International Journal of Pure and Applied Mathematics
Volume 118 No. 7 2018, 325-331
ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue
ijpam.eu

Spatial Outlier Detection Algorithm for Trajectory Data

B.A.Sabarish
Department of Computer Science and Engineering,
Amrita School of Engineering, Coimbatore
Amrita Vishwa Vidyapeetham, India
ba_sabarish@cb.amrita.edu

R.Karthi
Department of Computer Science and Engineering,
Amrita School of Engineering, Coimbatore
Amrita Vishwa Vidyapeetham, India
r_karthi@ch.amrita.edu

T.Gireesh Kumar
TIFAC CORE in Cyber Security
Amrita School of Engineering, Coimbatore
Amrita Vishwa Vidyapeetham, India
t_gireeshkumar@cb.amrita.edu

Abstract— **Trajectories are spatiotemporal data generated by moving objects containing the spatial position of object at various time intervals. GPS devices record this information and it is possible to construct trajectory of moving objects for analysis. Outlier analysis of trajectories is done to identify abnormal activities like intrusion detection, fraud detection, fault detection and rate event detection. In this paper, Trajectory Outlier Detection algorithm using Boundary (TODB) is proposed using a boundary construction algorithm and a binary classifier. In TODB, Convex Hull algorithm is used to construct the boundary and ray casting algorithm is used to build the binary classifier. TODB is tested for its accuracy using real world data sets. Experimental results on real world data sets demonstrate that TODB correctly classify normal and outlier trajectories.**

Keywords—GPS,Trajectory; Spatial Outlier Detection; Convex hull algorithm;Classification

I. Introduction

Trajectory is a path generated by a moving object in space as a function of time. It is represented as a series of time stamped location points in chronological order. Many locations based device like GPS, mobile phones can be used to record the coordinates and time stamp of the moving object. Trajectories are generated by various moving objects such as vehicle, person, animal, hurricane, robots etc. These moving objects generate large amount of data which needs to be analysed using many data mining techniques [1].

Trajectory data mining has become an important area of research and there is growing interest to perform data analysis over trajectory data [1]. Trajectory data mining is used in many applications such as tracking of human for human behaviour analysis, customized advertising and identification of important tourist locations. Animal trajectories are used to study and identify their moving patterns and to safe guard them [3]. Hurricane trajectories are used to identify their movement and predict climatic changes over time [3]. Outlier is a pattern which is dissimilar to rest of the patterns in the data set. Outlier trajectories are pattern that show abnormal behaviour in their movement.

In this paper, we propose a boundary based algorithm for trajectory outlier detection. The trajectory outlier detection algorithm consists of the following phases: In the first phase, pre-processing of trajectories are done to remove noise and duplicates in data. In the second phase boundary for the trajectories are detected using convex hull algorithms. In the last phase trajectories are classified as normal or outlier using the ray casting algorithm.

The main contribution in this paper is outlier trajectories are identified using boundary method and their classification is done based on the constructed boundary. Other outlier detection algorithms use approaches mainly based on the density or distance based approach. The proposed approach is a simplified algorithm for outlier detection and updates of new trajectories can be easily performed to existing trajectories. The effectiveness of the proposed algorithm is demonstrated using various real world data sets. The rest of the paper is organized as follows. Section II discusses literature work related to outlier detection. Section III presents the problem statement. Section IV defines the proposed trajectory outlier detection algorithm. Section V presents the data sets used for analysis and results after evaluation. Finally, Section VI provide the conclusions from this study.

II. RELATED WORKS

A number of outlier trajectory detection algorithms are proposed and a briefly review of the methods are given below. An online anomaly trajectory detection method called onATrade where real-time analysis of taxi trajectories is done to identify irregular or illegal driving behaviour [4]. The paper proposes a two-step process: route recommendation and online detection. Route recommendation step identifies most commonly used routes from historical data. In the second step, the online detection system keeps track of driving route and warns driver of illegal driving behaviour and detects driving anomalies [4]. A taxi fraud detection system is developed by analysing a large amount of GPS data generated from taxi drivers. Interesting sites are identified from the GPS logs which are taken as source and end nodes. Travel route evidence and driving distance evidence with Dempster's rule are used to identify fraudulent taxi drivers by checking relevant historical traces of GPS data [5].

An algorithm to generate trajectory clusters for carpooling applications based on POIs around the trajectory. Optics based methodology is used to find of similarity measure for grouping of trajectories. Davies-Bouldin Index values obtained for the clustered trajectory is used as an index measure to evaluate the clustered trajectories [6]. A unified approach by formulating the (k, l)-means optimization problem whose solution consists of k clusters and outliers. An iterative algorithm is used to compute the similarity measures which can be expressed in the form of Bregman divergence.

A path outlier detection algorithm to find the outliers in transportation networks has been developed [8]. Spatial-region and perimeter based metric with path segmentation approach is used to capture local feature differences and similarity in data. Partition and detect framework is proposed where the trajectory is split into line segments and algorithm detects outlier in sub trajectories [3]. Distance based and density based approaches are used for detection. A two level partitioning method is used to improve speed of detection using the principle of pruning. A novel methodology to identify outlier considering both internal and external properties of the trajectory has been proposed. Trajectory structures are represented using speed, direction, angle and location features [9]. Structural similarity includes distance measure between all the four features of the trajectory. Using the distance measure outliers are classified.

A new framework is proposed named ROAM (Rule- and Motif-based Anomaly Detection in Moving Objects). In ROAM, object trajectories are expressed using discrete pattern fragments called motifs. Features are extracted from the fragments to form a hierarchical feature space, which provide a multi-resolution view of the data [10]. A rule based classifier is designed to which explores the feature space and classify the trajectories. An algorithm where an object in a data set is an outlier if a least fraction of the objects in dataset lies at a distance greater than from other objects is proposed [11]. This approach depends on the global distribution of data points and is affected by data of different density distributions. An outlier detection algorithm based on local outlier factor (LOF) of each object, which depends on the local density of its neighbourhood. Data points with a high LOF value are detected as outliers and calculation of LOF is computationally expensive [12].

III. PROBLEM STATEMENT

This paper focus on trajectory outlier detection algorithm using boundary analysis and classification of trajectories. Trajectory TR is defined as a sequence of chronological points. $TR = [p_1, p_2, .. p_n]$, where n is the number of points in the trajectory which varies for each trajectory. Each point p is a d-dimensional point representing the location and time information. The set T represents a set of trajectories $T = [TR_1, TR_2, ..., TR_m]$, where TR_i is a trajectory and m is the total number of trajectories that are used for analysis. The trajectory set T is given as input to the convex hull algorithm which returns the representative points of the convex hull. These representative points form the boundary for outlier detection.

The Boundary is represented as a set of points $B = [p_1, p_2, .. p_b]$, where b is the total number of boundary points generated by convex hull algorithm. Classifier is based on the principle of ray casting algorithm, which test if the given trajectory TR lies inside or outside the boundary B.

IV. TRAJECTORY OUTLIER DETECTION ALGORITHM

In this section, we discuss the proposed trajectory outlier detection algorithm. The algorithm consists of three phases: In the Pre-processing phase, trajectories are pre-processed to remove noise and duplicate values. The pre-processed trajectories are stored and given as input to the second phase of the algorithm. In the second phase, boundary for the trajectories are detected using convex hull algorithm. In the last phase trajectories are classified as normal or outlier using the ray casting algorithm. The proposed algorithm Trajectory Outlier Detection Using Boundary (TODB) is given below:

Algorithm TODB *(Trajectory Outlier Detection Using Boundary)*

```
Input: A Set of Trajectories $T = [TR_1, TR_2, ...., TR_m]$, where m is total number of trajectories,
Test trajectory $TR = [p_1, p_2, .. p_n]$, where n is the number of points in the trajectory which is to be tested for normal or outlier trajectory.
Output: Test Trajectory $TR$ is normal or outlier trajectory.
Step 1: Pre-processing
    For each $TR_i \in T$ do check for noise and timing duplicates and process the trajectories.
Step 2: Construction of Boundary
    For each $TR_i \in T$ do
    Construct the boundary B using the Graham convex hull algorithm
    Boundary $B = [p_1, p_2, .. p_b]$, where b is the total number of boundary points generated by convex hull algorithm.
Step3: Outlier classifier
    Test for new trajectory TR using binary classifier and classify TR as normal or outlier trajectory.
```

A. Data pre-processing

Pre-processing is an important step in trajectory analysis. Spatial and temporal information are recorded to show the trajectory path generated by the user. Huge amount of data is generated which contain redundant and noisy information which are recorded and logged every second by GPS devices. In pre-processing phase, collecting and cleaning a set of trajectories are done offline. It will determine trajectories that are used for building the classifier and trajectories to be used for testing. Transformed and pre-processed trajectories is used for outlier analysis.

B. Convex Hull Algorithm

Given a set of points in a plane, the convex hull is the smallest convex polygon that encloses all the points of the set [13]. Convex Hull is used in many applications convex relaxations, region cropping in images, shortest path finding for robots, tracking of the spatial extend of a disease outbreak, animal movement boundary tracking, area of rainfall using sensors etc

In this paper for trajectory outlier detection, trajectories are given as input to the convex hull algorithm. Convex hull algorithm proposed by Graham is used for convex hull generation [14]. The Convex hull algorithm for boundary generation is outlined and given below. The output of the algorithm is a smallest convex polygon which represents an optimal boundary enclosing all input trajectories. These boundary points are identified and periodically updated when new trajectories are added to the data set. The boundary identified by the convex hull algorithm is used for classification of trajectories as normal and outlier trajectories. The proposed algorithm convex hull algorithm is given below:

Algorithm Convex Hull:

```
Input: A Set of Trajectories $T = [TR_1, TR_2, ...., TR_m]$, where m is total number of trajectories,
Output: Boundary $B = [p_1, p_2, .. p_b]$, where b is the total number of boundary points generated by convex hull algorithm.
Step1:
    Step 1.1: For each trajectory $TR_i \in T$, sort all point in $TR = [p_1, p_2, .. p_n]$, find the point with the lowest y coordinate value from $T$. Let k be the total points in $T$.
    Step 1.2: If there are two points with the same y value, then the point with the smaller x is value is considered.
    Step 1.3: Put the selected point in an array Point at the first position Point [0].
Step2:
    Step 2.1: Consider the remaining k-1 points and sort them by polar angle in counter clockwise order around Point[0]. If polar angle of two points is same, then put the nearest point first. The sorted points are in the array Point[0] ... Point[k].
Step 3:
    $top[S] \leftarrow 0$
    $top[S] \leftarrow 0$
    Step3.1: Create an empty stack S and set
        Push (S, Point [0])
        Push (S, Point [1])
        Push (S, Point [2])
    $i \leftarrow 3 \cdots k$ Step 3.2: For $i \leftarrow 3 \cdots k$
        While angle formed by points Next_to_Top[S], Top[s]and Point[i] make a non-left turn then Pop[S]
        Push (S, Point[i])
Step 4: Copy $B \leftarrow S$, where B is the boundary points of the convex hull.
```

C. Classification of Trajectories

In this section, we propose a classification algorithm for trajectory outlier detection. Trajectories are classified as normal or intruders based on the movement pattern within the boundary as generated by the convex hull algorithm. Convex hull algorithm constructs a convex polygon as boundary enclosing all valid input trajectories[2]. Classification algorithm uses the principle of ray casting algorithm and checks if the given trajectory is an outlier or not. Input to the classification algorithm is the convex hull B and trajectory T to be tested as normal or outlier. Ray casting algorithm checks if each point of the trajectory T lies within the boundary B or not. If the trajectory points of T lie outside the boundary B then the trajectory T is classified as outlier, else it is classified as a normal trajectory.

Algorithm Outlier_Classifier

Input: Test trajectory $T = [TR_1, TR_2, ..., TR_m]$, where n is the number of points in the trajectory, Boundary $B = [p_1, p_2, .. p_b]$, where b is the total number of boundary points generated by convex hull algorithm.

Output: Test Trajectory TR is normal or outlier trajectory.

```
$m \leftarrow 0, c \leftarrow 0$
Step 1: Initialize $m \leftarrow 0, c \leftarrow 0$
Step2:
        For each point $p_i \in TR$
        {
        Draw a line from point $p_i$ in the trajectory to
        infinity
        Count c the number of times the line intersects with
        the boundary edges B.
        If c is odd then
         {
                Point $p_i$ lies inside the boundary
                Increment m
        }
        else
        {
                If ( $p_i$ is a vertex of polygon)
                {
                        Point $p_i$ lies inside the boundary
                        Increment m
                }
                else
                {
                        Point $p_i$ lies outside the
                boundary.
                        Break
}
        }
        }
Step3:
        if $(m = n)$ then
                classify the trajectory $TR$ as normal.
        else
                classify the trajectory $TR$ as intruder.
```

Many of the proposed algorithms in literature for outlier analysis used the properties of trajectory like speed, velocity, angle, sub trajectories *etc.* for analysis. The advantage of the proposed algorithm is we make use of the boundary principle, which captures the close trajectories with similar properties implicitly.

V. DATASET

In this section, we implement TODB algorithm and investigate its accuracy of different datasets with labelled trajectories. The main objective is to investigate how accurately TODB discriminates abnormal trajectories from normal trajectories. To evaluate the performance of TODB algorithm, two data sets are considered. The datasets used in this study are outlined below:

Dataset1: The GPS trajectories dataset. This data is available in UCI repository (https://archive.ics.uci.edu/ml/datasets /GPS+Trajectories). This dataset contains 163 trajectories, with 15 features (trajectory id, speed, distance, rating, rating-bus, rating-weather etc). Each trajectory point is captured as spatiotemporal information with id, latitude, longitude and date-time information. Offline pre-processing of trajectories is done to remove noisy trajectories including redundant and short trajectories (trajectories having less than 10 GPS points). After processing 93 trajectories are selected for the study.

Dataset2: The Coimbatore-Amrita Dataset: This dataset was created by the authors and contains of 43 trajectories. These trajectories are the path traversed by faculty and students of Amrita University in Coimbatore city. The users are requested to draw their commonly used routes in google map and trajectories are generated from them. The steps involved for trajectory generation are: 1) User select the route (trajectory) from source to destination within Coimbatore city. 2) Keyhole Markup Language (KML) file for the route is generated from google map 3) KML file is pre-processed and converted into a text file to get the trajectory information.

VI. EXPERIMENTAL RESULTS AND DISCUSSION

A. GPS Dataset

To test the TODB algorithm, we first use GPS dataset from UCI repository. The boundary is constructed using the convex hull algorithm for all the selected trajectories. GPS dataset contain 93 trajectories which are used for training and testing. 90% of the trajectories are used for training and 10 % of the trajectories are used for testing. The experiment is repeated for 5 different runs by randomly sampling trajectories which are used for testing and training. GPS Visualizer is an online utility that creates maps and profiles from geographic data. This tool is used in this study for visualizing trajectory data.

Figure 1 shows a sample output where the boundary generated by convex hull algorithm is represented by a red boundary. For testing, 7 trajectories are chosen which is shown in figure 1. The tested 7 trajectories are normal trajectories and they all lie inside the boundary of convex hull. TODB algorithm classify all these 7 trajectories correctly as normal trajectories. Figure 2 shows a sample output where the boundary generated by convex hull algorithm is represented by a red boundary. For testing, 7 trajectories are chosen which is shown in figure 2.

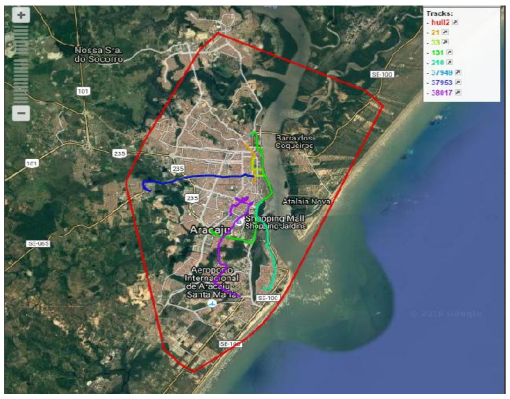

Fig 1: Trajectories from GPS data set (Normal Trajectories)

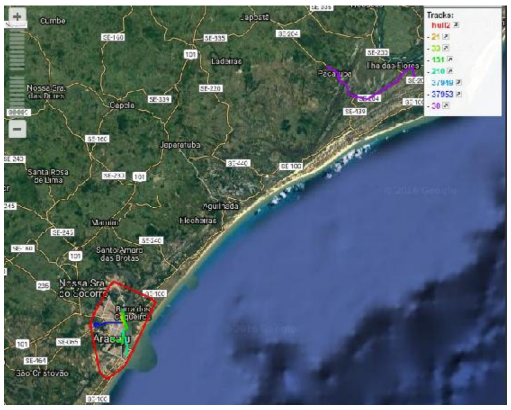

Fig 2: Trajectories from GPS data set
(Normal and outlier trajectories)

For testing, 6 trajectories are taken as normal trajectories and 1 trajectories is considered as outliers. From Figure 2, 6 normal trajectories lie inside the boundary, and are identified by TODB as normal trajectories. Only 1 outlier trajectory is present and is identified as outlier by TODB as the trajectories lie fully outside the boundary. TODB algorithm classifies all these 7 trajectories correctly as normal or outlier.

Using GPS dataset, the experiment is repeated for 5 different runs by randomly sampling trajectories. During these runs, the boundary generated by convex hull is same for all the runs. For testing, 35 trajectories are considered, and TODB classify all these trajectories correctly.

B. *Coimbatore Amrita Dataset*

Coimbatore-Amrita Dataset contains 43 trajectories which are used for analysis. Trajectories are generated by simulation using google map. 90% of the trajectories are used for training and 10% of the trajectories are used for testing. Figure 3 shows a sample run where the boundary generated by convex hull algorithm is represented by a red boundary. For testing 7 trajectories which are normal trajectories are considered and TODB algorithm classify all these trajectories correctly.

Figure 4 shows a sample output where the boundary generated by convex hull algorithm is shown by red boundary. For testing, 7 tested trajectories which are normal trajectories and outliers are considered and are show in the figure 4. For testing, 5 trajectories are taken as normal trajectories and 2 trajectories are considered as outliers. From Figure 4, 5 normal trajectories lie inside the boundary, and are identified by TODB as normal trajectories. The 2 outlier trajectories show in blue and violet are identified as outlier trajectories as these trajectories lie partially or fully outside the boundary. The proposed TODB algorithm classify all 7 tested trajectories are normal or outliers correctly.

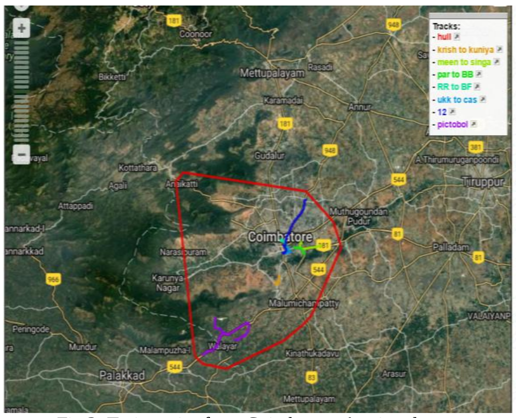

Fig 3: Trajectories from Coimbatore-Amrita data set
(Normal trajectories)

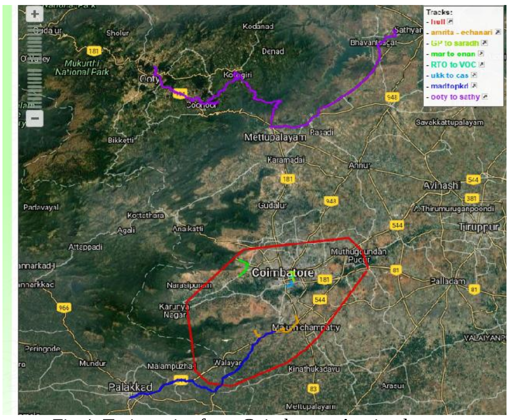

Fig 4: Trajectories from Coimbatore-Amrita data set
(Normal and outlier trajectories)

Figure 5 shows a sample output and the boundary generated by convex hull algorithm represented by red boundary. Here 6 tested trajectories which are normal and 1 outlier trajectory are considered for testing. From Figure 5, 4 normal trajectories lie inside the boundary, and are identified by TODB as normal trajectories correctly. Only 1 outlier trajectories exist in the 7 tested trajectories which is represented by violet colour. The outlier trajectory lies outside the boundary and is classified by TODB algorithm correctly. Two trajectories represented in green and orange are normal trajectories which are classified as outliers erroneously by the classifier.

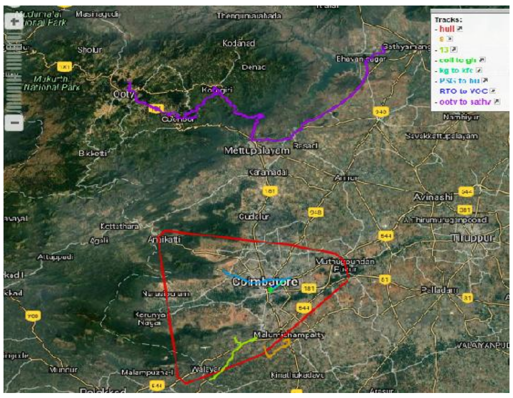

Fig 5: Trajectories from Coimbatore-Amrita data set
(normal and outlier trajectories)

Using Coimbatore_Amrita dataset, the experiment is repeated for 5 different runs by randomly sampling trajectories. During these runs, 35 trajectories are tested and only 2 normal trajectories are identified wrongly as outliers.

VII. CONCLUSION

In this paper, a new algorithm TODB is proposed for trajectory outlier detection using the principle of boundary construction and classification. The advantage is the simplicity of the algorithm and it also captures outlier easily. The algorithm can be updated for new trajectories by changing the boundary without modifying the classifier. In this work, spatial information of trajectories is only considered for classification. The algorithm has application towards surveillance analysis, animal movement tracking, human tracking and sensor based boundary analysis.

REFERENCES

[1]. Yu Zheng ,Trajectory Data Mining: An Overview. ACM Transaction on. Intelligent System and Technology. Vol. 6, Issue 3, Article 29, 2015.
[2]. D. M. Menon and Dr. Radhika N., “Anomaly detection in smart grid traffic data for home area network”, in 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT), 2016.
[3]. Lee, Jae-Gil, Jiawei Han, and Xiaolei Li. "Trajectory outlier detection: A partition-and-detect framework." In Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on, pp. 140-149. IEEE, 2008.
[4]. Zhou, Zuojian, Wanchun Dou, Guochao Jia, Chunhua Hu, Xiaolong Xu, Xiaotong Wu, and Jingui Pan. "A method for real-time trajectory monitoring to improve taxi service using GPS big data." Information & Management 53, no. 8 pp.964-977, 2016
[5]. Ge, Yong, Hui Xiong, Chuanren Liu, and Zhi-Hua Zhou. "A taxi driving fraud detection system." In Data Mining (ICDM), 2011 IEEE 11th International Conference on, pp. 181-190. IEEE, 2011
[6]. Cruz, Michael O., Hendrik Macedo, and Adolfo Guimaraes. "Grouping similar trajectories for carpooling purposes." In Intelligent Systems (BRACIS), 2015 Brazilian Conference on, pp. 234-239. IEEE, 2015.
[7]. Chawla, Sanjay, and Aristides Gionis. "k-means–: A unified approach to clustering and outlier detection." In Proceedings of the 2013 SIAM International Conference on Data Mining, pp. 189-197. Society for Industrial and Applied Mathematics, 2013.
[8]. Lu, Qifeng, Feng Chen, and Kathleen Hancock. "On path anomaly detection in a large transportation network." Computers, Environment and Urban Systems 33, no. 6, pp.: 448-462, 2009.
[9]. Yuan, Guan, Shixiong Xia, Lei Zhang, Yong Zhou, and Cheng Ji. "Trajectory outlier detection algorithm based on structural features." Journal of Computational Information Systems 7, no. 11 pp: 4137-4144, 2011
[10]. Li, Xiaolei, Jiawei Han, Sangkyum Kim, and Hector Gonzalez. "Roam: Rule-and motif-based anomaly detection in massive moving object data sets." In Proceedings of the 2007 SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, pp. 273-284, 2007.
[11]. Knorr, Edwin M., Raymond T. Ng, and Vladimir Tucakov. "Distance-based outliers: algorithms and applications." The VLDB Journal—The International Journal on Very Large Data Bases 8, no. 3-4 ,pp.237-253, 2000
[12]. Breunig, Markus M., Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. "LOF: identifying density-based local outliers." In ACM sigmod record, vol. 29, no. 2, pp. 93-104. ACM, 2000.
[13]. Shimrat, Moshe. "Algorithm 112: position of point relative to polygon." Communications of the ACM 5, no. 8 ,pp.434 ,1962.
[14]. Graham, Ronald L. "An efficient algorith for determining the convex hull of a finite planar set." Information processing letters 1, no. 4,pp: 132-133, 1972.