Question #1 Report

Implementation of common hard assignment clustering algorithms

Abhinav Khanna(2017CSB1061)

CS524: Data mining

INTRODUCTION

In the area of clustering algorithms using hard assignment, 2 major algorithms were discussed: K-means and DBscan. The purpose of this project is to implement these algorithms on various datasets, analyze their space and time complexity, and conclude which algorithm is better for which dataset. The scope of this project is limited by the hardware capabilities, such as RAM, number of cores/processors, and CPU clock cycles, availability and usage of GPUs et al.

THEORETICAL FOUNDATION

K-means clustering is a representative based clustering algorithm. We first choose K-random means, and form clusters, where each point is assigned a cluster nearest to it. In our case, I have used the Euclidean distance as the distance metric. New means are calculated, and then new clusters are formed following the same condition. This process is repeated until there is convergence of the means.

DBscan is a density based clustering method. We firstly identify the core points in the dataset. Then, for each core point which is not yet a part of any cluster, we perform a recursive method to identify all the points that are density connected(including other core points). This gives the clusters, as well as the noise points.

IMPLEMENTATION

- ### K-means Algorithm

  Data Preparation

  Dataset file is read, and each point is stored as a list. Thus, our entire dataset is a list of lists.

  Clustering implementation

  As explained above, firstly random means are chosen, and then updation of clusters is done until convergence of means. The algorithm itself is run 30 times, with values of K=1 to K=30. A graph is plotted containing values of SSE vs K, so as to find the optimal value of K for further use.

- ### DBscan

  Data Preparation

  Data points are stored as a list of lists.

  Clustering implementation

  The algorithm is run for a given value of epsilon and min_pts. This is to

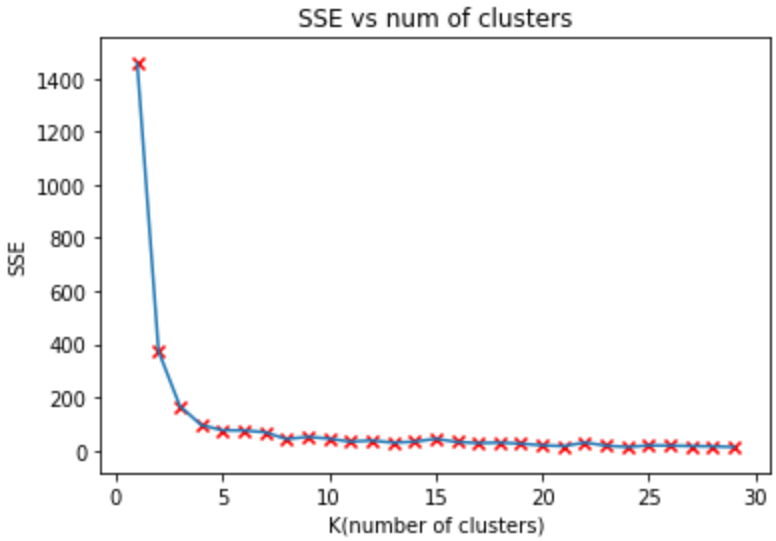
determine the core points, border points and noise points. As explained earlier, clusters are formed on the basis of the collection of density connected core points and their corresponding border points. Initially all clusters are assigned cluster_number = -1. After running the algorithm, data points that have been assigned to some cluster are core/border points, while others are classified as noise points.

The input parameters epsilon and min_pts are set so as to get the same number of clusters as the optimum number of clusters inferred from the K-means clustering.

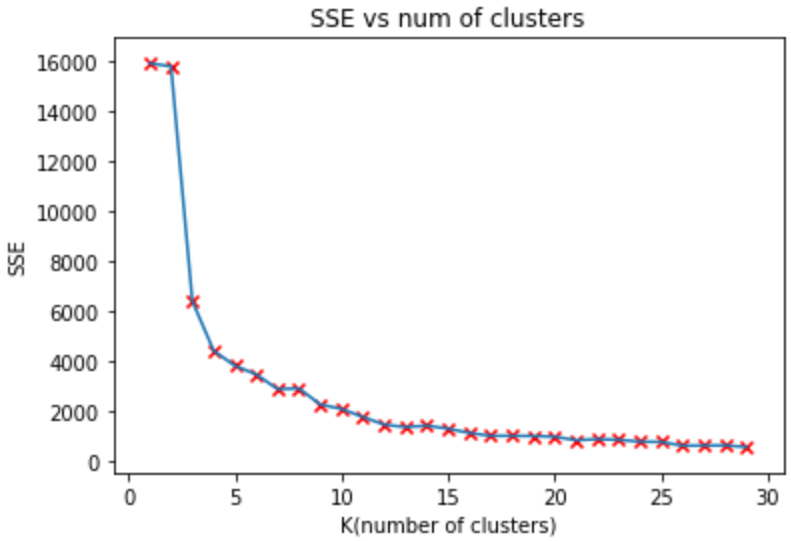
OUTPUTS AND OBSERVATIONS

- ### K-MEANS clustering

  Iris dataset: 150 points



  2d spiral dataset: 1000 points



  As is clearly visible, the value of **K=3** is the optimum number of clusters for the Iris dataset. We can also see that the K-means algorithm does not work effectively

for the 2d spiral dataset.

Time taken for Iris dataset (3 clusters): 0.005 sec

Time taken for 2d spiral dataset(2 clusters): 0.095 sec

- DBscan

Parameters	Iris dataset	2d spiral dataset
# of clusters	3	2
SSE	232.4	14374.85
# of noise points	13	23

Thus, if we compare the results of DBscan with K-means for Iris dataset, we see that K-means gives SSE of 166.9, whereas DBscan gives SSE 232.4 . Thus, the optimal number of clusters is 3, and **K-means is a better option for Iris dataset**. For the 2d spiral dataset, the number of clusters is 2. Using K-means clustering, we get SSE 15758.67, whereas with DBscan we get SSE 14374.85. Therefore, **DBscan is a better algorithm for 2d spiral dataset.**

Time taken for Iris dataset (3 clusters): 0.0838 sec

Time taken for 2d spiral dataset(2 clusters): 2.32 sec

CONCLUSION

Thus, we can conclude that in terms of performance i.e. minimizing SSE, there is no clear winner. K-means clustering may work for some dataset, but DBscan may be better for others. However, in terms of time complexity, for both the algorithms, we can see that **DBscan is much slower than K-means.** This can be attributed to the fact that DBscan involves scanning the whole dataset for each data point, to find epsilon-neighborhood, which leads to O(n^2) time complexity.

REFERENCES

1. Data Mining and Analysis: Fundamental Concepts and Algorithms: Textbook by Mohammed J. Zaki and Wagner Meira
2. Class notes