CS6890: Fraud Analytics Using Predictive and Social Network Techniques, 2020

Assignment 1: Identifying fraudulent Taxpayers using Spectral Clustering

SAI HARSHA KOTTAPALLI

CS17BTECH11036

OBJECTIVE

The objective for this assignment is to identify fraudulent tax payers from the given dataset containing various details(unlabelled) regarding their transactions.

Since, the data is unlabelled, we try to implement an unsupervised learning algorithm to try to split the data into different clusters in hopes of identifying the fraudulent taxpayers.

We can't depend on K-means always as it only tends to find spherical clusters which might cause it to perform poorly.

For this assignment, we will use spectral Clustering to achieve this objective where we try to do a minimal graph cut such that we obtain multiple Strongly connected graphs representing clusters. We use similarity graphs to encode local neighbourhood information to attain the initial graph structure. Since, identifying the minimum cut is NP-hard we use graph laplacian for efficient approximation.

Finally depending on the number of clusters we have obtained after spectral clustering we will try to identify which ones are fraudulent. The main motivation to follow a clustering algorithm for this purpose is from the idea that points assigned to the same cluster are highly similar to one another and the points in other clusters are highly dissimilar from current cluster points.

DATA SET SUMMARY

Our dataset has 9 features. Data in the dataset is not labelled. We also notice that the dataset is normalized but we do not know what method was chosen to normalize the data. I have tried normalizing but have found that data was divided into equal halves, so I haven;t applied normalization on it for clustering. We have 1163 rows of data each representing a person's transaction history features.

EXPERIMENTATION RESULTS

Applying any sort of normalization on the data, skews up the result (the algorithm tends to cluster the data into two equal halves), this is expected as data was already normalized by some method, and it wouldn't help much in the case of clustering algorithms where distance is considered for similarity. This can be explained by the fact that doing so result in the data points being rearranged in the n dimensional space and hence potentially losing its local neighbourhood information or structural information.

We then move on to calculating the similarity matrix. Here, we represent each data point as the vertex of a graph and an undirected edge between each node representing a similarity value. This value is obtained by applying gaussian kernel:

$$e^{-\sum_{i=1}^{9}(X_i-Y_i)^2}$$

We then make sure all self edges are set to 0, as we don't want the node itself to be the nearest neighbor. Out of curiosity, i have even tried running with self edge 1, but this resulted in imaginary eigenvalues so it is a must to zero the self edges.

Now to obtain the adjacency matrix, we find the three most nearest neighbours with the help of the similarity matrix. Upon finding them we assign an undirected edge between this node and its 3 nearest neighbours. Doing so for all the nodes would result in a graph which we treat as the adjacency matrix(A) for further part of the clustering algorithm.

We then create a degree matrix of size n x n for normalizing the adjacency matrix.

Now as we are modelling the clustering algorithm as a graph cut problem, we use graph laplacian for efficient approximation.

Now we create normalized Laplacian matrix using:

$$D^{-0.5}(D - A)D^{-0.5}$$

After this we try to get the eigenvalues and the corresponding eigenvectors for the laplacian matrix. We notice that the values are real and non-negative numbers. As explained in the class, the reason we are trying to do eigen decomposition is that it essentially leads to cluster objective functions(the eigenvalues).

At this point we can find out optimal value k that is number of clusters which we have to divide the dataset into by two method:

1. By taking the largest eigen gap,

   $$\Delta_k = \left|\lambda_k - \lambda_{k-1}\right|$$

2. Number of eigenvalues almost equal to zero tells us how many clusters we can divide into.

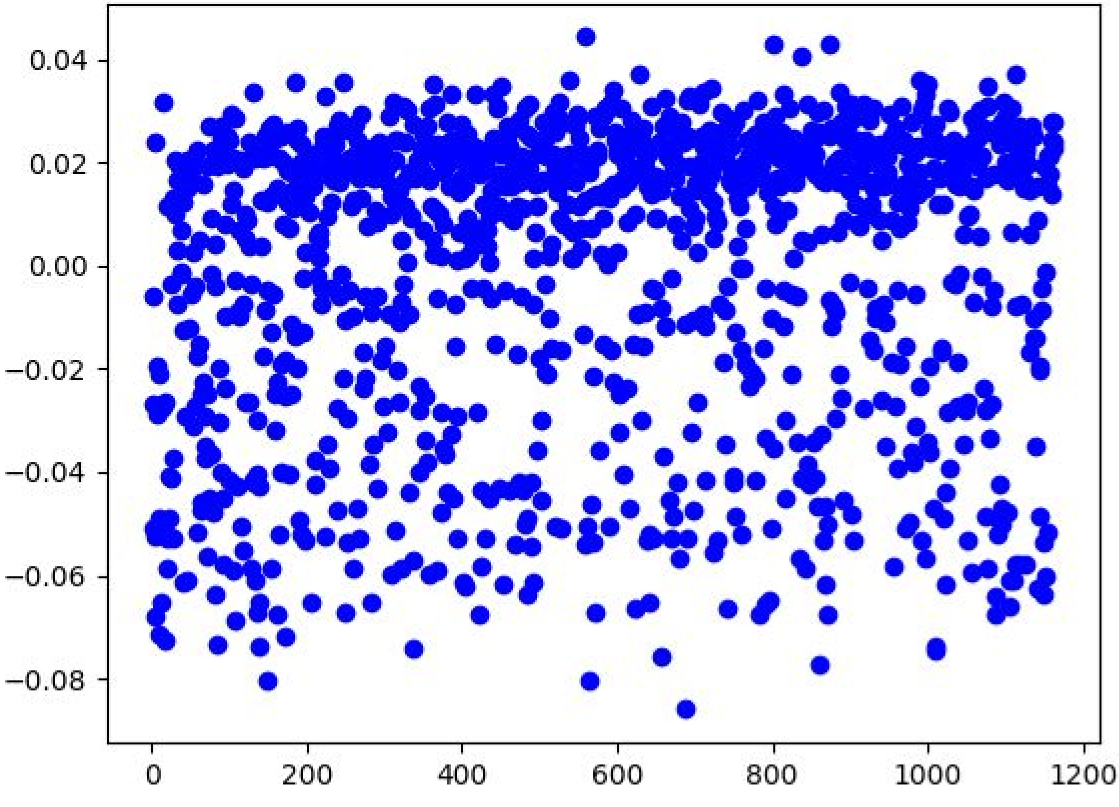
As proven through code, we notice that through both these methods we get optimal number of clusters as 2.

Now since this is the case, we split the data points to 2 clusters using the Fiedler value trick where we put the threshold value as 0. That is after taking eigen values in nondecreasing order and corresponding eigenvectors, we map each value from the 2nd eigen vector to each data point. Values > 0 are classified into cluster1 while values <= 0 are classified into cluster 2.

Doing so, we obtain 732 data points in one cluster while 431 data points in the other cluster.

We chose to split data points using the Fiedler value trick since K-means needs high computation power.

For visualization purpose, i tried to plot the Fieldler value and got the following result,



As we can see, the points whose Fiedler value > 0 are very close to each other when compared to those < 0. Hence, I believe it is safe to assume that the genuine people are the ones which Fiedler value came out to be > 0 (732) while the comparatively spread out ones(which can be seen as outliers) with value < 0 (432) are fraudulent.

Given that in society fraudulent are always the outliers and minority(is why society still works) it would also be safe to assume that the smaller cluster is of fraudulent people, which also gives same result as the reasoning above.