Assignment 5 (due by midnight MST the day prior to Live Session 10)

Purpose: In this homework, you will perform appropriate clustering operations on several different datasets. And to provide a rationale for your model selection and evaluation decisions. Additional questions are included to evaluate your understanding of selected clustering concepts/methods.

The homework prompts require you to exercise 'best professional judgement' in relation to your decision-making and modeling process. And to share a rationale for your approach/interpretation in clear & concise terms.

Please note that there are a wide variety of clustering algorithms available, each with various strengths and weaknesses. This homework focuses on a limited subset of common clustering algorithms: Kmeans, Gaussian Mixture Model, DBSCAN, and Agglomerative Hierarchical.

Please show code and calculations.

Learning Objectives

You should be able to:

- Apply appropriate steps to clean and/or prepare data for cluster analyses.
- Undertake exploratory data analysis to support model selection.
- Select an appropriate clustering & evaluation method(s) for a given dataset.
- Validate model results following clustering when labelled data is available.
- Discuss the strengths/weaknesses of various clustering methods.
- Leverage clustering methods for outlier detection

Problem 1a. Kmeans

Please load the following dataset: 'x1_vals.npy'

Conduct an initial exploratory data analysis (EDA) to evaluate the data. Create at least two different types of visualizations to help you evaluate possible values for K (the number of clusters). Implement two different analytical methods to narrow your choice of K prior to modeling.

Use scikit-learn to fit a basic kmeans clustering model with random initialization and reproducible results. Then create a plot of your results that distinguishes each of the clusters by color. Extract values for your cluster centroids, the number of iterations to convergence, as well as a value that serves as a measure of cluster 'coherence'.

Questions:

A. What method(s) did you use to identify an appropriate value for K? Why did you select this method? (5 pts)

B. What value did you select for K? Does your EDA support this choice? (2 pts)

C. How many iterations were required before your model converged? (2 pts)

D. What were the values for each of your cluster centroids? (2 pts)

E. What kmeans measure serves as a proxy for cluster coherence? What value did your model return? Discuss your interpretation of this value. (5 pts)

Problem 1b. Silhouette Plot

Using the data and Kmeans model from Problem 1a, create a set of silhouette plots (i.e., an appropriate range of k values) by adapting the following code for your use:

```
from sklearn.metrics import silhouette_samples, silhouette_score

# Range of values for k

range_n_clusters = [2, 3, 4, 5, 6]

plt.figure(figsize=(len(range_n_clusters) * 6, 8))

for i, n_clusters in enumerate(range_n_clusters):
    # Initialize the clusterer
    clusterer = KMeans(n_clusters=n_clusters, random_state=42)
    cluster_labels = clusterer.fit_predict(X)

    # Compute the silhouette scores
    silhouette_avg = silhouette_score(X, cluster_labels)
    sample_silhouette_values = silhouette_samples(X, cluster_labels)

    plt.subplot(1, len(range_n_clusters), i + 1)
    y_lower = 10

    for j in range(n_clusters):
        ith_cluster_silhouette_values = sample_silhouette_values[cluster_labels == j]
        ith_cluster_silhouette_values.sort()
        size_cluster_j = ith_cluster_silhouette_values.shape[0]
```

```
        y_upper = y_lower + size_cluster_j

        color = plt.cm.get_cmap("Spectral")(float(j) / n_clusters)
        plt.fill_betweenx(np.arange(y_lower, y_upper), 0, ith_cluster_silhouette_values,
facecolor=color, edgecolor=color, alpha=0.7)

        plt.text(-0.05, y_lower + 0.5 * size_cluster_j, str(j))
        y_lower = y_upper + 10

    plt.title("Silhouette plot for {} clusters".format(n_clusters))
    plt.xlabel("Silhouette coefficient values")
    plt.ylabel("Cluster label")

    plt.axvline(x=silhouette_avg, color="red", linestyle="--")
    plt.yticks([])
    plt.xticks(np.arange(-0.1, 1.1, 0.2))
    plt.xlim([-0.1, 1.0])

plt.show()
```

Note that X refers to your dataset.

Questions:

A. What information does a silhouette analysis provide? (5 pts)

B. Describe how to interpret the individual silhouette plots in terms of cluster cohesion and separation. What are the key features of the graphs that aid your evaluation? (5 pts)

C. From the silhouette plots you created, what value of K affords the best cluster assignments? (3 pts)

Problem 2. External Validation

When labels are available to distinguish groupings within a dataset, external cluster validation can be used to evaluate how well clustering results match the external criteria. In this problem, you will create a synthetic dataset with known cluster labels, implement a Kmeans model using the data, and then apply an appropriate technique for external validation.

Use sklearn's 'make_blobs' to create a synthetic dataset for clustering. Your data set should include 5 clusters with similar variance and number of observations, and 2 features. Ensure that your result is reproducible.

Use Kmeans to fit your model. And then save out predicted cluster assignments for each of your observations.

Select and employ an appropriate analytical method to assess the degree of agreement between your predicted vs. actual cluster assignments – i.e., external validation. You may need to do some research to identify the former. Create scatterplots for both predicted and actual cluster assignments using matplotlib's subplot() method.

Questions:

A. what method did you select to assess cluster agreement and why? (5 pts)

B. What do your results of this assessment suggest? (5 pts)

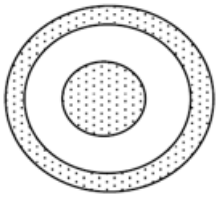
Problem 3a – Concepts: Interpreting SSE

Total SSE is the sum of the SSE for each separate attribute in the Kmeans algorithm. What does it mean if the SSE for one variable is low for all clusters? Low for just one cluster? High for all clusters? High for just one cluster? How could you use the per variable SSE information to improve your clustering? (5 pts)

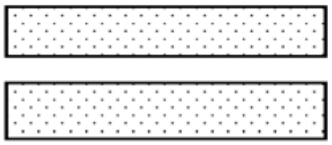
Problem 3b. Local and Global Objective Functions

K-means. For the following sets of two-dimensional points, (1) provide a sketch of how they would be split into clusters by K-means for the given number of clusters and (2) indicate approximately where the resulting centroids would be. Assume that we are using the squared error objective function. If you believe that there is more than one possible solution, then please indicate whether each solution is a global or local minimum (draw pictures to represent your responses). Darker areas indicate higher density. Assume a uniform density within each shaded area. (6 pts)

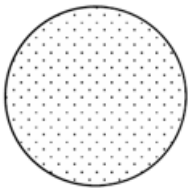
(a) $k = 3$.



(b) k = 2

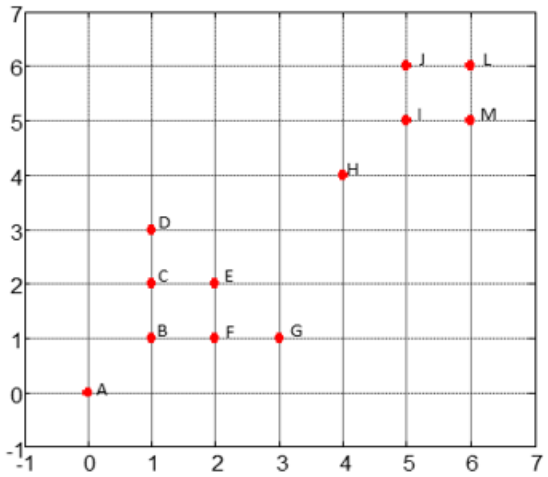


(c) k = 2



Problem 3c. Density clustering

Suppose we apply DBSCAN to cluster the following dataset using Euclidean distance.



A point is a core point if its density (num point within EPS) is ≥ MinPts. Given that MinPts =3 and EPS = , answer the following questions.

a) Label all point as ‘core points’, ‘boundary points’, and ‘noise’. (5 pts)

b) What is the clustering result (i.e., how will the data cluster)? (5 pts)

Problem 3d. Entropy vs. SSE

Assume you are given a data set of objects, each of which is assigned to one of two classes, and suppose that C1 and C2 are two clusterings produced from this data set. If entropy judges C1 to be a more accurate clustering than C2, is it necessary that SSE

will also judge C1 to be a more accurate clustering than C2? (5 pts)

Problem 4. Selecting an Appropriate Clustering Algorithm

Please import the following dataset – x4_vals.npy

Conduct an initial EDA to evaluate the data. Create at least two different types of visualizations to help you evaluate possible values for K (the number of clusters).

Parameterize and implement the following clustering algorithms using this dataset: Kmeans, Gaussian Mixture Model, DBSCAN, Agglomerative Hierarchical.

Create a plot of your results for each algorithm that distinguishes clusters by color.

Select the algorithm/parameters that you believe provides the 'best' clustering results. Note: you may find it helpful to iterate between clustering algorithms/ parameterizations and related graphs to make this selection.

Questions:

A. What method(s) did you use to identify an appropriate value for K? Why did you select this method(s)? (5 pts)

B. What value did you select for K? Does your EDA support this choice? (5 pts)

C. Why is the algorithm that you selected for your final clustering model the best choice among those you evaluated? (5 pts)

Problem 5. Comparing Algorithms

Compare the relative strengths and weaknesses of the clustering methods listed in Problem 3. Include your observations in a table form. (10 pts)

Problem 6. Dimensionality Reduction (10 pts)

Please import the following dataset – network.csv

Dimensional reduction is a standard preprocessing step before clustering for high dimensional datasets. This owes to the fact that distance measures do not work well

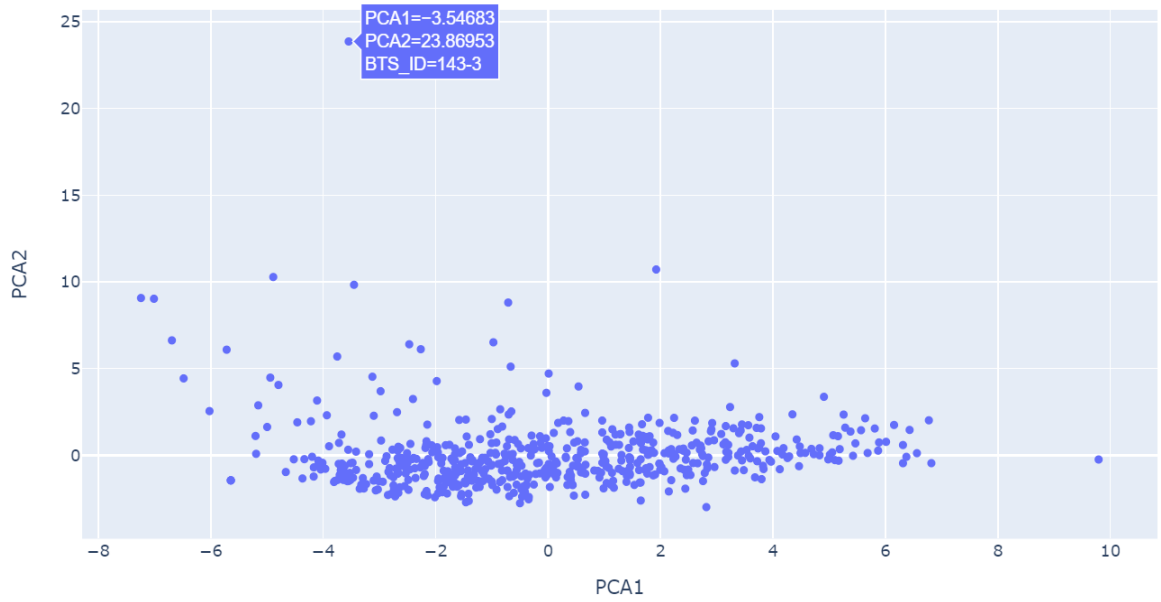
in high dimensional spaces. It's worth noting, however, that the ability to cluster data owes more to the feature signal-to-noise ratio than to the number of dimensions.

For this problem we are going to read a dataset that contains performance information from LTE cell sites operated by a rural provider. The dataset is comma delimited and contains 33 columns. The 'day' column is a string in the 2020-12-16 format (year-month-day). You will need to convert this to pandas datetime. The BTS_ID column is a unique identifier for the cell site and radio. The first number is the cell site and the number after the dash indicates the radio. Do not open and save the dataset in excel. Excel will convert all the BTS_ID's that is thinks look vaguely like a date into a date (10-1 will be converted to October 1st). The remaining columns are different performance measurements for how well the mobile phones are doing while on that site. Some measures are best when they are very small values and some are best when they are large. It would require a deep dive into the LTE cell phone standard to explain all the measurements. Some of the key values are FDD_Cell_DL_MAC_Data_Volume_MB which shows how much data was downloaded in MegaBytes. RRC_Drop_Pct is the percentage of data sessions that were dropped due to bad coverage.

It is very time consuming for the engineers in charge of this network to look at all 31 measurements for each cell site to see if there are any problems. We need to simplify the problem by doing dimensionality reduction and finding sites that are not "normal". The goal of this analysis is to find cell sites that are performing differently than the normal sites by seeing how close their performance is to all the other sites. We are going to reduce the 31 dimensions down to two dimensions and then find outliers from the main cluster.

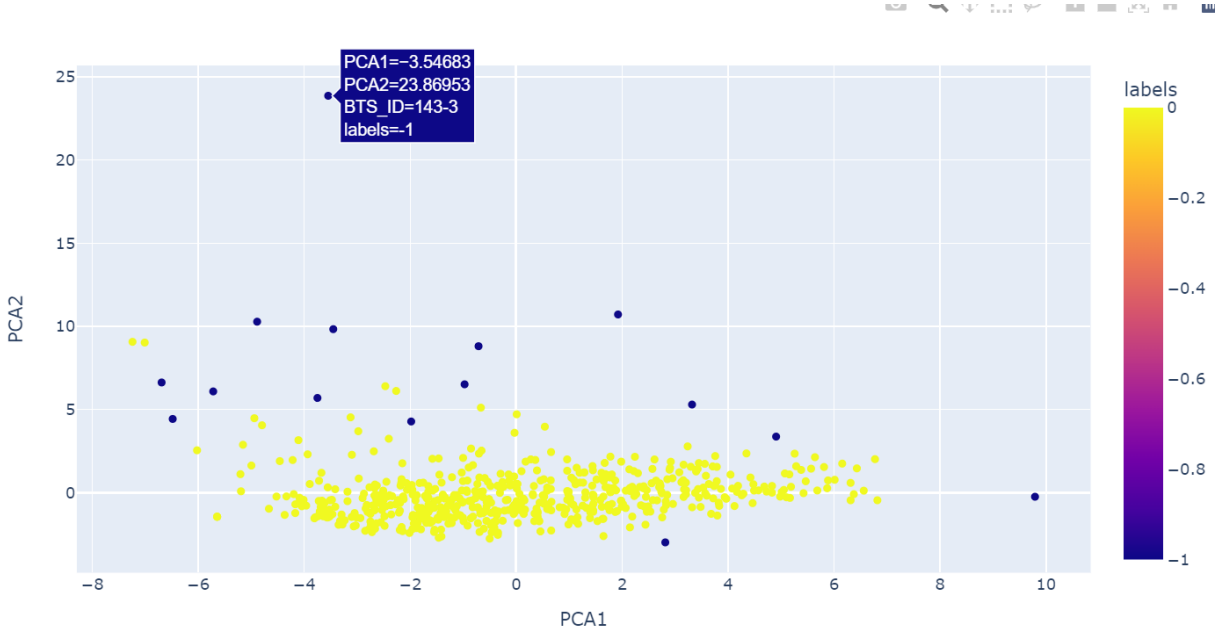
Part 1: filter the dataframe down to just the last day, 2021-06-13. Drop the day column and save the BTS_ID column to another variable. Standard scale the data and run PCA with two principal components. Join the BTS_ID data back into the PCA dataframe so it can be used with the plot. Plot the data using plotly scatter plot with the hover data showing the BTS_ID. Labeling all points in scatter plot is going to be impossible to read, so hover data allows you to read the points you want and not overlap with the others.

The plot should look like this when you hover over BTS_ID 143-3:



Part 2: Run DBSCAN on the PCA data from above to find outliers, list BTS_ID's that are outliers (the points DBSCAN calls noise). Use eps=1 and min_samples=2 to get a reasonable number of outliers. Make the labels into a column in the PCA dataframe. Next, change the labels in the dataframe to be -1 if it is an outlier and 0 if it is not. Now do a plotly plot with the color being the labels column of the dataframe so we can clearly see the outliers.

The plot should look like this when you hover over BTS_ID 143-3:



Part 3: Perform PCA on all the dates in the dataset except "2021-06-13" just like you did above with a single date. Compute the mean Euclidian distance from the center of

the plot (coordinate 0,0) for each BTS_ID. Remember each BTS_ID has measurements for each day so get the mean distance across all the days. Compute the distance from the center of the plot for each BTS_ID using the "2021-06-13" data.

Print a list of all the outlier sites are farther from the "normal", meaning center of the plot, than their mean over the last 6 months. Print a list of all the sites that are getting better (meaning heading toward the other sites performance in the center of the cluster).

The sites that are outliers and getting worse should be reported to the engineers in charge of system performance. The outliers that are getting better show the engineers are making some progress.

Are the engineers making progress with the system performance?