PROJECT REPORT ON
K-MEANS OUTLIER REMOVAL USING MAHALANOBIS DISTANCE ALGORITHM

Submitted in partial fulfilment of the requirements for
the award of the degree of

BACHELOR OF TECHNOLOGY

Submitted by

119003054 Hari priya R Computer Science and Engineering

Under the Guidance of

Smt Dr A Joy Christy
AP-II

SCHOOL OF COMPUTING
SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
(SASTRA DEEMED TO BE UNIVERSITY)
(A University Established under section 3 of the UGC Act, 1956)
TIRUMALAISAMUDRAM
THANJAVUR – 613 401.

April(2019)

SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
(SASTRA DEEMED TO BE UNIVERSITY)
(A University Established under section 3 of the UGC Act, 1956)
TIRUMALAISAMUDRAM, THANJAVUR – 613401

BONAFIDE CERTIFICATE

Certified that this project work entitled **"K MEANS K-MEANS OUTLIER REMOVAL USING MAHALANOBIS DISTANCE ALGORITHM"** submitted to the Shanmugha Arts, Science, Technology & Research Academy (SASTRA Deemed to be University), Thirumalaisamudram - 613401 by Haripriya R (119003054), Computer Science and Engineering in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** in their respective programme. This work is an original and independent work carried out under my guidance, during the period December 2018 –April 2019.

Smt Dr A Joy Christy

ASSOCIATE DEAN
SCHOOL OF COMPUTING

Submitted for Project Viva Voce held on_____

Examiner –I

Examiner-II

SHANMUGHA
ARTS, SCIENCE, TECHNOLOGY & RESEARCH ACADEMY
(SASTRA DEEMED TO BE UNIVERSITY)
(A University Established under section 3 of the UGC Act, 1956)
TIRUMALAISAMUDRAM, THANJAVUR – 613401

DECLARATION

I submit this project work entitled "**K MEANS K-MEANS OUTLIER REMOVAL USING MAHALANOBIS DISTANCE ALGORITHM**" to the Shanmugha Arts, Science, Technology & Research Academy (SASTRA) Deemed to be University, Tirumalaisamudram–613401, in partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY and declare that it is my original work carried out under the guidance of Prof. **Smt Dr A Joy Christy** School of Computing, SASTRA.

Name1: Hari Priya R Signature1:

Reg. No.: 119003054

Date: May 5, 2019

Place: Thanjavur, India

ACKNOWLEDGEMENTS

The satisfaction that accompanies the successful completion of this project would be incomplete without the mention of the people who made it possible, without whose constant guidance and encouragement would have made efforts go in vain. We consider ourselves privileged to express my gratitude and respect towards all those guided us through the completion of this project. We cordially thank our Registrar **Dr. G. Bhalachandran** , Dean – Planning & Development **Dr. S. Vaidhyasubramaniam** and Dean – Sponsored Research – Dr. S. Swaminathan for providing us the infrastructure to carry out the project.

We would like to express my utmost gratitude to our Dean **Dr. A. Umamakeswari,** who gave us the opportunity to complete the project.

We convey thanks to our Project guide **Smt. Dr. Joy Christy A, Assistant Professor** for providing encouragement, constant support and guidance which was of great help to complete the project successfully.

We would also thank our institution and faculty members. We also extend my heartfelt

thanks to my family, friends and well-wishers.

Team No: 18SOCUG087

SYNOPSIS

K-MEANS OUTLIER REMOVAL USING MAHALANOBIS DISTANCE ALGORITHM

119003054 - Hari Priya R - B.Tech.- Computer Science and Engineering

Outlier detection is a part of data analytics that helps user to find discrepancies in working machine by applying outlier detection algorithm on the captured data for every fixed interval. An outlier is a data point that exhibits different properties from other points that are due to some external or internal forces. These outliers can be detected by clustering the data points. To detect outliers, optimal clustering of data points is important. Problem that arises quite frequently in statistics is identification of groups or clusters of data within a population or sample. The most widely used procedure to identify clusters in a set of observations is K-Means using Euclidean distance. However, Euclidean distance is not so efficient for finding anomaly in multivariate space. To remedy this shortfall in the K-Means algorithm the Mahalanobis distance metric was used to capture the variance structure of the clusters that is followed by the application of ESVA-LCPSD algorithm to detect the outliers. If this method serves as a significant improvement over its competitors, then it will provide a useful tool for detecting rare items, events or observations which raise suspicions by differing significantly from the majority of the data.

Keywords: Mahalanobis distance, Multivariate outlier detection, ESVA- LCPSD.

Ms Dr A Joy Christy, AP-II

Table of Contents

CHAPTER 1 : INTRODUCTON

In this Information era, it is believed that information leads to power and success. Future of many companies and government organizations relies on the information what they have with them. With the improvement in the storage techniques now it is possible to collect and store a tremendous volume of information. Organizations have been collecting an immeasurable data from simple text documents to more complex information such as Medical data, Satellite data, spatial data and multimedia data. Mining of these data, using sophisticated mathematical algorithms, provides much useful information regarding the probability of future events, unusual events that might be interesting or data errors that require further investigation.

Data mining is the process of uncovering patterns and finding anomalies and relationships in large datasets that can be used to make predictions about future trends. The main purpose of data mining is extracting valuable information from available data. It is also popularly known as Knowledge Discovery in Databases (KDD). Data Mining comprises of few steps starting from preliminary raw data collections to some form of identifying new knowledge It is an iterative process and uses the following steps such as Data cleaning, Data integration, Data selection, Data transformation, Data mining, Pattern evaluation and Knowledge Representation. Once the extracted information is offered to the user, the assessment measures can be improved and further refined to get more fitting results.

One of the important applications of data mining is outlier detection. Outlier detection is the process of detecting and subsequently excluding inappropriate data from the given set of data. An outlier is a piece of data that deviates drastically from the standard norm or average of the data set. Outlier detection has two-steps viz., Clustering and detecting deviated data among the clustered sets. Therefore, the process of grouping observations into cluster is a foremost problem in analysing data sets. So far, the most widely used algorithm to identify clusters in a set of observations is K-Means. But, the main constraint of this algorithm is that it uses Euclidean distance metric, which is prone to noisy data and outliers, which in turn gives a non-spherical cluster. Also, this distance suite well only for univariate datasets. Hence, this paper introduces the technique of Mahalanobis distance (MD) to detect an observation having an unusual pattern. The MD measures the relative distance between two variables with respect to the mean of the multivariate data. These calculated distance values are used by Extreme Value Analysis algorithm ESVA-LCPSD to find outliers and thereby eliminating the need of deciding threshold value manually.

MD incorporates multivariate approach to find distance using covariance between the features of the dataset. Formula that uses distances from each observation to the central mean (M):

$$D= [(X_i - M)^T C^{-1}(X_i – M)]^{0.5}$$

Where

D = Mahalanobis distance

X_i = an object vector

M = arithmetic mean vector

C^{-1} = Inverse Covariance matrix of independent variables

T = Indicates vector should be transposed

Some important characteristics of MD that helps to provide better clustering are:

- It accounts for the fact that the variances in each direction are different.
- It accounts for the covariance between variables.
- It reduces to the familiar Euclidean distance for uncorrelated variables with unit variance.

The Mahalanobis distance accounts for the variance of each variable and the covariance between variables. Geometrically, it does this by transforming the data into standardized uncorrelated data and computing the ordinary Euclidean distance for the transformed data. In this way, it provides a way to measure distances that takes into account the scale of the data. The distance obtained is analysed using ESVA-LCPSD, an extreme value analysis based algorithm, to get the outliers with the help of algorithm itself instead of finding them with the help of threshold that is set manually.

CHAPTER 2 : PROBLEM STATEMENT

For detecting outliers in a set of observations, it is important to cluster the points accurately. Clusters are characterized by groups of data points which are in “close” proximity to one another. While it is much easier to visually detect clusters in univariate or bivariate data, the task becomes increasingly difficult as the dimensionality of the data increases. One of the largest used clustering algorithms is K-Means using Euclidean distance. Euclidean distance suffers from a scaling effect that describes a situation where the variability of one parameter masks the variability of another parameter and it happens when the measurement ranges or scales of two parameters are different, making it difficult to find the true outliers. Hence, to overcome this shortfall, the idea of using K-Means with Mahalanobis distance (MD) is put forth. The MD methodology distinguishes multivariable datagroups by a univariate distance measure, which is calculated from the correlated values of all the parameters that the dataset is dependent on. Thus, MD value is calculated using the normalized value of observations and their correlation coefficients thus making MD more efficient. After obtaining univariate distance, **an Extreme Value Analysis based algorithm**, ESVA-LCPSD is used to detect the outliers.

CHAPTER 3

LITERATURE REVIEW

Kadam and Pund [1] reviewed several approaches to detect outliers, including the cluster-based approach. Aparna and Nair [4] proposed the CHB-K-Means algorithm by using a weighted attribute matrix to detect outliers. Jiang et al. [5] proposed two initialization methods for the k-modes algorithm to choose initial cluster centers that are not outliers. Although much work has been done on outlier analysis, few of them perform clustering and detect outliers simultaneously. In this section, we focus on clustering methods with the built-in mechanism of outlier detection and give a review of those methods.

Jiang et al. [5] proposed a two-phase clustering algorithm for outlier detection. In the first phase, the *k* -means algorithm is modified to partition the data in such a way that a data point is assigned to be a new cluster centre if the data point is far away from all clusters. In the second phase, a minimum spanning tree is constructed based on the cluster centers obtained from the first phase. Clusters in small sub trees are considered as outliers. He et al. [6] introduced the concept of cluster-based local outlier and de- signed a measure, called cluster based local outlier factor (CBLOF), to identify such outliers.

Hautamäki et al. [7] proposed the ORC (Outlier Removal Clustering) algorithm to identify clusters and outliers from a dataset simultaneously. The ORC algorithm consists of two consecutive stages: the first stage is a purely k -means algorithm; the second stage iteratively removes the data points that are far away from their cluster centroids.

Zhou et al. [9] proposed a three-stage *k*- means algorithm to cluster data and detect outliers. In the first stage, the fuzzy c -means algorithm is applied to cluster the data. In the second stage, local outliers are identified and the cluster centers are recalculated. In the third stage, certain clusters are merged and global outliers are identified.

Ahmed and Naser [11] proposed the ODC (Outlier Detection and Clustering) algorithm to detect outliers. The ODC algorithm is a modified version of the k -means algorithm. In the ODC algorithm, a data point that is at least p times the average distance away from its centroid is considered as an outlier. Whang et al. [2] pro- posed the NEO- k -means (Non-exhaustive Overlapping k -means) algorithm, which is also able to identify outliers during the clustering process.

Guojun Gan and K.Michael [14] proposed *k*-Means Outlier Removal Algorithm that extends k -means algorithm to provide data clustering and outlier detection simultaneously by introducing an additional "cluster" to the algorithm to hold all outliers. Unlike most existing clustering algorithms with outlier detection, the KMOR algorithm assigns all outliers into a group naturally during the clustering process.

Jayakumar and Thomas [10] proposed an approach to detect outliers based on the Mahalanobis distance. Sachin Kumar [13] had employed a probabilistic approach for defining warning and fault threshold MD values to improve upon the traditional approaches where threshold MD values are decided by experts and demonstrates that the approach to define threshold MD value is a major improvement. Igor Melnykov [3] proposed a new algorithm to extend k-means algorithm with Mahalanobis distance to take into account of variance structure of the clusters.

CHAPTER 4 : SOFTWARE AND HARDWARE SPECIFICATIONS

This project has been implemented using Python 3.7.1 Compiler. Python provides many built-in packages for data analytics application and thereby making it easy to learn and code. Moreover, it has built-in packages for illustrating results in much clearer formats like graphs, charts, scatter-plots. Thus, python is considered for implementing this project.

Packages that are used in this project are Numpy, Sklearn, Matplotlib, mpl_toolkits. Numpy is used to handle very large datasets. Sklearn package contains cluster function that can be used to cluster the data. Matplotlib is used for graphical representation of datasets and how they are clustered using graphs, scatter-plots and others. Thus these packages should be pre-installed before implementing.

Main hardware requirement of the project is the type of Processor, Memory, RAM. 32-bit or 64-bit Operating System, having x32 or x64-based processor. Processor used is Intel(R) Core™ i3-6100U CPU @2.30GHz 2.30GHz.

CHAPTER 5 : PROPOSED MODEL

Aim of the proposed model is to find the outliers in the dataset more accurately. After clustering the points using *k*-Means with Mahalanobis distance algorithm, final step is to find the outliers based on the extreme value Analysis algorithm that takes the threshold for deciding outlier as a part of algorithm rather than fixing it manually. The steps to be followed to find the outliers are shown in the fig 5.1.

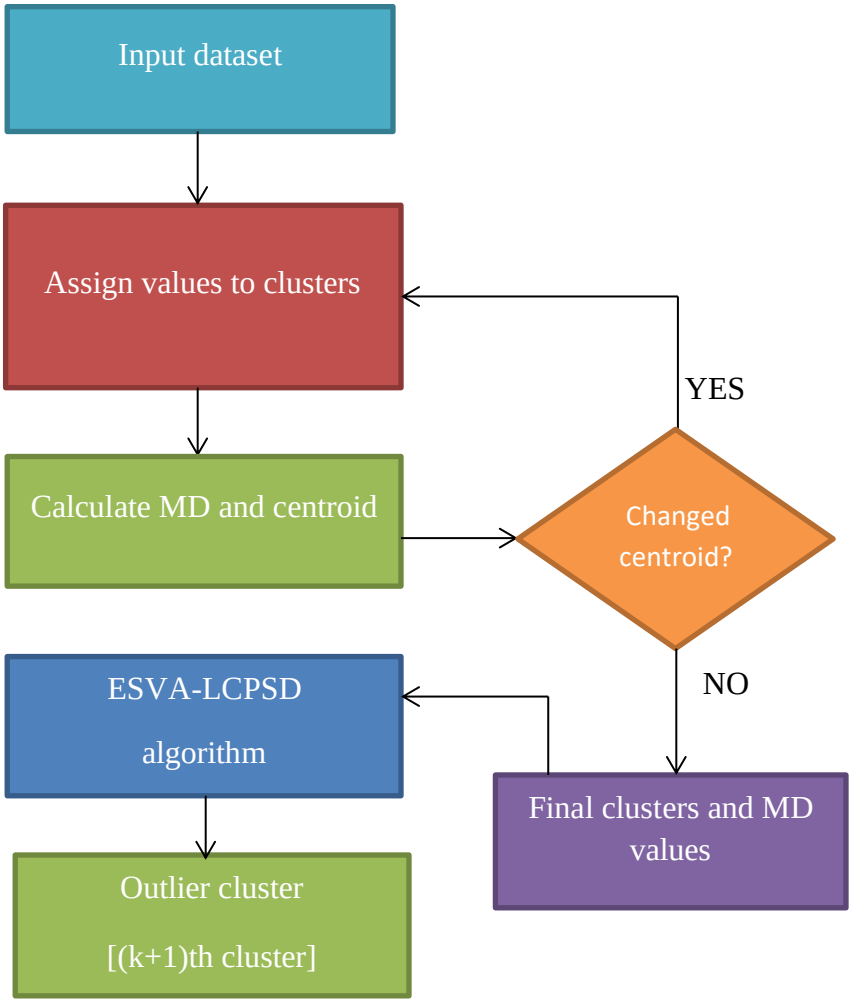


Fig 5.1 Steps to find outliers using k-means with Mahalanobis distance based clustering and ESVA-LCPSD algorithm

Final MD values are given as input for calculating threshold using Extreme value analysis algorithm, ESVA-LCPSD, to detect the outliers. Thus, this algorithm has three steps:

- K-means clustering
- Distance calculation using Mahalanobis Distance metric
- ESVA-LCPSD, Extreme value analysis to set threshold.

Brief of each step is depicted as follows. The K-Means clustering is portrayed in Fig 5.2.

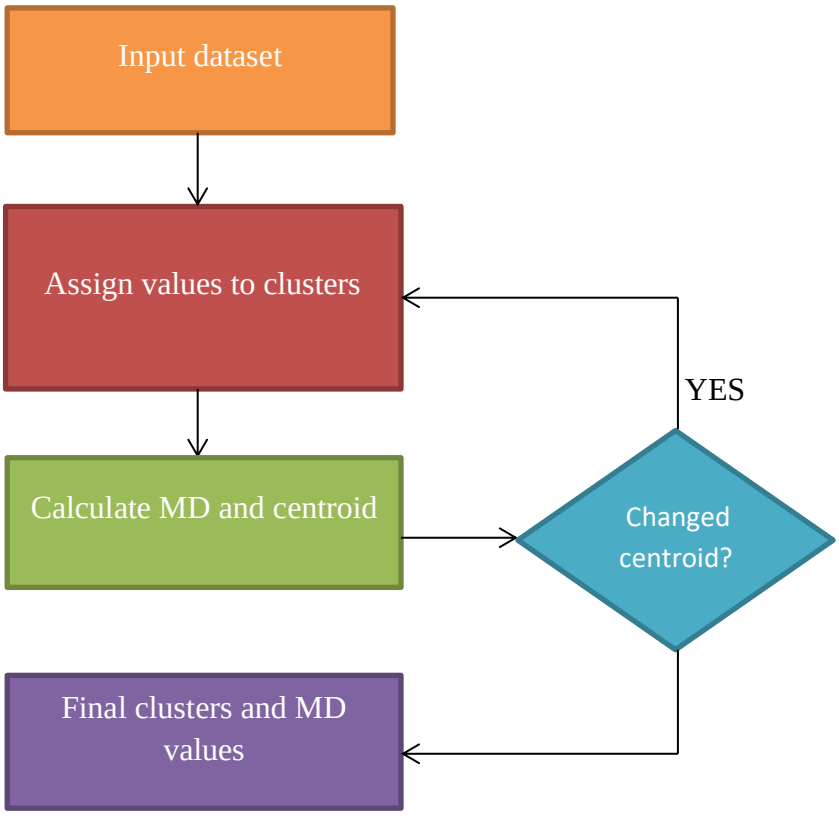


Fig 5.2 Algorithm of Modified k-means Clustering using Mahalanobis Distance

As shown in the figure K-Means algorithm takes 'n' points as input and clusters these 'n' points into each of the 'k' clusters as much as possible. If any of the point cannot be clustered, then it is taken as an outlier. As an extension of k-means algorithm, this algorithm maintains (k+1) clusters where k clusters are to hold the clustered points while $(k+1)^{th}$ cluster holds the outlier points. After, choosing the cluster centres at random, MD values and mean

for each cluster is recalculated at each step as a part of every iteration until the optimal cluster is obtained. Algorithm to calculate MD is shown in Fig 5.2.

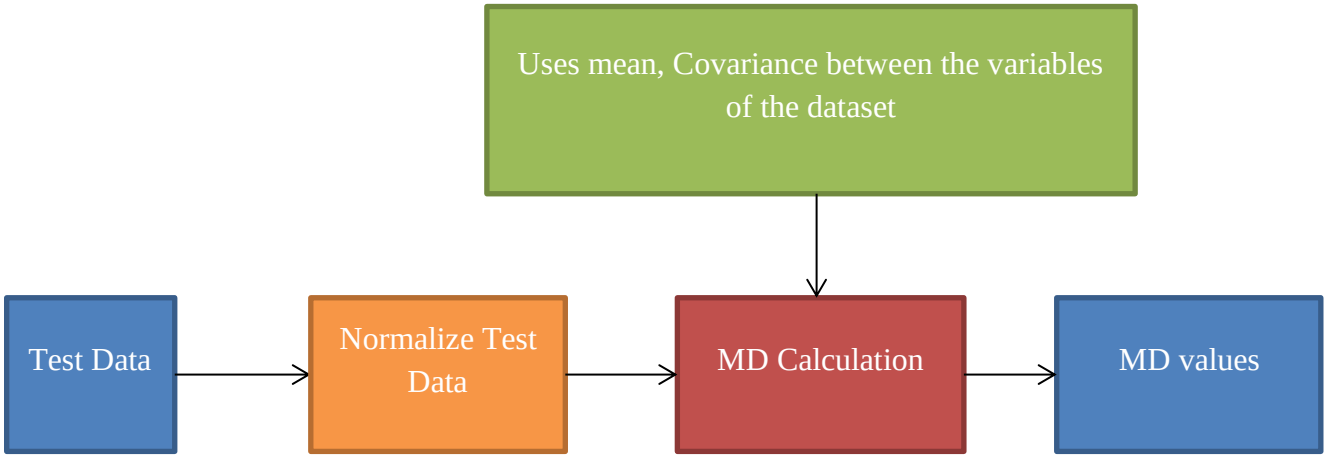


Fig 5.3 MD Calculation using test data

The Mahalanobis Distance measures the relative distance between two variables with respect to the mean of the multivariate data. Thus, the univariate distance of multivariate dataset is obtained with the Mahalanobis distance that can be used to cluster the test data using k-means clustering until the predefined number of iterations or till the difference between previous centroid and the current centroid is so marginal. Formula to calculate Mahalanobis distance is:

$$D= [(X_i - M)^T C^{-1}(X_i – M)]^{0.5}$$

Where

D = Mahalanobis distance

X_i = an object vector

M = arithmetic mean vector

C^{-1} = Inverse Covariance matrix of independent variables

T = Indicates vector should be transposed

From the MD values that is obtained, Extreme SR value analysis with single representation of data objects is implemented to identify the outliers. The ESVA-LCPSD method is used to find the threshold for the normal data and to detect the outliers. Normally, the threshold value is decided by the user manually which is a big shortfall since he/she cannot predict the threshold value always and that may in turn gives false outliers as output.

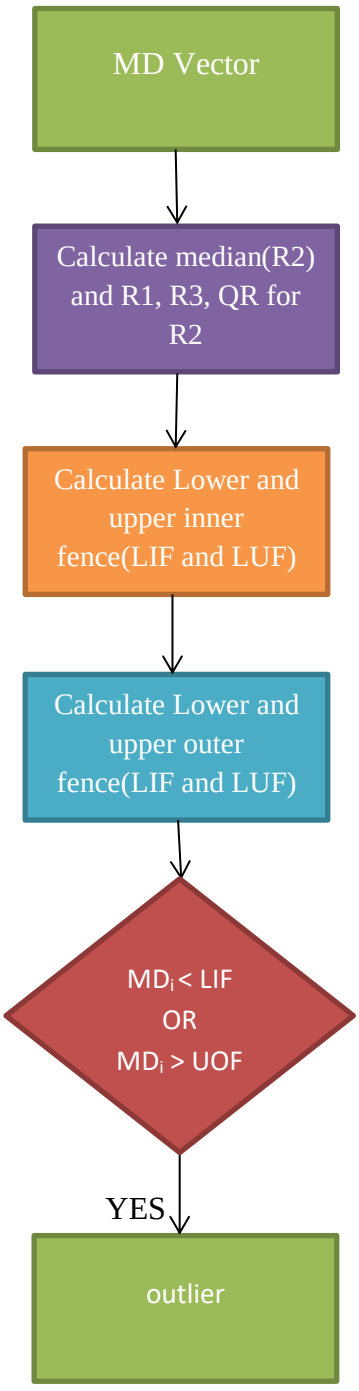


Fig 5.3 ESVA-LCPSD algorithm

CHAPTER 6 : METHODOLOGY AND APPROACH

In this chapter, algorithm of K-means outlier detection using Mahalanobis distance and ESVA-LCPSD algorithm is discussed. The dataset is obtained as the input for detecting the outliers. Modified k-means clustering using Mahalanobis distance takes dataset and k as input and outputs the deviated points. The pseudo code to follow is given below.

Algorithm 1

Input:

=> Dataset containing 'n' instances

=>k: the number of clusters

Output:

=> Outliers

Algorithm:

1. Initially, depending on the value of k, k random points are chosen as initial centroids. The distances of each data point from the centroids chosen earlier are evaluated using the Mahalanobis distance using the formula $\mathbf{D= [(X_i – X_j)^T C^{-1}(X_i – X_j)]^{0.5}}$.
2. The distance values are compared and the data point is assigned to the centroid which has shortest Mahalanobis distance (MD) value.
3. The previous steps are repeated. The process is stopped if the clusters obtained are same as that of the previous step.
4. Use Extreme Value Analysis algorithm ESVA-LCPSD **[Algorithm 2]** to find the outliers that are put into $(k+1)^{th}$ cluster.

To cluster the dataset, Mahalanobis distance is metric is used that incorporates covariance between the features of the dataset to achieve much efficient clustering in multivariate space. The formula to get the distance between the point and the cluster centre is,

$$D= [(X_i - M)^T C^{-1}(X_i – M)]^{0.5}$$

Where

D = Mahalanobis distance

X_i = an object vector

M = arithmetic mean vector

C^{-1} = Inverse Covariance matrix of independent variables

T = Indicates vector should be transposed

Final cluster's MD values are given as input to ESVA-LCPSD algorithm that uses Extreme value Analysis to find deviated data objects in the dataset.

Algorithm 2

Input:

=> MD values

Output:

=> outliers

Algorithm:

1. Calculate median R2 from MD values.
2. Compute lower and upper quartile range R1 and R3 respectively from R2.
3. Computer the range of inter-quartile RQby subtracting R3-R1.
4. Compute Lower Inner Fence(LIF) as R1-1.5*RQ.
5. Compute Upper Inner Fence(UIF) as R3+1.5*RQ.

6. Compute Lower Outer Fence(LOF) as R1-3*RQ.
7. Compute Upper Outer Fence(UOF) as R+3*RQ.
8. Outliers are the data objects goes beyond LIF and UOF of distribution.

The algorithm is approached using python to get better results. Python has built-in packages for data analytics application and thereby making it easy to learn and code. Moreover, it has built-in packages for illustrating results in much clearer formats like graphs, charts, scatter-plots. Thus, python is considered for implementing this project. Source code of the project is:

```
import numpy as np
import random
import sys
import pylab as pl
from matplotlib import pyplot as plt
from sklearn.cluster import KMeans
from mpl_toolkits.mplot3d import axes3d, Axes3D

def kMeans(X, K, maxIters = 10, plot_progress = None):

    centroids = X[np.random.choice(np.arange(len(X)), K), :]
    for i in range(maxIters):
        # Cluster Assignment step
        C = np.array([np.argmin([np.dot(x_i-y_k, x_i-y_k) for y_k in centroids]) for x_i in X])
```

```
        # Move centroids step
        centroids = [X[C == k].mean(axis = 0) for k in range(K)]
        if plot_progress != None: plot_progress(X, C, np.array(centroids))
    return np.array(centroids) , C


def normalize(original):
    dsz1=14500
    dsz2=10
    for j in range(0,dsz2):
        mx=0
        mn=1000000000
        for i in range(0,dsz1):
            if(original[i][j]>mx):
                mx=original[i][j]
            if(original[i][j]<mn):
                mn=original[i][j]
        for i in range(0,dsz1):
            original[i][j]=(original[i][j]-mn)/(mx-mn)
    return original



def MahalanobisDist(data, verbose=False):
    covariance_matrix = np.cov(data, rowvar=False)
    if is_pos_def(covariance_matrix):
        inv_covariance_matrix = np.linalg.inv(covariance_matrix)
```

```
        if is_pos_def(inv_covariance_matrix):
            vars_mean = []
            for i in range(data.shape[0]):
                vars_mean.append(list(data.mean(axis=0)))
            diff = data - vars_mean
            md = []
            for i in range(len(diff)):
                md.append(np.sqrt(diff[i].dot(inv_covariance_matrix).dot(diff[i])))

            if verbose:
                print("Covariance Matrix:\n {}\n".format(covariance_matrix))
                print("Inverse of Covariance Matrix:\n {}\n".format(inv_covariance_matrix))
                print("Variables Mean Vector:\n {}\n".format(vars_mean))
                print("Variables - Variables Mean Vector:\n {}\n".format(diff))
                print("Mahalanobis Distance:\n {}\n".format(md))
            return md
        else:
            print("Error: Inverse of Covariance Matrix is not positive definite!")
    else:
        print("Error: Covariance Matrix is not positive definite!")


def is_pos_def(A):
    if np.allclose(A, A.T):
        try:
```

```
            np.linalg.cholesky(A)
            return True
        except np.linalg.LinAlgError:
            return False
    else:
        return False


def update_U(res,Z,dsz1,dsz2,k,d_avg,U):
    outlier=0
    cur_dist=float(0)
    for i in range(0,k):
        U[i][1]=1000000000
        extreme=False
        #for j in range(0,10):
        MD = MahalanobisDist(data, verbose=True)
        om=np.sort(MD)
        print(om)
        q2=np.median(om)
        print("median",q2)
        q75, q25 = np.percentile(om, [75,25])
        print(q25,q75)
        iqr = q75 - q25
        print("iqr:" ,iqr)
        lif=q25-1*iqr
        print(lif)
```

```
        uif=q75+1*iqr
        print(uif)
        lof=q25-2*iqr
        print(lof)
        uof=q75+2*iqr
        print(uof)
        outliers = []
        for i in range(len(MD)):
            if ((MD[i] <=lof) or (MD[i]>=uof)):
                outliers.append(i)
                U[i][0]=1
            else:
                U[i][0]=0
                U[i][1]=MD[i]
    #print(U)

    print("Outliers Indices: {}\n".format(outliers))

    for ii in outliers:
        print(ii,data[ii])
    print("Total number of dataset:",len(data))
    print("Count of outliers:",len(outliers))

def kmor(res,k,gama,no,sigma,nmax,dsz1,dsz2):
    c=0
```

```
    r=0
    prev_p=float(0)
    cur_p=float(0)
    d_avg=float(0)
    s=[]
    U=[[0]*2]*dsz1
    Z=[[0.0]*10]*dsz1
    while(c<k):
        r=random.randint(50,100)%(dsz1+1)
        if(r not in s):
            s.append(r)
            c=c+1
    for i in range(r):
        Z.append(res[i])
    update_U(res,Z,dsz1,dsz2,k,1000000000,U)


def show(X, C, centroids, keep = False):
    import time
    time.sleep(0.5)
    plt.cla()
    plt.plot(X[C == 0, 0], X[C == 0, 1], '*b',
         X[C == 1, 0], X[C == 1, 1], '*r')
    plt.plot(centroids[:,0],centroids[:,1],'*m',markersize=10)
    plt.draw()
    if keep :
```

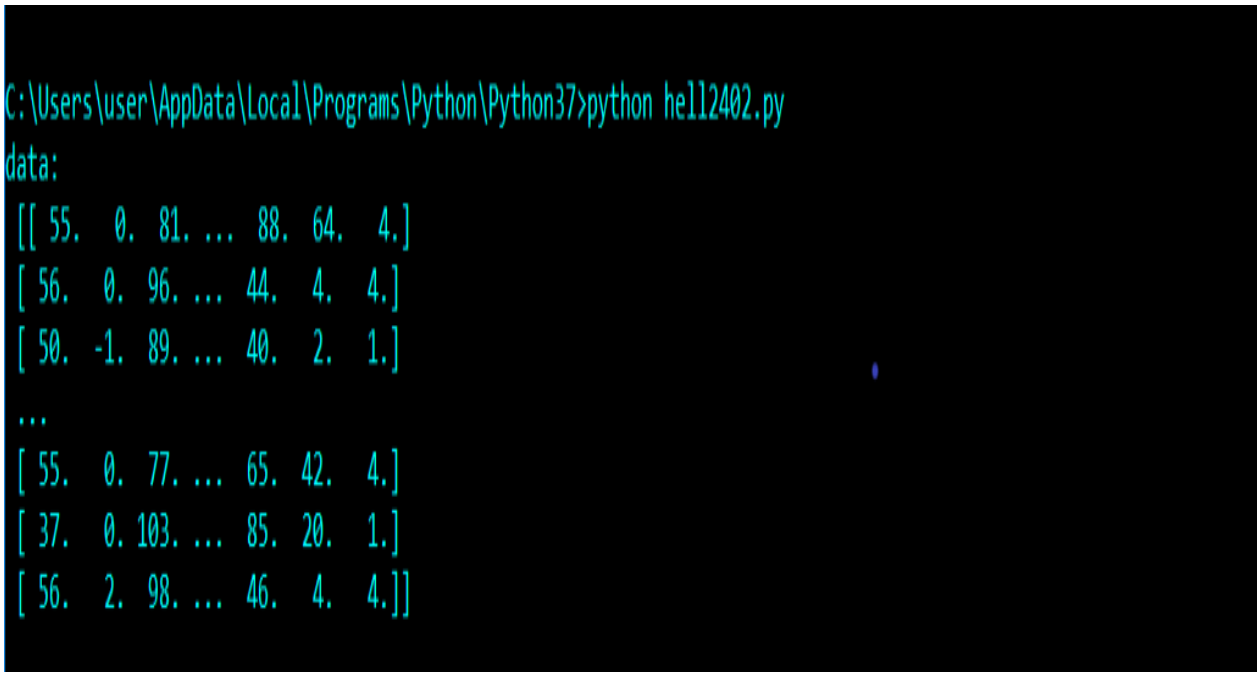
```
        plt.ioff()
        plt.show()



data = np.genfromtxt('C:\TDM-GCC-64\hp.csv',delimiter=' ')
print("data:\n {}\n".format(data))


U=[[0]*2]*14500
k=1
gama=3
nmax=100
sigma=1e-6
dsz1 =14500
dsz2=10
n0=int(dsz1*0.1)
kmor(data,k,gama,n0,sigma,nmax,dsz1,dsz2)
centroids, C = kMeans(data, K = 1, plot_progress = show)
show(data, C, centroids, True)
```

CHAPTER 7 : RESULTS AND DISCUSSION

Implementation of outlier detection using Mahalanobis distance produces much more accurate results, than K-Means Outlier Removal algorithm proposed by Guojun Gan and K.Michael [14]. Snapshots of the output got by both algorithms are embedded here for clarification.

To test the working of algorithm, shuttle dataset is used as the input and it contains 14500 test data that were captured by the sensors at regular time intervals.

FIRST PHASE:



First phase of output displays all the 14500 sets of original dataset.

SECOND PHASE:

Command Prompt - python hell2402.py

This phase gives the distance value for all 14500 datasets.

THIRD PHASE:

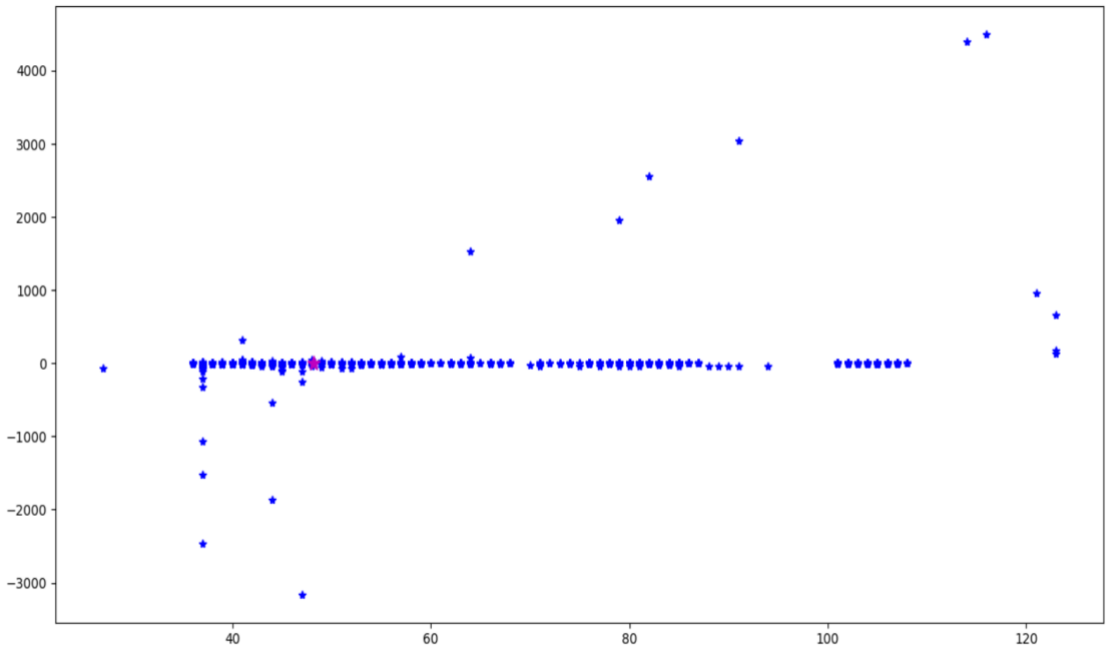
```
[  0.61076963   0.61274927   0.61688886 ...  58.21478679  93.77598032
 103.68157268]
median 2.352430296995397
R1: 1.8614581705548174
R3: 3.033105451772817
IQR: 1.1716472812179994
LIF: 0.689810889336818
UIF: 4.2047527329908165
LOF: -0.48183639188118144
UOF: 5.376400014208816
Outliers Indices: [56, 387, 529, 564, 610, 614, 695, 787, 1930, 2054, 2062, 2070, 2121, 2140, 2498, 2540, 2646,
5, 4434, 5119, 5164, 5217, 5236, 5315, 5579, 5691, 5715, 5724, 5761, 5945, 6024, 6311, 6358, 6550, 6594, 6615, 6
, 9275, 9342, 9438, 9638, 9857, 9933, 9989, 9992, 10099, 10136, 10307, 10319, 10470, 10886, 10921, 10945, 10998,
11804, 11817, 11900, 12080, 12110, 12423, 12442, 12471, 12629, 12838, 12951, 13196, 13272, 13308, 13397, 13530,
```

```
Select Command Prompt - python  hell2402.py
11087 [105.   -4. 107.   -2.   70.    0.    2.   37.   34.    5.]
11099 [105.    5. 107.    9.   72.    5.    1.   35.   34.    5.]
11160 [  44. -536.   88.    0.   44.    0.   44.   44.    0.    1.]
11264 [105.   -5. 106.    1.   72.    3.    2.   34.   32.    5.]
11313 [104.    5. 106.    6.   70. -29.    1.   36.   34.    5.]
11394 [  82. 2561.  106.   -4.   34.   -20.   24.   72.   48.    6.]
11397 [102.    4. 102.    1.   72.   18.    1.   30.   30.    5.]
11560 [123. 656. 105.    1.   36.   -4. -18.   69.   86.    5.]
11700 [105.   -5. 107.    0.   72.    0.    1.   35.   34.    5.]
11750 [ 4.5000e+01  0.0000e+00  1.1800e+02  1.7510e+03  3.1000e+02  1.5164e+04
  7.3000e+01 -1.9100e+02 -2.6400e+02  1.0000e+00]
11770 [105.   -2. 106.   -3.   72.    1.    1.   33.   32.    5.]
11802 [107.    1. 108.    0.   70.    0.    1.   38.   36.    5.]
11804 [105.    5. 106.    5.   72.    6.    1.   34.   32.    5.]
11817 [104.    0. 106.    0.   70.    0.    1.   36.   34.    5.]
11900 [ 37.   -1. 106. 258.   34.   -9.   69.   72.    4.    1.]
12080 [ 7.900e+01  0.000e+00  8.300e+01  0.000e+00  9.800e+01  8.098e+03
  4.000e+00 -1.400e+01 -1.800e+01  5.000e+00]
12110 [102.    3. 102.    0.   72.   12.    1.   30.   30.    5.]
12423 [108.    1. 109.    0.   72.    1.    1.   36.   36.    5.]
12442 [107.    0. 108.    0.   72.    8.    1.   36.   34.    5.]
12471 [104.    0. 104.   -4.   70.    0.    1.   35.   34.    5.]
12629 [105.    0. 106.    0.   70.    0.    1.   36.   34.    5.]
12838 [  85.    0.   88. -273.    0.    0.    3.   88.   84.    5.]
12951 [106.    0. 108.    0.   70.    0.    1.   38.   36.    5.]
13196 [106.   -5. 108.    2.   72.    0.    2.   35.   34.    5.]
13272 [102.    0. 102.   -6.   72.   25.    1.   30.   30.    5.]
13308 [106.    0. 108.    1.   72.    2.    1.   35.   34.    5.]
13397 [ 82.    0.   86.    0. -42. -29.    5. 130. 126.    5.]
13530 [105.    4. 106.    2.   72.    6.    1.   33.   32.    5.]
13554 [106.    0. 108.    0.   70.   -1.    1.   38.   36.    5.]
13650 [104.   -1. 106.   -3.   70.   -4.    1.   36.   34.    5.]
13839 [108.    2. 109.    0.   72.    4.    1.   36.   36.    5.]
14009 [105.   -5. 106.    2.   70.    0.    2.   36.   34.    5.]
14036 [123. 170. 105.    0.   36.   -1. -18.   69.   86.    5.]
14058 [ 37.    0.   76. 737.   20.    0.   40.   55.   16.    1.]
14112 [105.   -4. 106.    0.   70.    0.    2.   36.   34.    5.]
14331 [105.    4. 106.    4.   72.    5.    1.   34.   32.    5.]
Total number of dataset: 14500
Count of outliers: 124
```

The output of Third phase displays the number of outliers present in the given dataset that was calculated using ESVA-LCPSD algorithm.

FOURTH PHASE:

This phase prints the output in graphical form with the help of built-in matplotlib package. The dataset centre is marked using violet color and the data points are in blue. Outliers can be seen easily as they are scattered away from other data points.



In KMOR method, the number of outliers obtained was 54. While in this case, the algorithm has encountered 124 outliers. This proves that the proposed algorithm performs better than all other existing algorithms.

ALGORITHM	OUTLIERS
K-Means Outlier Removal Algorithm	54
Modified K-Means Using Mahalanobis Distance Outlier Detection Algorithm	124

CHAPTER 8 : CONCLUSION AND FUTURE WORK

Outlier contains noisy data which is researched in various domains like marketing, web mining, trend analysis and more. Various techniques are already being researched that is more generic. We surveyed on various algorithms which is more useful for the beginners. This paper compares KMOR and Modified K-means Clustering using Mahalanobis distance outlier detection algorithm to prove the efficiency of Mahalanobis distance based clustering outlier detection over other algorithms by implementing these algorithms on the real time shuttle dataset that contains 43,000 datasets roughly.

Modified K-means Clustering using Mahalanobis distance outlier detection algorithm is used to cluster and detect outliers using Mahalanobis distance. Mahalanobis distance could be used as an alternative for Euclidean distance as it employs multivariate approach. One of the major advantages is ESVA-LCPSD algorithm to find the outlier points. This is because threshold is set by the user manually in most of the cases that may not be optimal all the time. Hence, this algorithm decides the threshold using extreme value analysis automatically relieving the user from the burden of setting the threshold. Hence, this paper proves that Modified K-means clustering using Mahalanobis distance outlier detection algorithm is much efficient in case of both accuracy and executing time. It provides clusters of spherical shape and find true outliers that that KMOR algorithm has failed to provide .

One of the issues with using Mahalanobis distances is that accuracy of the distance is sensitive to initialization. Thus, for further enhancements, initialization procedure should be altered to select points close to cluster centres. Research has been going to identify points that will help finding the point close to clusters which have more neighbours to be favoured in the initialization step.

REFERENCES

1. **N.V. Kadam , M.A. Pund** , Joint approach for outlier detection, Int. J. Computer Science Applications, 6 (2) (2013) 445–448.
2. **J. Whang , I.S. Dhillon and D. Gleich** , Non-exhaustive overlapping k -means, in: "*SIAM International Conference on Data Mining (SDM)*", 2015 .
3. **Igor Melnykov, Volodymyr Melnykov,** On k-means algorithm with the use of Mahalanobis distances, Statistics & Probability Letters. 84 (2014) 88-95.
4. **K. Aparna, M.K. Nair**, Computational Intelligence in Data Mining, vol. 2, Springer, pp. 25–35.
5. **M. Jiang, S. Tseng and C. Su** , Two-phase clustering process for outliers detection, Pattern Recognition Letter. 22 (6–7) (2001) 691–700.
6. **Z. He, X. Xu and S. Deng** , Discovering cluster-based local outliers, Pattern Recognition. Letter, 24 (9–10) (2003) 1641–1650.
7. **V. Hautamaki, S. Cherednichenko , I. Kärkkäinen , T. Kinnunen , P. Fränti ,** Improving k-means by outlier removal, in: Proceedings of the *"14th Scandinavian Conference on Image Analysis, SCIA'05"*, 2005, pp. 978–987 .
8. G. Gan , Data Clustering in C ++ : An Object-Oriented Approach, Data Mining and Knowledge Discovery Series, Chapman & Hall/CRC Press, Boca Raton, FL, USA, 2011 .
9. **rednichenko , I. Kärkkäinen , T. Kinnunen , P. Fränti ,** Improving k-means by outlier removal, in: Proceedings of the "*14th Scandinavian Conference on Image Analysis*" , 2005 .
10. **S.-Y. Jiang, Q. An,** Clustering-based outlier detection method, in: "Fifth International Conference on Fuzzy Systems and Knowledge Discovery", 2008, pp. 429–433 .

11. **Y. Zhou, H. Yu, X. Cai**, A novel k-means algorithm for clustering and outlier detection, in: *"Second International Conference on Future Information Technology and Management Engineering"*, 2009, pp. 476–480 .
12. **G.S.D.S. Jayakumar, B.J. Thomas**, A new procedure of clustering based on multivariate outlier detection, J. Data Science, 11 (2013) 69–84 .
13. **M. Ahmed, A. Naser**, A novel approach for outlier detection and clustering improvement, in: "*Proceedings of the 8th IEEE Conference on Industrial Electronics and Applications (ICIEA)*", 2013, pp. 577–582 .
14. **S. Kumar, Tommy W. S. Chow and Michael Pecht**, Approach to Fault Identification for Electronic Products Using Mahalanobis Distance, "*IEEE transactions on instrumentation and measurement*". 59(2009) ,2055-2064.
15. **Guojun Gan , Michael Kwok-Po Ng**, *k* -means clustering with outlier removal ,in: Pattern Recognition Letters 90 (2017) 8–14 .