UNIVERSITY OF Nebraska Lincoln

Data Modeling for Systems Development

CSCE 411/811

Programming Assignment 1

Spring 2020

Cluster Based Data Analysis using K-Means, DBSCAN & GMM

Basic Info

The programming code will be graded on **both implementation and correctness**.

The written report should be based on the questions in this handout. It will be graded on content, conclusions, and presentation. It must be formatted according to the given template (posted on Canvas). The report will be graded as if the values obtained from the code portion were correct. The report should be short and to the point. The length should be between 2 to 4 pages of text plus any tables and graphs.

Assignment Goals

This assignment is intended to build the following skills:

- Cluster based data analysis using K-Means, DBSCAN and Gaussian Mixture Model clustering algorithms

---

Assignment Instructions

Note: you are not allowed to use the Scikit-Learn library or any library for building the K-Means and DBSCAN clustering models.

i. The code should be written in python in a Jupyter notebook. Use the following naming convention.
   <lastname1>_<firstname1>_<lastname2>_<firstname2>_assignment4.ipynb
ii. The Jupyter notebook should be submitted via webhandin.

Score Distribution

Part A: Create the Clustering Models (411: 40 pts & 811: 50 pts)
Part B: Cluster Based Data Analysis (411: 60 pts & 811: 80 pts)

Total: 411 (100 pts) & 811 (130 pts)

Part A: Create the Clustering Models (411: 40 pts & 811: 50 pts)

1. Write a function to compute the silhouette score. [**Extra Credit for 411 and Mandatory for 811**] [**10 pts**]

   silhouette_score (self, X, labels)

   Arguments:
   X : ndarray
   A numpy array containing samples to be used for prediction. Its rows represent data samples and columns represent features.

   labels : array
   1 D array of predicted cluster labels for each row (each sample) in X.

   Returns:
   silhouette : float
   Mean Silhouette Coefficient for all samples.

2. Implement a **KMeans** model class for performing clustering. It should have the following three methods. Note the that "fit" method should implement the **K++ Means** algorithm for initialization.
   [**20 pts**]

   a)
   fit(self, X, k, n_init, max_iter=300, tol=1e-4, distance_metric="euclidean")

   Arguments:
   X : ndarray
   A numpy array containing samples to be used for prediction. Its rows represent data samples and columns represent features.

   k : int
   The number of clusters to form as well as the number of centroids to generate.

   n_init : int

Number of times the k-means algorithm will be run with different centroid seeds. The final results will be the best output of n_init consecutive runs in terms of score (distortion measure or inertia).

max_iter : int

Maximum number of iterations of the k-means algorithm for a single run.

tol : float

Relative tolerance with regards to inertia to declare convergence.

distance_metric : string

The string value could be one of the following: “euclidean”, “manhattan”.

Returns:

1D array of predicted cluster labels for each row (each sample) in X.

b)

score(self, X)

Arguments:

X : ndarray

A numpy array containing samples to be used for prediction. Its rows represent data samples and columns represent features.

Returns:

score : float

K-Means objective function value (inertia or distortion measure). It is the sum of the squared distances between each training instance and its closest centroid

c)

__init__(self)

It’s a standard python initialization function so we can instantiate the class. Just “pass” this.

3. Implement a **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)** model class for performing clustering. It should have the following two methods.

[20 pts]

a)

fit(self, X, eps, min_samples, distance_metric="euclidean")

Arguments:

X : ndarray

A numpy array containing samples to be used for prediction. Its rows represent data samples and columns represent features.

eps : float

The maximum distance between two samples for one to be considered as in the neighborhood of the other. This is not a maximum bound on the distances of points within a cluster. This is the most important DBSCAN parameter to choose appropriately for your data set and distance function.

min_samples : int

The number of samples (or total weight) in a neighborhood for a point to be considered as a core point. This includes the point itself.

distance_metric : string

The string value could be one of the following: "euclidean", "manhattan".

Returns:

1D array of predicted cluster labels for each row (each sample) in X. Noisy samples are given the label -1.

b)

__init__(self)

It's a standard python initialization function so we can instantiate the class. Just "pass" this.

Part B: Cluster Based Data Analysis (411: 60 pts & 811: 80 pts)

You will use the **MovieLens** dataset (*movies_metadata.csv* file) for analysis. It contains information on 45,000 movies. Features include posters, backdrops, budget, revenue, release dates, languages, production countries and companies. You will use the following numeric columns for clustering: *budget, popularity, revenue, runtime, vote_average, vote_count*

Read your data as a Panda's DataFrame object.

1. Check if rows contain any null values. **[1 pts]**
2. Drop all the rows with null values. **[1 pts]**
3. Take only the movies that has more than 30 votes [vote_count > 30]. **[2 pts]**
4. Standardize the data. You may use sklearn.preprocessing.StandardScaler function. **[1 pts]**

K-Means:

5. Generate multiple clustering models by varying k. Find optimal k from the plot of score (distortion/inertia) vs k. **[5 pts]**
6. Generate multiple clustering models by varying k. Find optimal k from the plot of silhouette score vs k. [Note: 411 students may use *sklearn.metrics.silhouette_score* function to get the silhouette score] **[5 pts]**
7. Generate silhouette diagram by plotting every instance's silhouette coefficient, sorted by the cluster they are assigned to and by the value of the coefficient. You may use required *sklearn* functions. **[2 pts]**
8. Analyze the plots from above three steps and determine the optimal value of k. **[2 pts]**
9. Using the optimal value of k, perform clustering using your KMeans model. **[2 pts]**
10. Which distance metric did you use and why? **[2 pts]**
11. Interpret the clusters based on the features used for clustering. For example, how is the cluster size related to the features (revenue, vote_count, etc.)? **[4 pts]**
12. Use sklearn's KMeans model to find optimal number of clusters. You will have to use both inertia and silhouette score based analysis to determine optimal k. **[4 pts]**
13. Compare your solution with sklearn's KMeans model: (a) Report the inertia scores, (2) number of clusters and (3) algorithm running times for both models. **[4 pts]**

DBSCAN:

14. Perform clustering using your DBSCAN model. Find optimal values for the following three parameters: eps, min_samples, distance_metric

[8 pts]

15. Interpret the clusters based on the features used for clustering. For example, how is the cluster size related to the features (revenue, vote_count, etc.)?

[4 pts]

16. Use sklearn's DBSCAN model to find optimal number of clusters.

[4 pts]

17. Compare your solution with sklearn's DBSCAN model: (a) Report the eps and min_samples values, (2) number of clusters and (3) algorithm running times for both models.

[4 pts]

K-Means vs. DBSCAN

18. Compare the clusters obtained from your KMeans and DBSCAN models. Which algorithm provided better result? Justify.

[5 pts]

[**Following questions are Extra Credit for 411 and Mandatory for 811**]

19. Perform clustering using sklearn's Gaussian Mixture Model class by finding optimal number of clusters using Bayesian information criterion (BIC) and the Akaike information criterion (AIC).

[10 pts]

20. Interpret the clusters based on the features used for clustering. For example, how is the cluster size related to the features (revenue, vote_count, etc.)?

[5 pts]

21. Compare GMM clusters with KMeans and DBSCAN clusters. Based on the interpretations of the three clustering, which one you think did the best clustering and why?

[5 pts]