Outlier Detection in Text Data: An Unsupervised Method Based On Text Similarity and Density Peak

Mahnaz Taleb Sereshki
Imam Khomeini International University Faculty of Technical and Engineering

M_talebsereshki@edu.ikiu.ac.ir*Corresponding Author*
ORCiD: https://orcid.org/0000-0002-8321-7381

Morteza Mohammadi Zanjireh
Imam Khomeini International University Faculty of Technical and Engineering

DOI:
10.21203/rs.2.24181/v1

SUBJECT AREAS
Information Retrieval and Management

KEYWORDS
Outlier, Text Data, Outlier Detection, Preprocess

Abstract

Outlier detection is problematic in the text data processing. In this paper, a novel method for detecting outliers in text data is proposed. To recognize the outliers, we concentrate on the similarity between every two documents and the density of similar documents. Afterward, an algorithm, which is based on these two elements, is written. We conducted a series of experiments on real datasets with different volumes to evaluate the performance of our proposed method. Experimental results obviously show that our method outperforms than a few previous methods especially when the volume of a dataset is medium. Furthermore, they represent that, we have no performance in one of the powerful outlier detection algorithms due to small datasets.

1. Introduction

The accurate processing and analysis of data change into the concern of many data specialists. On the other hand, generating data has increased significantly, and data analyzers have confronted the amount of dirty data, therefore cleaning data can help them to approach their goal. Outlier detection is one of the groups of cleaning data. In other words, outlier detection is defined as the algorithms, which find abnormal data[1], and the abnormal data is defined as data that does not follow the main patterns of the dataset[2][3]. In certain circumstances, the goal of systems is finding outliers and studying why these data are in the systems. The importance of detecting outliers is evident in analysis systems of credit fraud prevention and network intrusion detection[4].

Detecting outliers in unstructured and semi-structured data, which text data are a member of, has great value because the processing of these kinds of data is complex, and finding outliers can help data scientists to increase the accuracy of the analyses. Although, outliers have multiple definitions in text data and detecting outliers is not easy.

Numerous researchers worked on outlier detection in text documents. Hence, different aspects of the text properties are considered[5]. For instance, a few of them changed the text into numbers and used algorithms, which are appropriate for numeric data[6]. Also, several researchers chose limited words such as the title of documents, for detecting outliers and finding the pattern of datasets[7]. We can see a number of studies about dynamic datasets such as social media and finding abnormal data in these cases[8][9] and also it is said about the importance of knowledge before analyzing[1].

With studying previous works, we can see certain limitations for implementation, such as requiring prior knowledge, a lack of text-specific algorithms, and limitation in the number of words processed. The intention of this paper is presenting a method for detecting outliers in the text data that can be flexible in a different situation, appropriate for text features, and without limitation in implementation. The rest of this paper is organized as follows. Section 2 reviews the groups of outlier detection algorithms and two powerful outlier detection (named LOF and DBSCAN), which are used in our experiments. Section 3 is representing and analyzing our proposed method. We discuss the conditions of our experiments in Section 4. Section 5 demonstrates our results in different conditions. Finally, we can see the conclusion in Section 6.

2. Related Work

Outlier detection algorithms generally are categorized into five groups. The first one is related to the algorithms based on classification. In this case, data should have labels and are used for training classification algorithms[10]. The next one is clustering-based outlier detection. In these algorithms, those data, which are not similar to patterns of the general data, are shown as outliers[10][11]. The third group is the algorithms, which consider the nearest neighbor. In these algorithms, similarity to the nearest neighbor is important. Those data that are not similar to their neighbors are considered abnormal data[10]. The fourth group uses probabilities and statistic equations for computing the probability of being data in an area, if data exists in an area with less probability, this data is outlier[12][10][13]. Finally, the last group is related to the distributed one. These algorithms are used for datasets, which are distributed in different systems, such as big data. These kinds of algorithms are designed to find outliers by considering the patterns of all data in all systems[4][14].

The rest of this section is representing two popular algorithms, that are used in our experiments.

2.1. Density-Based Spatial Clustering of Applications with Noise(DBSCAN)

DBSCAN is one of the density-based clustering algorithms[15]. In this algorithm, two parameters are inputs (μ, ε), and points or data are categorized as core points, reachable points, and outliers. They are defined as follow:

Core points are the points with at least μ neighbors in the border of ε from itself.

Reachable points are those points, which are not core points but are located in the border of ε from one or more than one core point.
Outliers are those points which are not core or reachable points.

2.2. Local Outlier Factor (LOF)

The goal of LOF, which is based on the local density and nearest neighbors, is computing the local outlier factor, which is outliers' determination[16]. For computing LOF of each point, calculating the reachability distance between the points and their k nearest neighbors, and also Local Reachability Density (LRD) of the points are necessary.

Reachability distance between two points (p, q) is defined as:

reach-dist(p, q) = max(k-distance(q), d(p, q)) (1)

In Equation 1, k-distance(q) is the distance between q and its kth nearest neighbor, and d(p, q) is the distance between p and q. the concept of reachability distance is shown in Figure 2.

And local reachability density of each point is defined as follow:

$$LRD(p)=1/\left(\frac{\sum_{q\epsilon knn(p)} reach-dist(p.q)}{||k-neighborhood||}\right) \qquad (2)$$

In Equation 2, q is the neighbor of p, reach-dist(p, q) is the reachability distance between p and q, and k-neighborhood is the number of p's neighbors.

Finally, the Local Outlier Factor of each point is defined as:

$$LOF(p)=\left(\frac{\sum_{\boldsymbol{q\ \epsilon\ knn(p)}}\frac{\boldsymbol{LRD(q)}}{\boldsymbol{LRD(p)}}}{||\boldsymbol{k-neighborhood}||}\right) \qquad (3)$$

In Equation 3, q is the neighbor of p, LRD(p) and LRD(q) are local reachability density of p and local reachability density of q respectively, and k-neighborhood is the number of p's neighbors.

Whatever LOF is higher, the point is more abnormal.

3. Discovering The Structure Based On Text Similarity And Density Peak

As we present in Section 3, the clustering algorithm is one of the groups of outlier detection, and this kind of algorithm can help to find abnormal data and the patterns of datasets. As mentioned in [17], the center of each cluster is the point with a higher density than its neighbors and more different from the points with higher densities. In this case, the measurement of similarity is the distance between two points, but we prepare the algorithm for text data, therefore, cosine similarity is used. As a result, we propound local density for each document, denoting the number of text data or documents, that are similar to this data. For this purpose, threshold and local density are considered as follow:

$$\rho_i = \sum_j X(t_i \,.\, t_j) \,,\ X(a,b) = \begin{cases} 1 & \text{Similarity } (a, b) > \text{threshold} \\ 0 & \text{Similarity } (a, b) < \text{threshold} \end{cases} \qquad (4)$$

In Equation 4, ρ_i is the local density of document i, and j is the number of the documents in a dataset and t_i is an ith document in the dataset. Figure 3 demonstrates computing ρ_i.

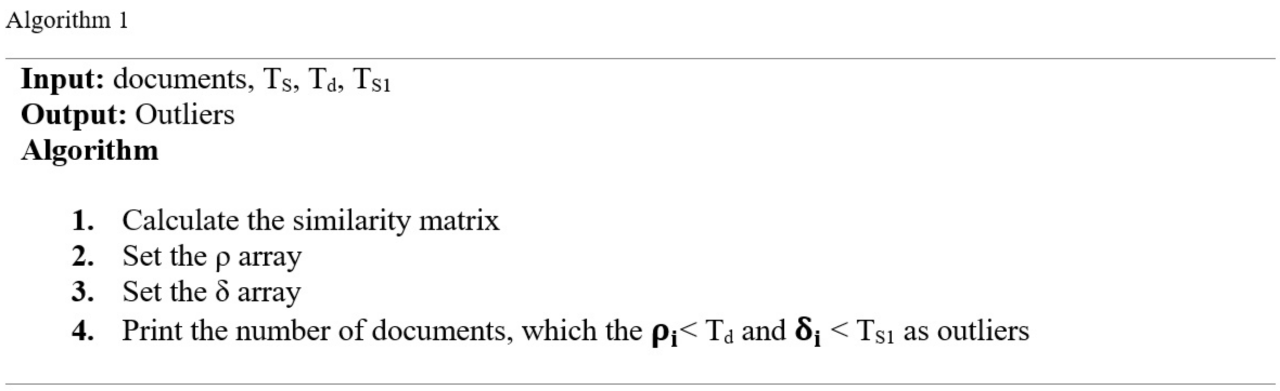
δ_i is the similarity between document i and the most similar document with higher local density. For example, as we can see in Figure 4, δ for the black document is the similarity between the gray document and itself.

Eventually, we suggest outliers that are more different from the behavior of central data. In other words, outliers are texts with less ρ_i and less δ_i

Considering the purpose of finding outliers in text data and density peak, the rest of this section is explaining our new method.

First of all, the similarity of each pair of text is calculated and is saved in a matrix. As is shown in Figure 5.

Then a threshold should be set for the similarity between documents. Now for each document, ρ_i is computed and is saved in ρ array. Afterward, δ_i should be recognized for all documents and is saved in δ array. Finally, outliers are searched as the □$_{□}$ and δ_i are lower than the thresholds. We can see the steps of the algorithm in Algorithm 1.



Algorithm 1

```
Input: documents, $T_S$, $T_d$, $T_{S1}$
Output: Outliers
Algorithm

    1. Calculate the similarity matrix
    2. Set the ρ array
    3. Set the δ array
    4. Print the number of documents, which the $\rho_i < T_d$ and $\delta_i < T_{S1}$ as outliers
```

4. Methods

In order to test the effectiveness of our proposed outlier detection in the text, three datasets are used in our experiment. Each dataset is tested in different conditions. Firstly, they are prepared for entering into classification algorithms by different outlier detection algorithms. In these cases, we clean data with our method, LOF, DBSCAN and in another status, the datasets are unchanged. Finally, we calculate the accuracies of four different classification algorithms and compare them with each other. The goal of these experiments is to investigate the effect of cleaning outliers with our method on the performance of classification algorithms. We use python 3.6 for implementation and moreover, the SKlearn library is utilized for implementing the classification algorithms. In the following, the details are expressed, and the results are presented.

4.1. Datasets

Three datasets are used for the experiments, that two of them are for the BBC website and one of them is collected from different newsgroups. The first one is BBC news, which consists of 2225 documents from the BBC website from 2004 to 2005. They have five labels in the majors of business, entertainment, politics, sport, and technology[18]. The second one is BBC-Sport that consists of 737 documents from the BBC Sport website from 2004 to 2005. They have five labels in majors of athletics, cricket, football, rugby, and tennis[18]. Eventually, the last one is 20-newsgroup dataset. It is collected from 20 different newsgroups, that 1000 articles were taken from every newsgroup, and an overall number of documents is around 20000. Each document consists of the subject line and body[19], that in our experiment both of them are utilized for detecting outliers and classification algorithms.

4.2. Classification Algorithms

In our experiments, four different classification algorithms (K Nearest Neighbor, Decision Tree, Random Forest, and Naïve Bayes) are implemented to compare different situations. Furthermore, Term Frequency(TF), Term Frequency-Inverse Document Frequency(TF-IDF), and 3-gram are used for the inputs of these algorithms. We briefly describe these four algorithms at the rest of this section. The first algorithm is K Nearest Neighbor(KNN). In this algorithm, a group of labeled data is kept, then once a new data enters into the system, it is labeled regarding its K nearest neighbors[20][21]. For implementing this algorithm, different values of K are considered. The second algorithm is Decision Tree(DT), which makes a tree from labeled data when a new data enters into the algorithm, which is similar to yes and no question game, predicts the label[22]. The third selected algorithm is the Random Forest(RF). This algorithm makes the number of random decision trees with different properties, then the label of new data is decided from the result of these trees[23]. For implementation, the algorithm is run ten times, and the average of all results is used in our experiment. The last algorithm is Naïve Bayes, which is a statistical algorithm that takes labeled data to generate statistical models, then when a new data enters into the system, it refers to the model and the model predicts the label[24][25].

5. Results And Discussion

Table 1 is the experimental results of the classification algorithms with the conditions listed on the BBC dataset.

Table 1. The accuracy of Decision Tree, Random forest, and Naïve Bayes algorithms on the BBC dataset

Algorithm		Decision Tree				
Outlier detection		Without deleting outliers	LOF (K=7)	DBSCAN (ε=3,μ=2)	Our Method	
Accuracy	TF	0.8071	0.8606	0.9354	0.8669	
	TF_IDF	0.8071	0.8507	0.8116	0.8394	
	N-gram (n=3)	0.7533	0.7114	0.7623	0.7477	
Algorithm		Random Forest				
Outlier detection		Without deleting outliers	LOF (K=7)	DBSCAN (ε=3,μ=2)	Our Method	
Accuracy	TF	0.8917	0.9197	0.9548	0.9149	
	TF_IDF	0.9058	0.9217	0.9031	0.9278	
	N-gram (n=3)	0.8466	0.8235	0.8463	0.8522	
Algorithm		Naïve Bayes				
Outlier detection		Without deleting outliers	LOF (K=7)	DBSCAN (ε=3,μ=2)	Our Method	
Accuracy	TF	0.9596	0.98	1.0	0.9862	
	TF_IDF	0.9686	0.985	0.9686	0.9816	
	N-gram (n=3)	0.9327	0.9651	0.9327	0.9633	

As shown by this table, after deleting outliers by our method, the accuracies of Decision Tree algorithms (except in vector of N-gram) are improved. the accuracy of DT diminishes when data cleaning occurs by LOF and our method with n-gram input. Perhaps in these cases, separating the criteria of DT face the problem of deleting outliers. In all situations, the accuracies of Random Forest improve after implementing our algorithm, and in most situations was first. Furthermore, with our method and TF vector, the accuracies of Naïve Bayes increased. In this status, DBSCAN has better operation among different outlier detections, but the accuracy of this method is one, that is not good at all times, due to overfitting. In the others, our algorithm has good functions, or it is near the top accuracy.

Figures 6 demonstrates that, we can get, the accuracies of K nearest neighbor after implementation of our method are top (in all conditions with k between 4 to 14).

In the BBC dataset, if all conditions are compared, our algorithm has the best accuracy in KNN classification, but in the others, the accuracy is increased by our method, except in one situation that the accuracies of our method and LOF decreased toward the accuracy of the raw dataset.

Table 2 shows the accuracies of Decision Tree, Random Tree, and Naïve Bayes on the BBC Sports dataset.

As shown in Table 2, after implementing our algorithm, the accuracies of the decision tree with TF and TF-IDF vectors are the best. However, it decreases in the N-gram vector toward the accuracy of the raw dataset. This increase is obtained in the same condition on the BBC dataset, likewise. Considering the accuracy of Random Forest, we can see that the accuracy of our method with N-gram vectors decreases and in the other cases, it is not as well as LOF. Also, it does not have a good operation in Naïve Bayes, as well.

Table 2. The accuracies of Decision Tree, Random Forest, and Naïve Bayes algorithms on the BBC Sport dataset

Algorithm		**Decision Tree**				
Outlier detection		**Without deleting outliers**	**LOF (K=7)**	**DBSCAN (ε=3,μ=2)**	**Our Method**	
Accuracy	**TF**	**0.8648**	**0.8656**	**0.8918**	**0.9166**	

	TF_IDF	0.8243	0.8208	0.7972	0.8472	
	N-gram (n=3)	0.9054	0.9402	0.9324	0.875	
Algorithm		Random Forest				
Outlier detection		Without deleting outliers	LOF (K=7)	DBSCAN (ε=3,μ=2)	Our Method	
Accuracy	TF	0.8828	0.9124	0.8702	0.8935	
	TF_IDF	0.8945	0.9363	0.8909	0.9120	
	N-gram (n=3)	0.8918	0.9333	0.9072	0.8861	
Algorithm		Naïve Bayes				
Outlier detection		Without deleting outliers	LOF (K=7)	DBSCAN (ε=3,μ=2)	Our Method	
Accuracy	TF	0.9864	0.9253	0.9864	0.9722	
	TF_IDF	0.9054	0.9253	0.9054	0.9166	
	N-gram (n=3)	0.9459	0.9104	0.9459	0.9166	

Figures 7 shows the results of KNN in different conditions. Generally, in most cases, we can see accuracy fluctuation. For example, the accuracy of DBSCAN with TF vectors and K = 8 is near 0.9 and with K = 13 and 14 is under 0.5. Besides, in this condition and N-gram vectors, the accuracies of the raw dataset are the best. It shows that with TF-IDF our algorithm and LOF are better than DBSCAN and the raw data.

According to Section 4.1, the BBC Sports dataset has a few numbers of documents, and consequently, our algorithm is based on the density. In fact, the low number of text data can affect our method. Also, because of a low number of data, the underfitting is possible. The BBC Sport dataset is sports news, therefore most of these kinds of news are similar to each other, and detecting outliers is quite difficult in these conditions.

Table 3 represents the accuracies of classification algorithms on the 20-newsgroup dataset.

Table 3. The accuracies of the decision tree, random forest, and Naïve Bayes algorithms on the 20-newsgroups dataset

Algorithm		**Decision Tree**				
Outlier detection		**Without deleting outliers**	**LOF (K=7)**	**DBSCAN (ε=3,μ=2)**	**Our Method**	
Accuracy	**TF**	**0.6519**	**0.6594**	**0.6666**	**0.6556**	
	TF_IDF	**0.6236**	**0.6555**	**0.6068**	**0.6512**	
	N-gram (n=3)	**0.4876**	**0.4749**	**0.4885**	**0.4679**	
Algorithm		**Random Forest**				
Outlier detection		**Without deleting outliers**	**LOF (K=7)**	**DBSCAN (ε=3,μ=2)**	**Our Method**	
Accuracy	**TF**	**0.6723**	**0.6906**	**0.8740**	**0.6764**	
	TF_IDF	**0.7166**	**0.7213**	**0.7217**	**0.7278**	
	N-gram (n=3)	**0.5585**	**0.5417**	**0.5548**	**0.5652**	

Algorithm		**Naïve Bayes**				
Outlier detection		**Without deleting outliers**	**LOF (K=7)**	**DBSCAN (ε=3,μ=2)**	**Our Method**	
Accuracy	**TF**	**0.8418**	**0.8370**	**0.7777**	**0.8345**	
	TF_IDF	**0.8445**	**0.8518**	**0.8445**	**0.8443**	
	N-gram (n=3)	**0.6554**	**0.6506**	**0.6554**	**0.6637**	

Table 3 is shown, that the accuracies of decision tree except for the N-gram vectors (this pattern is repeated in the other datasets, as well) are improved and sometimes it is better than DBSCAN. In random forest, we can see growth in all conditions, and in TF-IDF and N-gram vectors our method is the most appropriate. Considering the Naïve Bayes, we can observe, that our algorithm with N-gram vectors is the best, in other conditions the accuracies decreases.

Figures 13 to 15 demonstrate the accuracies of the KNN algorithm in different conditions. In these cases, we can see an increment of accuracy. It is obvious that in TF_IDF vectors LOF is the best. Generally, we cannot see a sudden decrease in our algorithm on the 20-newsgroups, and in most conditions, the accuracy is improved by our algorithm and also is better than the others.

All in all, our algorithm has acceptable functionality in different situations, especially on the BBC and 20-newsgroups. Also, we can see the best operation in KNN on the BBC dataset, and the random forest has the growth of accuracies in all conditions. It is obvious, the most irregularity of accuracy occurs on the BBC Sports dataset.

6. Conclusion

According to text properties, processing of this kind of data is problematic. However, detecting abnormal data can make processing easier. Thus, in this paper, we presented a new method for finding abnormal data in text datasets. We tried to find the texts, that their properties were not similar to the center of the clusters. Following this idea, we designed the method based on density-based. Then, we conducted a series of experiments for comparing our method with two popular outlier detections(LOF and DBSCAN). In our experiments, three datasets (BBC, BBC Sport, and 20-newsgroups) with different volumes were selected. The accuracies of these three datasets in 48 conditions for 4 classification algorithms (KNN, DT, RF, and Naïve Bayes) were computed. In most conditions, not only, our proposed algorithm outperformed LOF and DBSCAN algorithms but did not have the changing problem seen in the DBSCAN algorithm. If the conditions are compared together, it is obvious that our algorithm does not have a good function on datasets with similar documents and a low number of data. Due to underfitting and overfitting problems.

Finally, our proposed algorithm is independent of the knowledge of systems, and it can be used for short or long text and can be implemented on most datasets. Furthermore, the flexibility of our algorithm can be increased by employing other similarity methods such as semantic similarity.

Abbreviations

DBSCAN: Density-Based Spatial Clustering of Applications with Noise; LOF: Local Outlier Factor; LRD: Local Reachability Density; TF: Term Frequency; TF-IDF: Term Frequency-Inverse Document Frequency; KNN: K Nearest Neighbor; DT: Decision Tree; RF: Random Forest.

Declarations

Authors’ contributions

MT was the major contributor in conception and design, and also implemented and performed the algorithms. MZ helped for the interpretation of the results of this experiment and was involved in the preparation of the manuscript and also critically revised it. The manuscript was written by MT. Both authors read and approved the final manuscript.

Acknowledgment

Not applicable

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

The BBC and BBCSport Datasets that are used in our experiments are available in Insight Resources repository, http://mlg.ucd.ie/datasets/bbc.html; The 20-news dataset used in our experiment is available in The datasets of SKlearn library. All other data are available from the authors upon reasonable request.

Funding

Not applicable

References

[1]R. Kumaraswamy, A. Wazalwar, T. Khot, J. Shavlik, and S. Natarajan, “Anomaly Detection in Text : The Value of Domain Knowledge,” in *Proceedings of the Twenty-Eighth International Florida Artificial Intelligence Research Society Conference Anomaly,* 2015, pp. 225–228.

[2]B. S. Kumar and V. Ravi, “A survey of the applications of text mining in financial domain,” *Knowledge-Based Syst.,* vol. 114, pp. 128–147, 2016.

[3]Y. Zhao, Z. Nasrullah, and Z. Li, “PyOD : A Python Toolbox for Scalable Outlier Detection,” *J. Mach. Learn. Res.,* vol. 20, pp. 1–7, 2019.

[4]Y. Yan, L. Cao, C. Kuhlman, and E. Rundensteiner, “Distributed Local Outlier Detection in Big Data,” in *KDD ’17 Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* 2017, pp. 1225–1234.

[5]S. Vijayarani *et al.,* “Preprocessing Techniques for Text Mining - An Overview,” *Int. J. Comput. Sci. Commun. Networks,* vol. 5, no. 1, pp. 7–16, 2015.

[6]J. Allan, V. Lavrenko, D. Malin, and R. Swan, “Detections, Bounds, and Timelines: UMass and TDT–3,” *Inf. Retr. Boston.,* pp. 167–174, 2000.

[7]M. Platakis, D. Kotsakos, and D. Gunopulos, “Searching for events in the blogosphere,” *ACM 978–1–60558–487–4/09/04.,* p. 1225, 2009.

[8]J. Guzman and B. Poblete, “On-line Relevant Anomaly Detection in the Twitter Stream: An Efficient Bursty Keyword Detection Model,” in *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description - ODD ’13,* 2013, pp. 31–39.

[9]E. Schubert, M. Weiler, and H.-P. Kriegel, “SigniTrend: Scalable Detection of Emerging Topics in Textual Streams by Hashed Significance Thresholds,” in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD ’14,* 2014, pp. 871–880.

[10]V. J. Hodge and J. Austin, “A Survey of Outlier Detection Methodologies.,” *Artif. Intell. Rev.,* vol. 22, pp. 85–126, 2004.

[11]Y. Liu *et al.,* “Generative Adversarial Active Learning for Unsupervised Outlier Detection,” *IEEE Trans. Knowl. Data Eng.,* pp. 1–13, 2019.

[12]I. Ben-gal, “Outlier Detection,” in *Data Mining and Knowledge Discovery Handbook,* 2005, pp. 131–146.

[13]C. Barreyre, L. Boussouf, and B. Cabon, “Statistical Methods for Outlier detection in Space Telemetries,” *Sp. Oper. Inspiring Humankind’s Futur.,* 2019.

[14]X. Qin, L. Cao, E. A. Rundensteiner, and S. Madden, “Scalable Kernel Density Estimation-based Local Outlier Detection over Large Data Streams,” in *Proceedings of the 22nd International Conference on Extending Database Technology,* 2019.

[15]M. Ester, H. Kriegel, J. Sander, and X. Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” in *KDD–96 Proceedings.*

[16]M. M. Breunig *et al.,* “LOF: identifying density-based local outliers,” in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data - SIGMOD ’00,* 2000, vol. 29, no. 2, pp. 93–104.

[17]A. Rodriguez and A. Laio, “Clustering by fast search and find of density peaks Alex Rodriguez and Alessandro Laio,” *Science (80-.).,* vol. 1492, 2014.

[18]D. Greene, “Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering,” in *ICML ’06 Proceedings of the 23rd international conference on Machine learning,* 2006, pp. 377–384.

[19]T. Joachims, “A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization.pdf,” *Def. Tech. Inf. Cent.,* 1996.

[20]M. Oghbaie and M. M. Zanjireh, “Pairwise document similarity measure based on present term

set," *J. Big Data,* no. February, 2019.

[21]N. S. Altman, "An Introduction to Kernel and Nearest-Neighbour Nonparametric Regression," *Am. Stat.,* vol. 46, no. 3, pp. 175–185, 1992.

[22]B. Kami and M. Jakubczyk, "A framework for sensitivity analysis of decision trees," pp. 135–159, 2018.

[23]T. K. Ho, "Random Decision Forests Tin Kam Ho Perceptron training," *Proc. 3rd Int. Conf. Doc. Anal. Recognit.,* pp. 278–282, 1995.

[24]I. Rish, "An empirical study of the naive Bayes classifier," *Empir. methods Artif. Intell. Work. IJCAI,* vol. 22230, no. JANUARY 2001, pp. 41–46, 2001.

[25]J. Han, M. Kamber, and J. Pei, *Data Mining concept and techniques,* Third Edit. Elsevier, 2012.

Figures

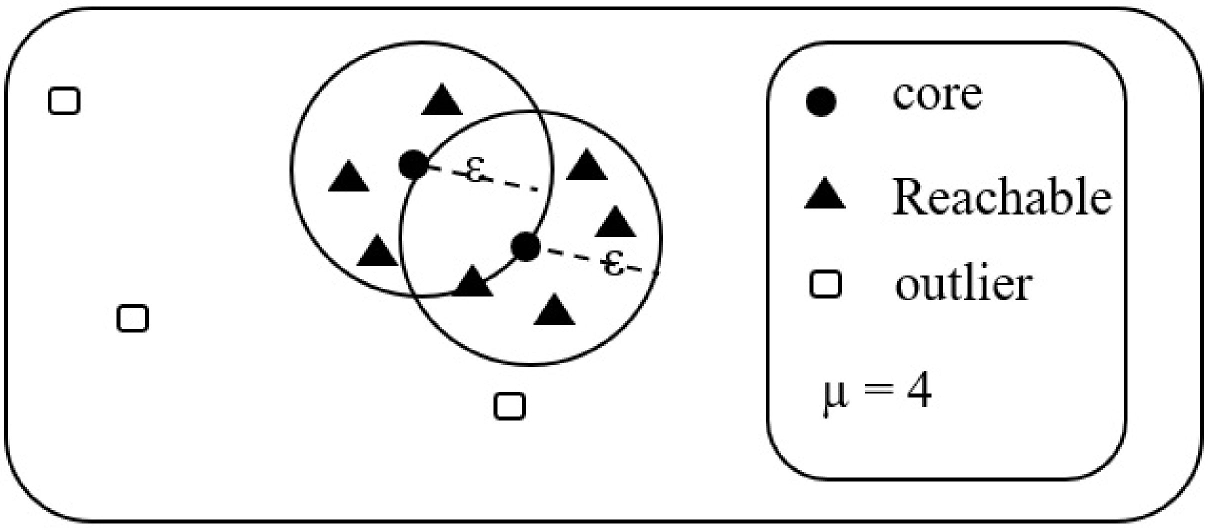


Figure 1

Core, Reachable, and Outlier Points in DBSCAN

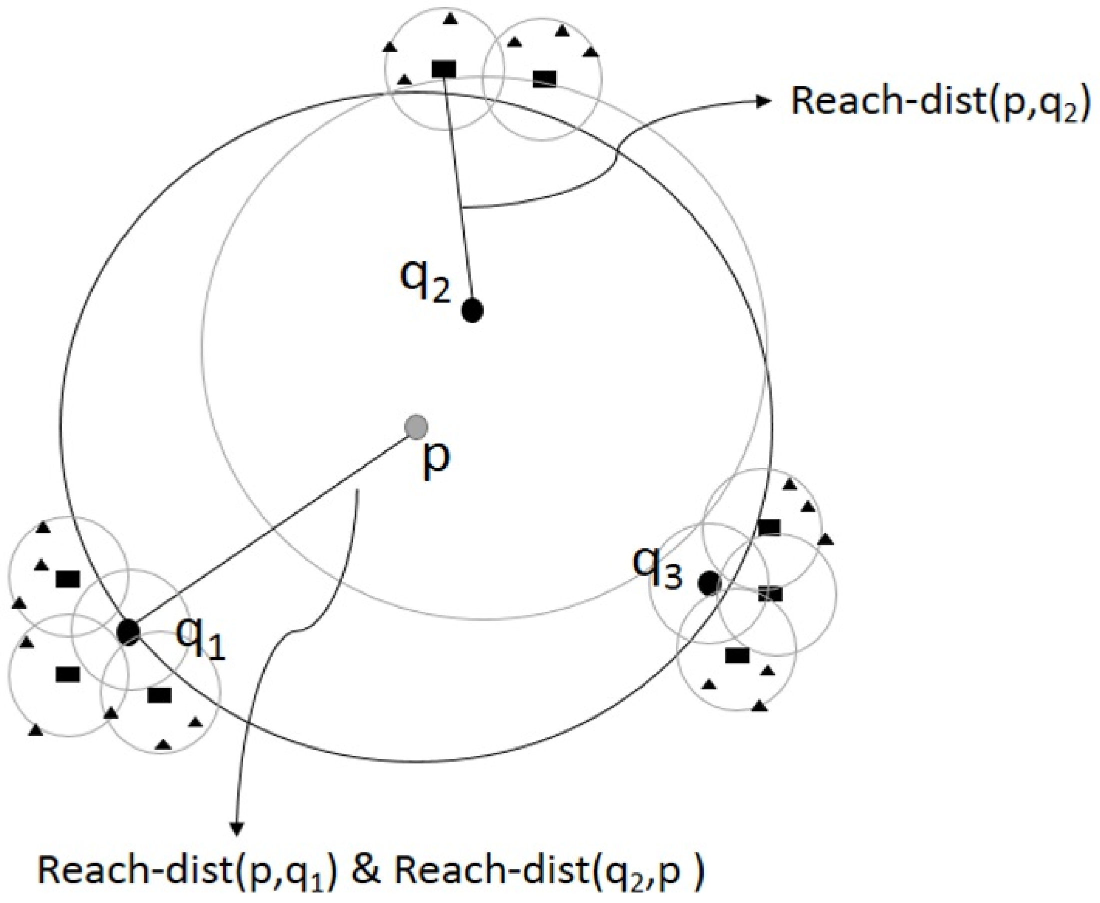


Figure 2

Reachability Distance Between Two Points[4]

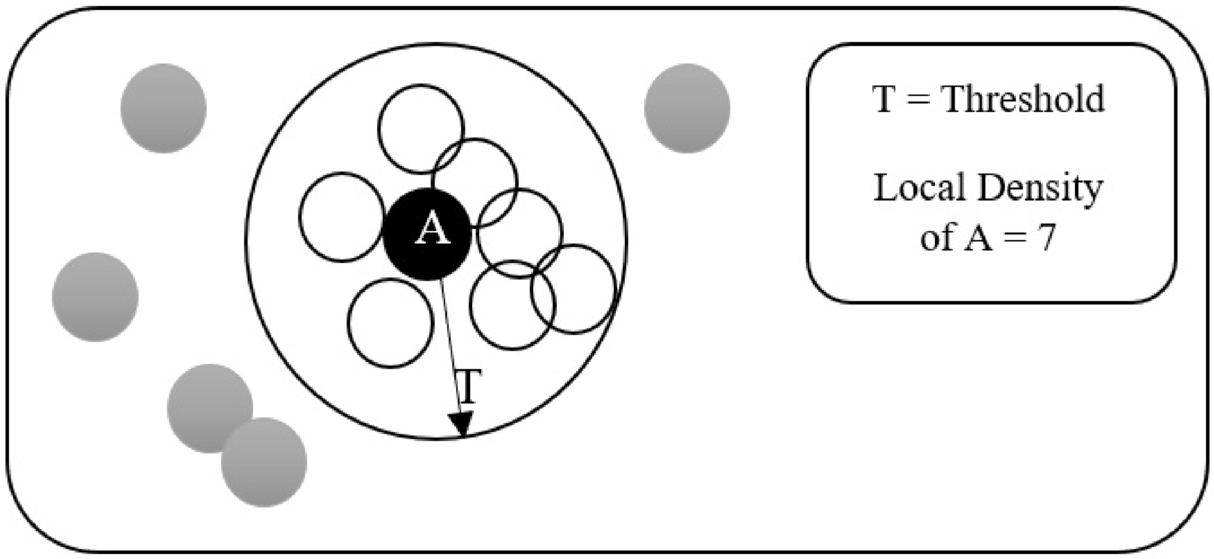


Figure 3

The local density of a document

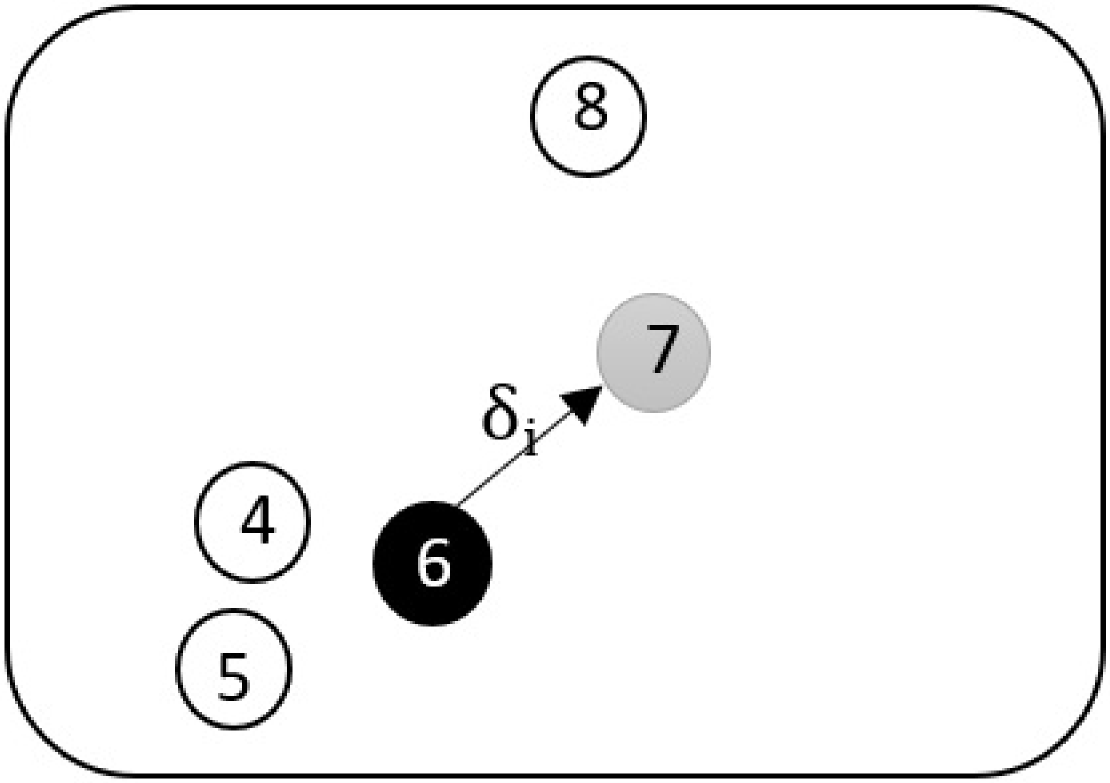


Figure 4

The computing of δi

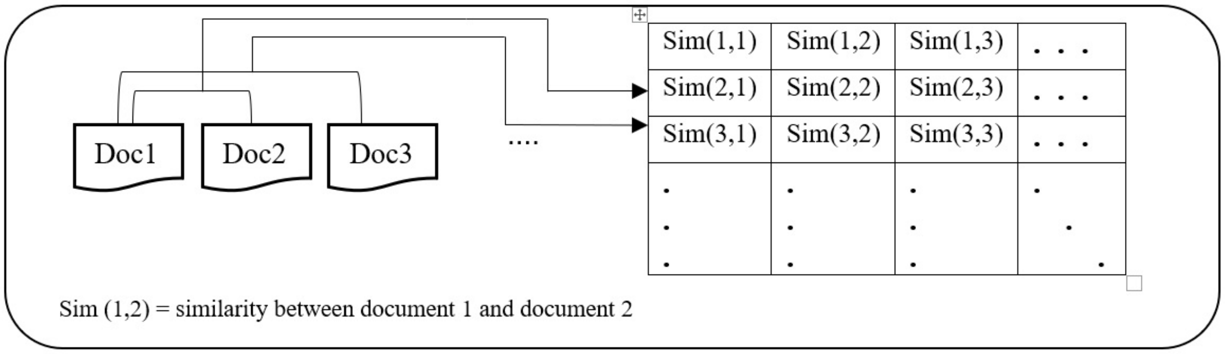

Doc1
Doc2
Doc3
....
Sim(1,1) Sim(1,2) Sim(1,3) . . .
Sim(2,1) Sim(2,2) Sim(2,3) . . .
Sim(3,1) Sim(3,2) Sim(3,3) . . .
Sim (1,2) = similarity between document 1 and document 2

Figure 5

The computing of similarity matrix.

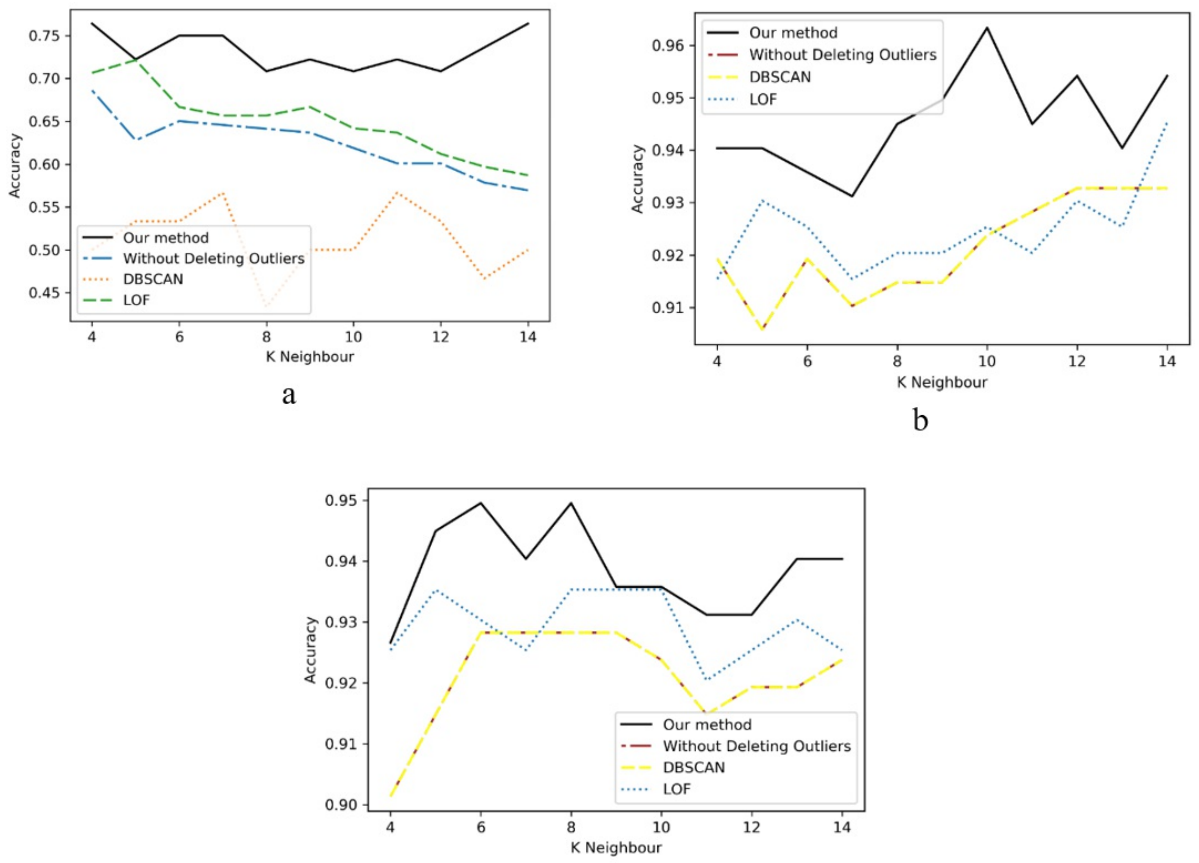
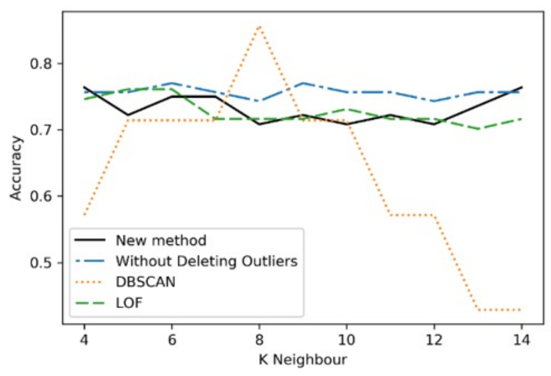
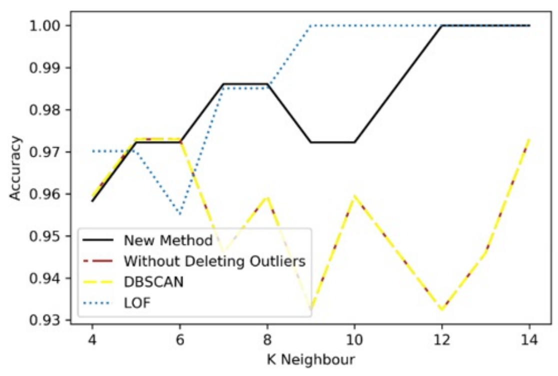


Figure 6

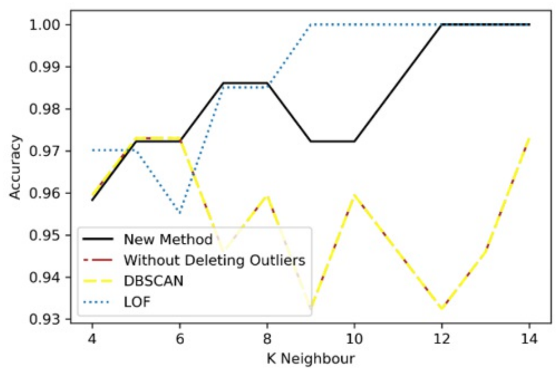
The accuracies of KNN on the BBC dataset with k between 4 to 14 with: a.TF, b.TF-IDF and c.N-gram vectors



a



b



c

Figure 7

The accuracies of KNN on the BBC Sport dataset with k between 4 to 14 with a.TF, b.TF-IDF and c.N-gram vectors

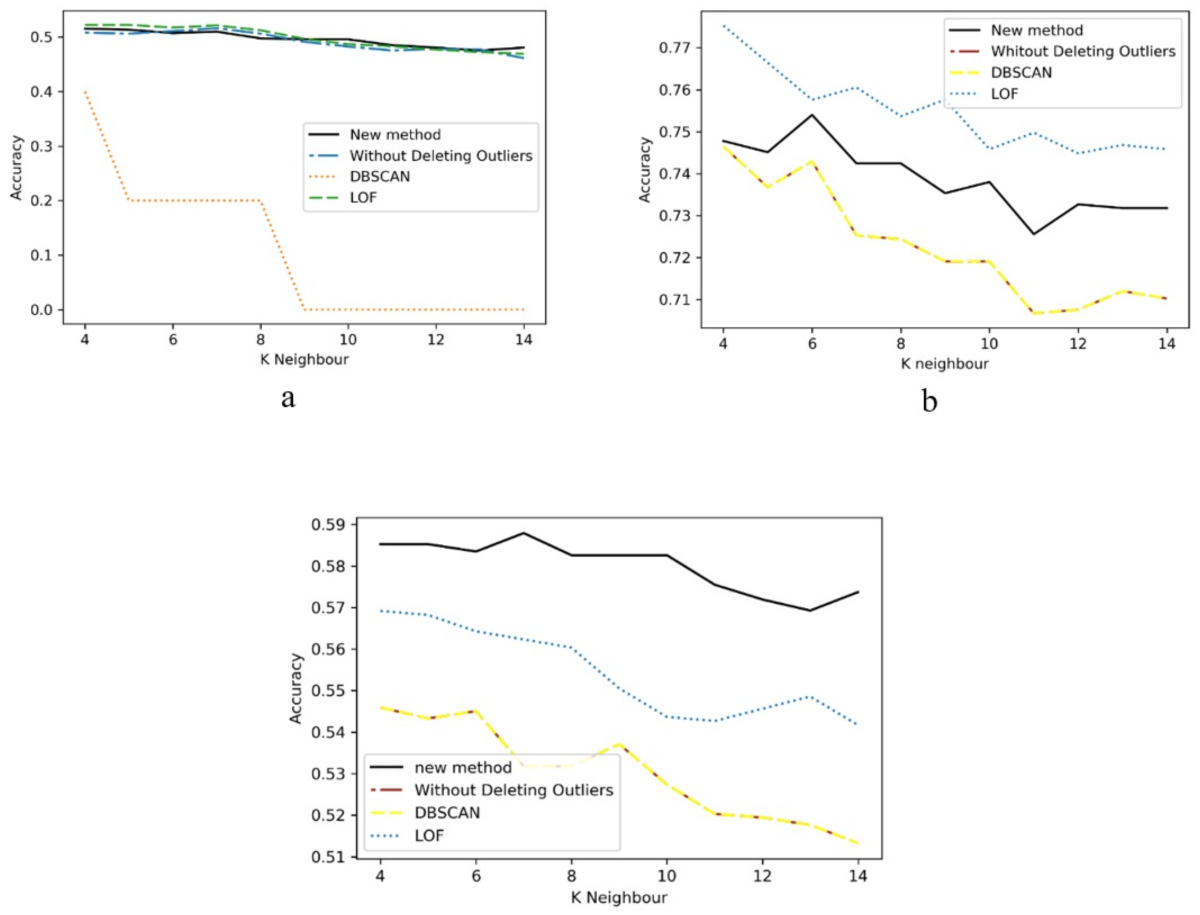


Figure 8

The accuracies of KNN on the 20-newsgroups dataset with k between 4 to 14 with a.TF,

b.TF-IDF and c.N-gram vectors