भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Indian Institute of Technology Hyderabad

Fraud Analytics (CS6890)

Assignment : 3 | Identifying the outliers using spectral clustering

Name	Roll Number
Manan Darji	CS22MTECH14004
Dhwani Jakhaniya	CS22MTECH14011
Ankit Sharma	CS22MTECH12003
Vishesh Kothari	CS22MTECH12004
Jayanti Mudliar	CS22MTECH14001

Contents

1 | Problem statement

- Identify outliers in a dataset of dealers using **Spectral Clustering.**
- We have ten extracted features for each dealer, and based on the features, we are trying to identify the outlier dealers using the spectral clustering algorithm.

2 | Description of the data set

Here, the dataset for spectral clustering consists of a collection of 1199 data points, each with ten features. These data points represent the dealer's data. The features of the data points could define various attributes of the objects being analyzed, such as features of a car sale, such as car model, date of sale, and price.

To tackle the problem of Data Processing domain, the initial phase involve is to analyze and visualize the dataset.To achieve this, we utilized the pandas library's Dataframe feature to read the CSV file and examined the data points from multiple dimensions. We have considered 5-dimensions rather than 2-dimensions or 10-dimensions(There are ten features of each dealer) because visualizing data sets with 10-dimensions was difficult for humans, and visualizing with 2-dimensions will not be sufficient.

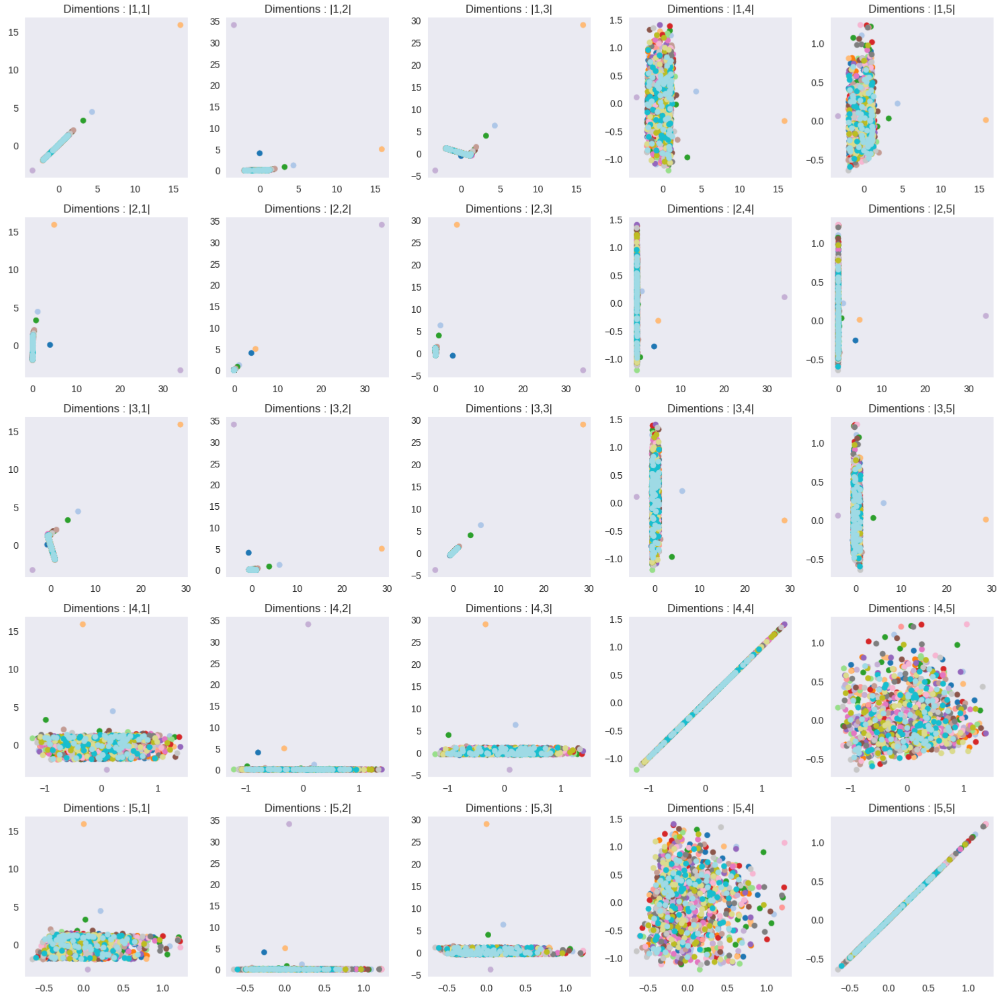


Figure 2.1: Input Data Visualization

3 | Algorithm Used

3.1 | Related Topics

3.1.1 | Spectral clustering

Spectral clustering is a technique that can identify similar data points based on their features. In this problem, the goal is to use spectral clustering to determine the sales significantly different from the rest in the dataset. Various factors, such as errors in the data, unusual market conditions, or customer behavior, could cause outliers. Identifying these outliers can provide valuable insights to the dealer and help them make informed decisions regarding their sales strategy.

3.2 | Approach we followed

Spectral clustering uses information from two critical metrics of a particular matrix: Eigen Values and Eigen Vectors. That is why we quickly converted the CSV data set into a graph and then to a matrix to cluster the dealers using spectral clustering. The following are the steps to convert the data set to LAPLACIAN MATRIX.

First, initialize a matrix of N * N size with all 0's, where N represents the number of nodes representing the number of rows in the data set. We feed the values to this matrix to make it an Adjacency matrix Mat. Now we know that we have an undirected, unweighted graph. Based on the class discussion, we calculated Cosine Similarity between all the nodes to weigh the edges and assigned that factor to the respective edge.

After each step, we visualize the output to know how the data set is clustered. So we plotted the resultant similarity matrix to visualize the clusters.

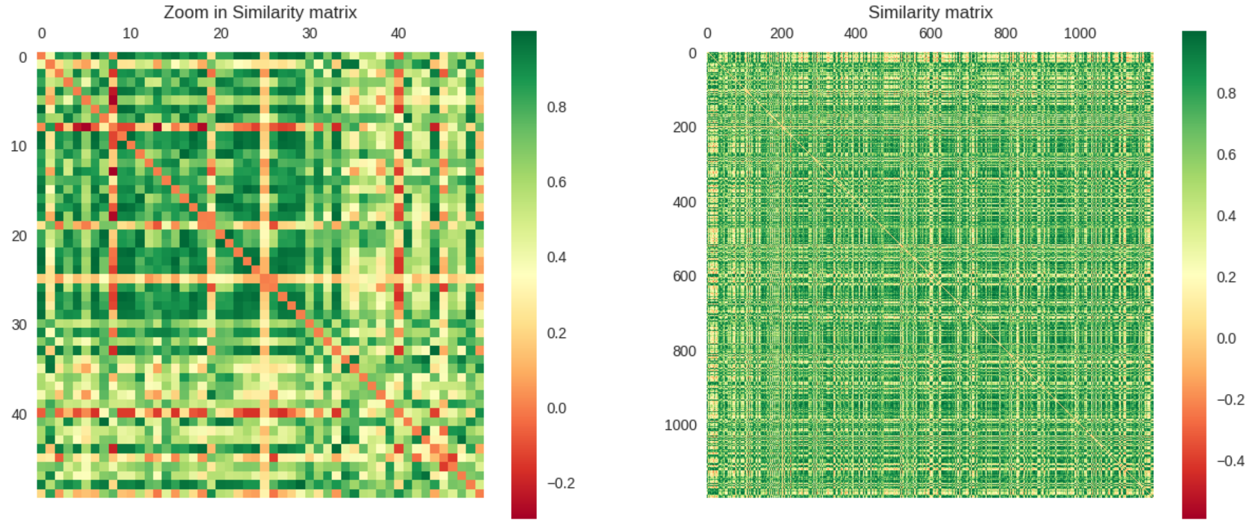


Figure 3.1: Similarity Adjacency Matrix before thresholding

The right side plot describes the similarity adjacency matrix in the above plot. On the left side, we zoom the 10*10 size matrix to properly visualize the distribution of values in the similarity adjacency matrix. The color scale beside the plots shows the value range from negative to positive. The diagonal matrix is of orange color(which means zero value) because we did not account for the similarity of nodes with themselves.

We used the numpy library to calculate cosine similarity between all the nodes and then reduce the edges by thresholding the cosine similarity factor. After hyper-tuning, we learned that 0.5 works best for better results. We made all values one, greater than 0.5, and the rest zero. We then plotted the modified similarity adjacency matrix.

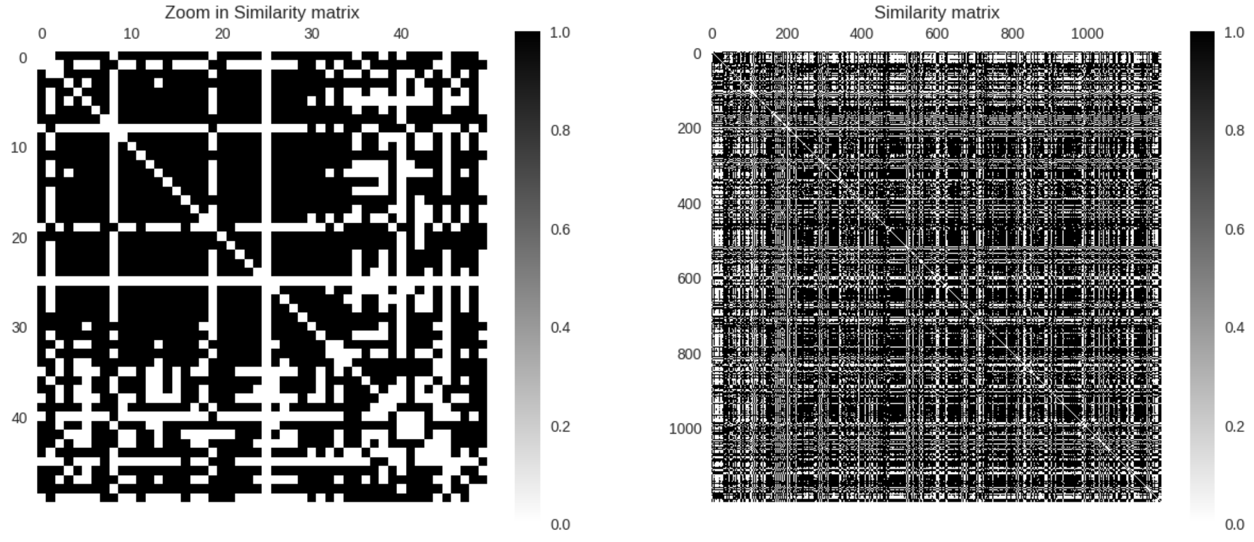
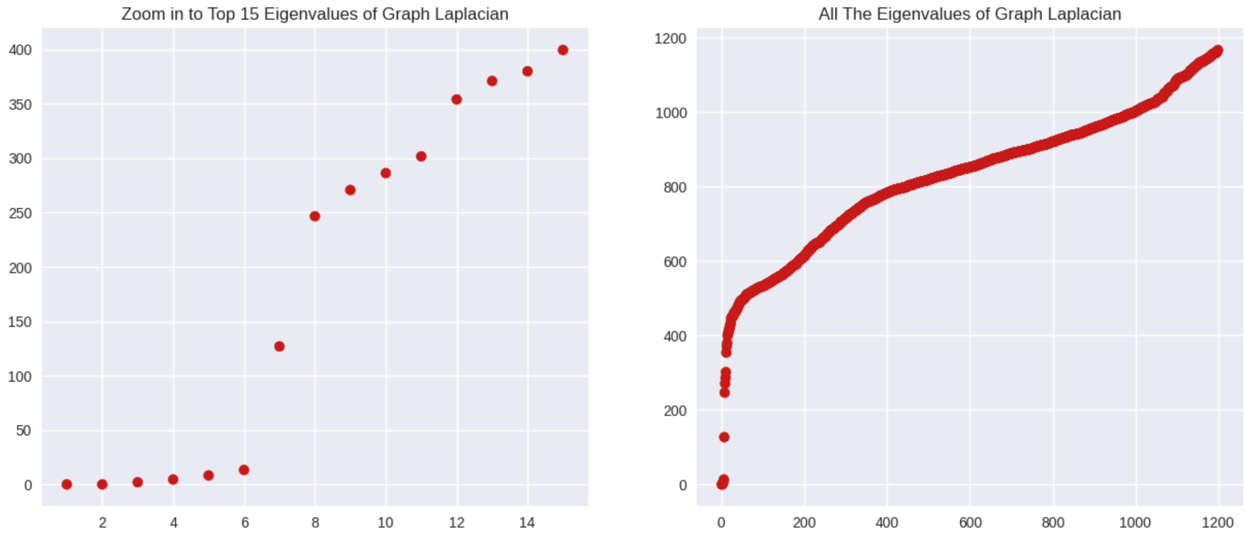


Figure 3.2: Similarity Adjacency Matrix after thresholding

As we can see from the above plots, all the values are binary (either 0 or 1). The diagonal elements are still 0. Then, we calculated Degree Matrix D. It is the diagonal matrix where the value at entry (x, x) is the degree of the node. Last, we got the LAPLACIAN MATRIX L by subtracting the adjacency matrix Mat from degree matrix D. Laplacian's diagonal is the degree of our nodes; apart from that, the negative edge weights.

Laplacian's eigenvalues tell how a graph is connected. For example, all the eigenvalues are zero when the graph is completely disconnected. After sorting the eigenvalues of the Laplacian Matrix, the first non-zero eigenvalue is known as the spectral gap. It gives some intuition of the density of the graph and how densely connected the graph is.

We calculated the eigenvalues and corresponding eigenvectors, sorted them accordingly to see the spectral gap, and got 1199 eigenvalues. We plotted all the eigenvalues in the X-Y plane to analyze those values.



-set

Figure 3.3: Eigen Value plot of Laplacian Matrix

Here we can observe the top 15 eigenvalues, and the top 6 values are almost zero, which indicates that we are close to having six separate connected components (clusters). Generally, we look for the first large gap between eigenvalues to find the number of clusters. We can also see a significant gap between (the sixth - seventh) and (seventh - eighth) eigenvalues. Having six eigenvalues before the gap indicates that there are most likely six clusters. The eigenvectors associated with these six eigenvalues contain information on clustering all the data points.

For making the clusters, we used the "K-Means algorithm" and hyper-tuned the K value from 2 to 7 (because the significant gap is after the seventh eigenvalue) to check the minimum loss while making the clusters.

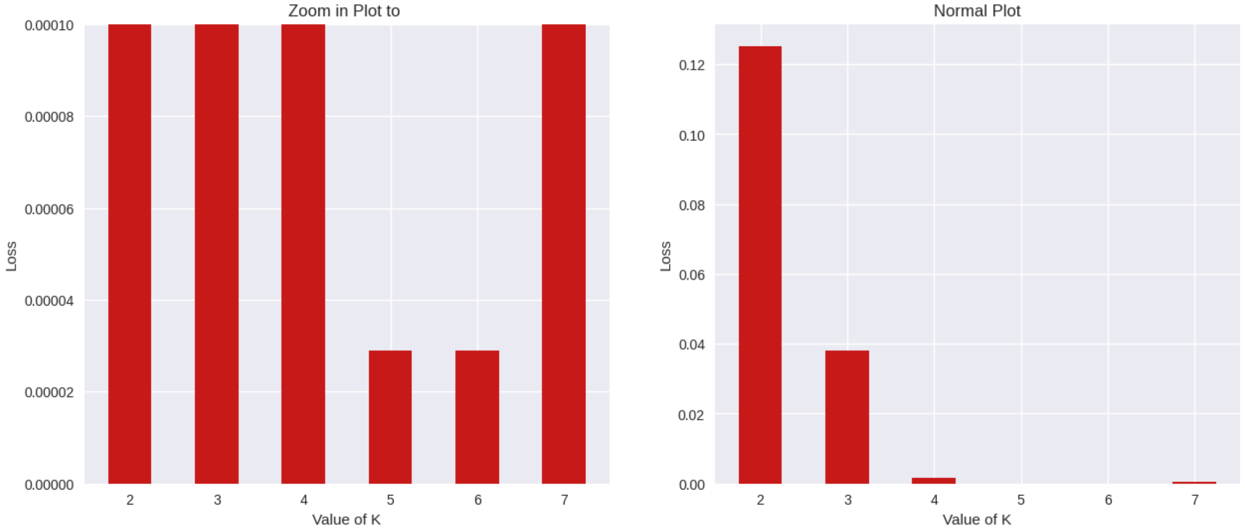


Figure 3.4: Loss plot of K means on Eigen Vectors

As we can see from the above plots, we got two values with minimum losses: five and six. We considered six as the value of K, made clusters, and plotted them with different colors using the PCA library to analyze and visualize them.

4 | Results

4.1 | Outlier's plot

We plotted these clusters in 5 dimensions to observe the outliers and usual dealers.

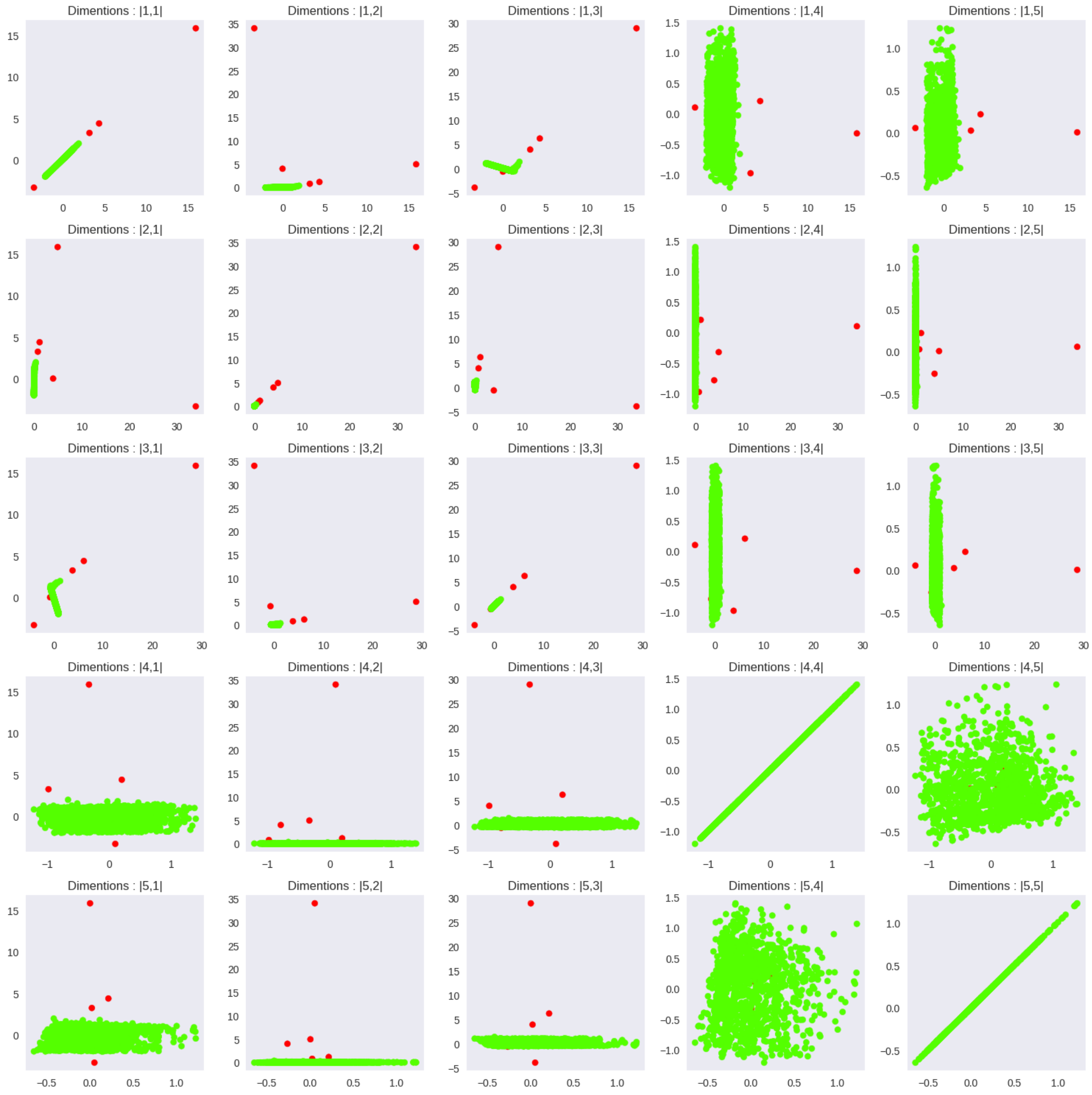


Figure 4.1: Final Clusters

From the above plots, we observed that there is a total of 6 clusters with values [1194, 1, 1, 1, 1, 1] in it. Last, we found the row numbers of dealers in the input dataset whose clusters are of size one and returned those dealers as output. And thus, the final outlier dealers are:
Total Outliers: 5
Outlier Indexs: 25, 102, 202, 249, 591