31 | 0.1 | 0.02 |

## 4. CONCLUSION

The work we have proposed is to detect outliers in clustering algorithms which is a feasible novel simple outlier detection algorithm that uses both distance and density to identify the outliers and does not use any user defined parameters. The density of each point is estimated only once and is used in the process of detecting the outliers. Also, the proposed algorithm ensures in identifying the same set of outliers every time the algorithm is executed, which is essential in the real life problems. In our proposed novel simple outlier detection algorithm, there is an unchanged 10-epoch process that is executed once for each dataset. Hence, a reasonable time is taken by the proposed model to parse over the datasets. It is set so to give the best differentiation between outliers and normal data points and avoiding false-positive outliers. It also limits the number of iterations through which the algorithm executes.

## REFERENCES

[1] D. R. Brillinger., “Data Analysis-Exploratory,” *American Journal of Political Science*, vol. 52, no. 3, pp. 705-722, 2011.

[2] S. S. Azimuddin and K. Desikan, “A simple density with distance based initial seed selection technique for K-means algorithm,” *CIT. J. Comput. information. Technology*, vol. 25, no. 4, pp. 291-300, 2017, doi: 10.20532/cit.2017.1003605.

[3] S. A. Sajidha, S. P. Chodnekar, and K. Desikan, “Initial seed selection for clustering: a distance and density based approach,” *Journal of King Saud University-Computer and Information Sciences,* 2018, doi: 10.1016/j.jksuci.2018.04.013.

[4] S. A. Sajidha, K. Desikan, and S. P. Chodnekar, “Initial seed algorithm for mixed data using modified K-means clustering algorithm,” *Arabian Journal of Science and Engineering*, vol. 45, pp. 2685-2703, 2020, doi: 10.1007/s13369-019-04121-0.

[5] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander, “LOF: Identifying density based local outliers,” *In the Proceedings of the 2000 ACM SIGMOID International Conference on Management of Data*, 2000, pp. 93-103, doi: 10.1145/335191.335388.

[6] V. Hautamäki, S. Cherednichenko, I. Kärkkäinen, T. Kinnunen, and P. Fränti, “Improving k-means by outlier removal,” *Scandinavian Conference on Image Analysis*, Springer, Berlin, Heidelberg, 2005, pp. 978-987.

[7] Y. Zhou, H. Yu and X. Cai, “A novel k-means algorithm for clustering and outlier detection,” *2nd IEEE international Conference on Future Information Technology and Management Engineering*, 2009, pp. 476-480, doi: 10.1109/FITME.2009.125.

[8] B. Tang and H. He, “A local density-based approach for outlier detection,” *Neurocomputing*, vol. 241.pp. 171-180, 2017, doi: 10.1016/j.neucom.2017.02.039.

[9] E. Schubert Zimek and H. P. Kriegel, “A survey on unsupervised outlier detection in high dimensional numerical data,” *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 5, no. 5. pp. 363-387, 2012, doi: 10.1002/sam.11161.

[10] H. P. Kriegel, M. Schubert, and A. Zimek, “Angle-based outlier detection in high dimensional data,” *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 2008*, 2008, pp. 444-452, doi: 10.1145/1401890.1401946.

[11] T. T. Dang, H. W. Ngan, and W. Liu, “Distance based k-nearest neighbors outlier detection method in large scale traffic data,” *IEEE International Conference on Digital Signal Processing*, 2015, pp. 507-510, doi: 10.1109/ICDSP.2015.7251924.

[12] H. P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, “LoOP: Local outlier probabilities,” *In Proceedings Of The 18th ACM Conference on Information and Knowledge Management*, pp. 507-510, 2009, doi: 10.1145/1645953.1646195.

[13] H. P. Kriegel, E. Schubert, and A. Zimek, “Outlier detection in axis parallel subspaces of high dimensional data,” *In Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Springer, Berlin, Heidelberg, 2009, pp. 831-838, doi: 10.1007/978-3-642-01307-2_86.

[14] F. Keller, E. Muller, and K. Bohm, “High contrast subspace for density based outlier ranking,” *28th IEEE Nternational Conference on Data Engineering*, pp. 1037-1048, 2012, doi: 10.1109/ICDE.2012.88.

[15] H. P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, “Outlier detection in the axis parallel subspaces of high dimensional data,” In *Proceedings of PAKDD*, pp. 831-838, 2009, doi: 10.1007/978-3-642-01307-2_86.

[16] G. Rekha, V. K. Reddy, and A. K. Tyagi, “Cirus-critical instances, removal based under sampling -A solution for class imbalance,” *IJHIS*, vol. 16, no. 2, pp. 55-66, 2020, doi: 10.3233/HIS-200279.

[17] G. Rekha, V. K. Reddy, and A. K. Tyagi, “An earth mover’s distance based under sampling approach for handling class- imbalanced data,” *International Journal of Intelligent Information and Database Systems*, vol. 13, no. 2/3/4, 2020, doi: 10.1504/IJIIDS.2020.109463.

[18] X. Xu, H. Liu, L. Li, and M. Yao, “A comparison of outlier detection techniques for high dimensional data,” *International Journal of Computational Intelligence Systems*, vol. 11, no. 1, pp. 652-662, 2018, doi: 10.2991/ijcis.11.1.50.

[19] Y. Wei, J. Jang-Jaccard, F. Sabrina, and T. McIntosh, "MSD-kmeans: A novel algorithm for efficient detection of global and local outliers," *Machine Learning, arXiv preprint arXiv: 1910.06588*, 2019.

[20] S. Kanjanawattana, "A novel outlier detection applied to an adaptive k-means," *International Journal of Machine Learning and Computing*, vol. 9, no. 5, pp. 569-574, 2019, doi: 10.18178/ijmlc.2019.9.5.841.

[21] A. Chatterjee Saha, S. Ghosh, N. Kumar, and R. Sarkar, "An ensemble approach to outlier detection using some conventional clustering algorithms," *Multimedia Tools and Applications*, pp. 1-25, 2020, doi: 10.1007/s11042-020-09628-5.

[22] P. Yang, D. Wang, Z. Wei, X. Du and T. Li, "An outlier detection approach based on improved self-organizing feature map clustering algorithm," *IEEE Access,* vol. 7, pp. 115914-115925, 2019, doi: 10.1109/ACCESS.2019.2922004.

[23] J. Yang, S. Rahardja and P. Fränti, "Outlier detection: how to threshold outlier scores?," In *Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud Computing,* 2019, pp. 1-6, doi: 10.1145/3371425.3371427.

[24] ELKI. [Online]. Available: https://elki-project.github.io/datasets/outlier

[25] UCI repository. [Online]. Available: https://archive.ics.uci.edu/ml/index.php