استاد: دکتر شاکری محمدطاها فخاریان، ۸۱۰۱۹۸۴۴۹

تمرین‌های تشریحی:

سوال ۱:

الف) برای محاسبه entropy، از فرمول زیر که در اسلایدها آمده است استفاده می‌کنیم:

$$H(C_i) = -\Sigma_{j=1}^{L} p_{ij} log(p_{ij}),\ where\ p_{ij} is\ the\ probability\ of\ an\ object\ in\ cluster\ i\ belonging\ to\ class\ j$$

پس مقدار entropy برای این خوشه‌بندی‌ها برابر است با:

$$H(C_1) = -\left(\frac{1}{693} log(\frac{1}{693}) + \frac{1}{693} log(\frac{1}{693}) + \frac{0}{693} log(\frac{0}{693}) + \frac{11}{693} log(\frac{11}{693}) + \frac{4}{693} log(\frac{4}{693}) + \frac{676}{693} log(\frac{676}{693})\right)$$
$$\rightarrow H(C_1) = 0.1998$$

$$H(C_2) = -\left(\frac{27}{1562} log(\frac{27}{1562}) + \frac{89}{1562} log(\frac{89}{1562}) + \frac{333}{1562} log(\frac{333}{1562}) + \frac{827}{1562} log(\frac{827}{1562}) + \frac{253}{1562} log(\frac{253}{1562}) + \frac{33}{1562} log(\frac{33}{1562})\right)$$
$$\rightarrow H(C_2) = 1.8405$$

$$H(C_3) = -\left(\frac{326}{949} log(\frac{326}{949}) + \frac{465}{949} log(\frac{465}{949}) + \frac{8}{949} log(\frac{8}{949}) + \frac{105}{949} log(\frac{105}{949}) + \frac{16}{949} log(\frac{16}{949}) + \frac{29}{949} log(\frac{29}{949})\right)$$
$$\rightarrow H(C_3) = 1.6964$$

همچنین مقدار entropy برای کل خوشه‌بندی برابر است با میانگین وزن‌دار entropyها:

$$H(C) = \frac{693}{3204} * 0.1998 + \frac{1562}{3204} * 1.8405 + \frac{949}{3204} * 1.6964 = 1.4429$$

ب) برای محاسبه purity کل داریم:

$$purity = \Sigma_{i=1}^{r} \frac{n_i}{n} purity_i = \frac{1}{n} \Sigma_{i=1}^{r} max_{j=1}^{k} \{n_{ij}\}$$

پس مقدار purity کل برای این خوشه‌بندی برابر است با:

$$purity = \frac{1}{3204}(676 + 827 + 465) = \frac{1968}{3204} = 0.61423$$

همچنین مقدار purity برای هر خوشه برابر است با:

$$purity_1 = \frac{676}{693} = 0.975$$
$$purity_2 = \frac{827}{1562} = 0.529$$
$$purity_3 = \frac{465}{949} = 0.489$$

ج) برای محاسبه این سه مقدار داریم:

$$prec_i = \frac{1}{n_i} max^{k}_{j=1}\{n_{ij}\} = \frac{n_{ij_i}}{n},\ recall_i = \frac{n_{ij_i}}{|T_{ij}|},\ F_i = \frac{2*recall_i*prec_i}{recall_i+prec_i},\ F = \frac{1}{r}\Sigma^{r}_{i=1}F_i$$

پس داریم:

Cluster	Precision	Recall	F-measure
#1	$\frac{676}{693} = 0.975$	$\frac{676}{738} = 0.915$	0.944
#2	$\frac{827}{1562} = 0.529$	$\frac{827}{943} = 0.876$	0.659
#3	$\frac{465}{949} = 0.489$	$\frac{465}{555} = 0.837$	0.617

و مقدار F کل نیز برابر است با:

$$F = \frac{F_1+F_2+F_3}{3} = 0.74$$

سوال ۲:

الف) از تعریف core-point استفاده می‌کنیم:

A data point p is a *core point* if **Nbhd**(p,$\mathcal{E}$) [$\mathcal{E}$-neighborhood of p] contains at least *minPts* ; |**Nbhd**(p,$\mathcal{E}$)| >= *minPts*.

طبق این تعریف، کافیست تعداد همسایگی‌های تمام دانش آموزان را چک کنیم. پس از چک کردن، به این دانش آموزان می‌رسیم(عدد کنار هر نقطه، تعداد همسایگی‌های آن نقطه(به همراه خودش) است):

C(6), E(6), F(6), H(5), I(4), L(4)

ب) از تعریف border-point استفاده می‌کنیم:

A data point q is a *border point* if **Nbhd**(q, $\mathcal{E}$) contains less than *minPts* data points, but q is *reachable* from some *core point* p.

طبق این تعریف، به دانش‌آموزان زیر می‌رسیم:

A, B, G, K, O, M

ج) با توجه به اینکه I یک core-point است، دانش‌آموزان E و F و H به طور مستقیم از I قابل دسترسی هستند.

د) چون M یک core-point نیست، هیچ دانش‌آموزی به طور مستقیم از آن قابل دسترسی نیست.

ه) در صورتی که در ردیف ۵ و ستون ۳ بنشیند، core-point نخواهد بود ولی از دو core-point قابل دسترسی است(H و L) که باعث می‌شود دو خوشه باهم ادغام نشوند ولی به دو خوشه متصل باشد.

و) در صورتی که در ردیف ۶ و ستون ۴ بنشیند، به core-point تبدیل می‌شود و لذا باعث می‌شود که دانش‌آموزان دو خوشه به هم متصل شوند و باعث ادغام دو خوشه شود.

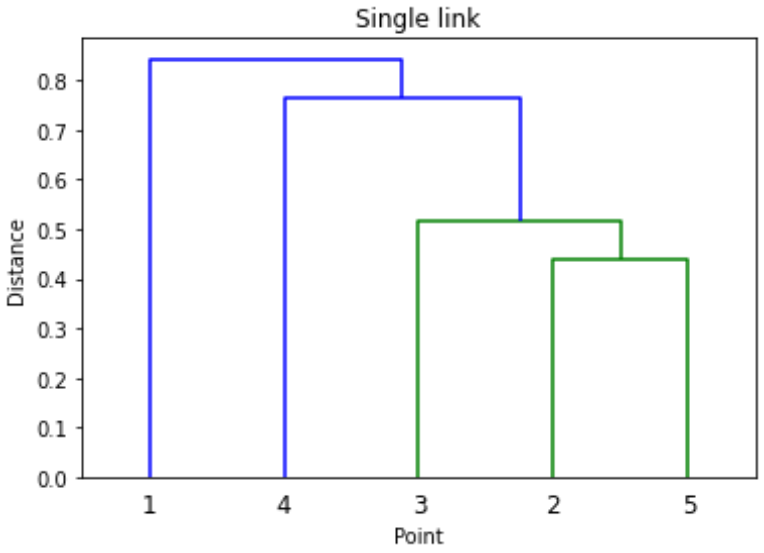
سوال ۳:

در الگوریتم Agglomerative، بسته به روش استفاده شده، در هر مرحله دو خوشه با کمترین فاصله(یا بیشترین شباهت) باهم ادغام می‌شوند. حال به هر دو روش می‌پردازیم:

روش single-link:

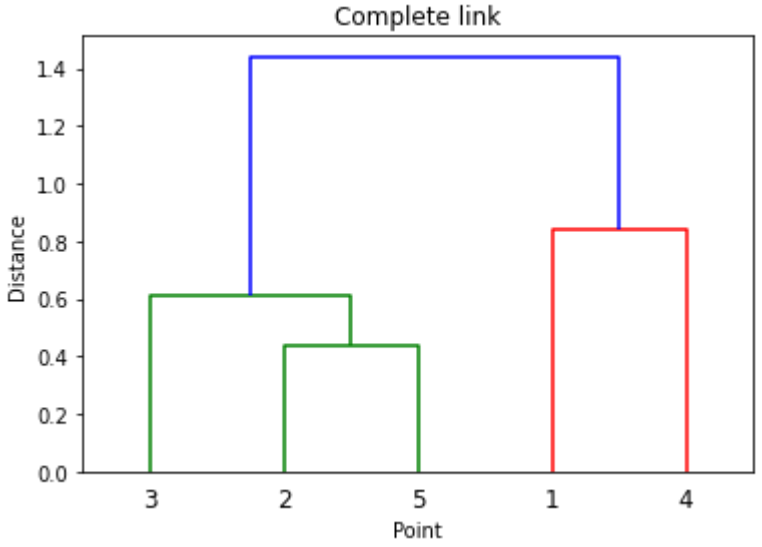
در این روش، کوتاهترین فاصله بین نقاط دو خوشه به عنوان فاصله دو خوشه از هم در نظر گرفته می‌شود. طبق جدول شباهت داده شده، ابتدا دو خوشه {p2} و {p5}(با شباهت 0.98) باهم ادغام می‌شوند، سپس دو خوشه {p3} و {p2, p5}(با شباهت 0.85) باهم ادغام می‌شوند، سپس دو خوشه {p4} و {p2, p5, p3}(با شباهت 0.85) باهم ادغام می‌شوند و نهایتاً p1 به بقیه می‌پیوندد.

برای رسم نمودار، به جای شباهت از فاصله استفاده کردیم که برابر است با D=1-S.



روش complete-link:

در این روش، بیشترین فاصله بین نقاط دو خوشه به عنوان فاصله دو خوشه از هم در نظر گرفته می‌شود. طبق جدول شباهت داده شده، ابتدا دو خوشه {p2} و {p5}(با شباهت 0.98) باهم ادغام می‌شوند، سپس دو خوشه {p3} و {p2, p5}(با شباهت 0.64) باهم ادغام می‌شوند، سپس دو خوشه {p1} و {p4}(با شباهت 0.55) باهم ادغام می‌شوند و نهایتاً دو خوشه {p1, p4} و {p2, p5, p3} باهم ادغام می‌شوند.



تمرین‌های عملی:

In this computer assignment, we are going to explore a dataset related to diabetic patients for clinics and hospitals in America, do some preprocessing and apply two clustering methods(KMeans and DBSCAN) on it.

A:

Since we want to prepare data for KMeans and DBSCAN clustering methods, we need to know the requirement for these two methods:

K-means input data requirements:

- Numerical variables only. K-means uses distance-based measurements to determine the similarity between data points. If you have categorical data, use K-modes clustering, if data is mixed, use K-prototype clustering.
- Data has no noises or outliers. K-means is very sensitive to outliers and noisy data.
- Data has symmetric distribution of variables (it isn't skewed). Real data always has outliers and noise, and it's difficult to get rid of it. Transformation data to normal distribution helps to reduce the impact of these issues. In this way, it's much easier for the algorithm to identify clusters.
- Variables on the same scale — have the same mean and variance, usually in a range -1.0 to 1.0 (standardized data) or 0.0 to 1.0 (normalized data). For the ML algorithm to consider all attributes as equal, they must all have the same scale.

- There is no collinearity (a high level of correlation between two variables). Correlated variables are not useful for ML segmentation algorithms because they represent the same characteristic of a segment. So correlated variables are nothing but noise.
- Few numbers of dimensions. As the number of dimensions (variables) increases, a distance-based similarity measure converges to a constant value between any given examples. The more variables the more difficult to find strict differences between instances.

According to these explanations, we first check whether there is any NaN value in the columns or not. There are some values with '?' in the columns. First, we'll drop the 'patient_nbr' column, since it acts as an ID and is not a good feature. Then, we'll replace '?' with NaN values to see the number of unknown values in the dataset. Here is the result of NaN counts after this transformation:

encounter_id	0
race	2273
gender	0
age	0
weight	98569
admission_type_id	0
discharge_disposition_id	0
admission_source_id	0
time_in_hospital	0
payer_code	40256
medical_specialty	49949
num_lab_procedures	0
num_procedures	0
num_medications	0
number_outpatient	0
number_emergency	0
number_inpatient	0
diag_1	21
diag_2	358
diag_3	1423
number_diagnoses	0
max_glu_serum	0
A1Cresult	0

metformin	0
repaglinide	0
nateglinide	0
chlorpropamide	0
glimepiride	0
acetohexamide	0
glipizide	0
glyburide	0
tolbutamide	0
pioglitazone	0
rosiglitazone	0
acarbose	0
miglitol	0
troglitazone	0
tolazamide	0
examide	0
citoglipton	0
insulin	0
glyburide-metformin	0
glipizide-metformin	0
glimepiride-pioglitazone	0
metformin-rosiglitazone	0
metformin-pioglitazone	0
change	0
diabetesMed	0
readmitted	0

According to this stats, we'll drop the 'weight', 'payer_code' and 'medical_specialty' columns, since about half of values in these columns are NaN. Then, we'll drop rows that have NaN values in their columns(remaining columns with NaN values have at most about 2000 rows, which consider nothing in about 100000 records.

Now it's time to deal with Non-numeric columns. We've plotted the bar plot for each column to see how many distinct values each column has. According to these columns, since there is a meaningful order in non-numeric columns, we'll use OrdinalEncoder to make them numeric.

Now, we'll extract encounter_id from the data for saving the final results. After that, we'll drop the columns in the data that have only one unique value. This

is done because these columns don't have additional information for us and it's necessary to drop them, in order to use z-score for noise elimination.

Now, we'll perform a normalizing(Min-Max Scaling) on the data to make scales similar to each other(between 0 to 1).

Now, it's time to eliminate probable noise from data. We use z-score for this purpose to find outliers and remove them from data.

Finally, we'll perform dimensionality reduction on the data using PCA. As the number of dimensions (variables) increases, a distance-based similarity measure converges to a constant value between any given examples. The more variables the more difficult to find strict differences between instances. Checking different number of dimensions ranging from 2 to 6, we decided to reduce dimensions to 2. In this way, we can visualize clusters better too.

Here is the final, preprocessed data:

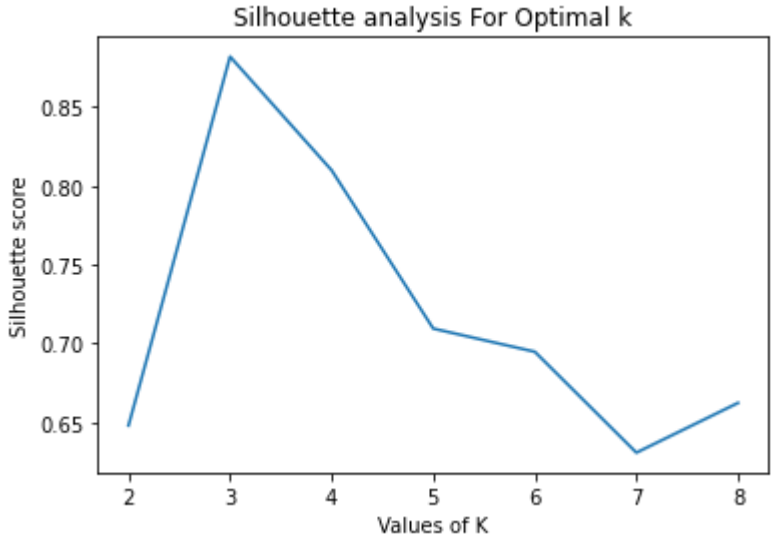
	0	1
0	-0.077694	0.333543
1	0.727273	-0.132537
2	0.700082	-0.223289
3	-0.007512	0.513120
4	-0.025141	0.439148
...	...	...
61330	0.715498	-0.112474
61331	0.652423	-0.349447
61332	-0.032742	0.449004
61333	0.604609	-0.436329
61334	-0.765449	-0.293982

61335 rows × 2 columns

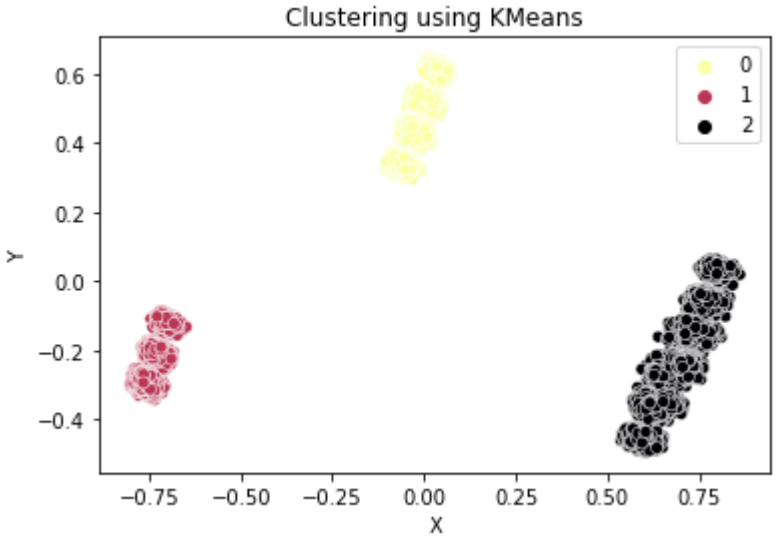
B:

Now let's apply clustering methods.

Let's see the results using KMeans method:



It seems that best number of clusters for KMeans is 3. Here is the result of clustering using KMeans:



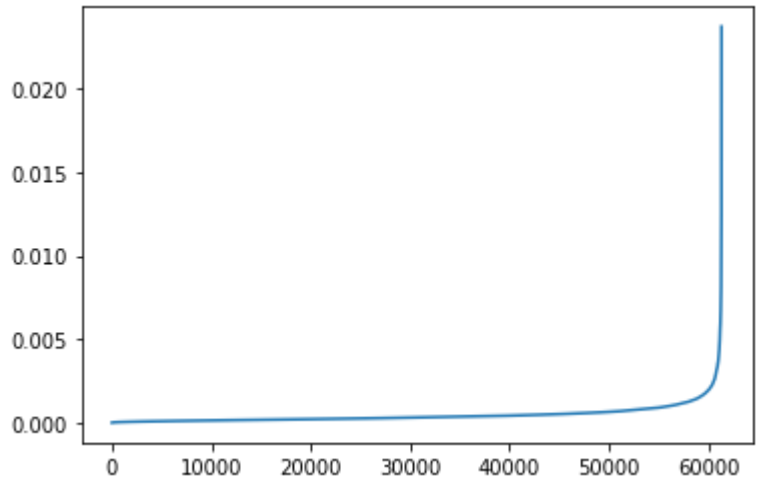
Now let's try DBSCAN method.

There is no automatic way to determine the MinPts value for DBSCAN. Ultimately, the MinPts value should be set using domain knowledge and familiarity with the data set. From some research I've done, here are a few rules of thumb for selecting the MinPts value:

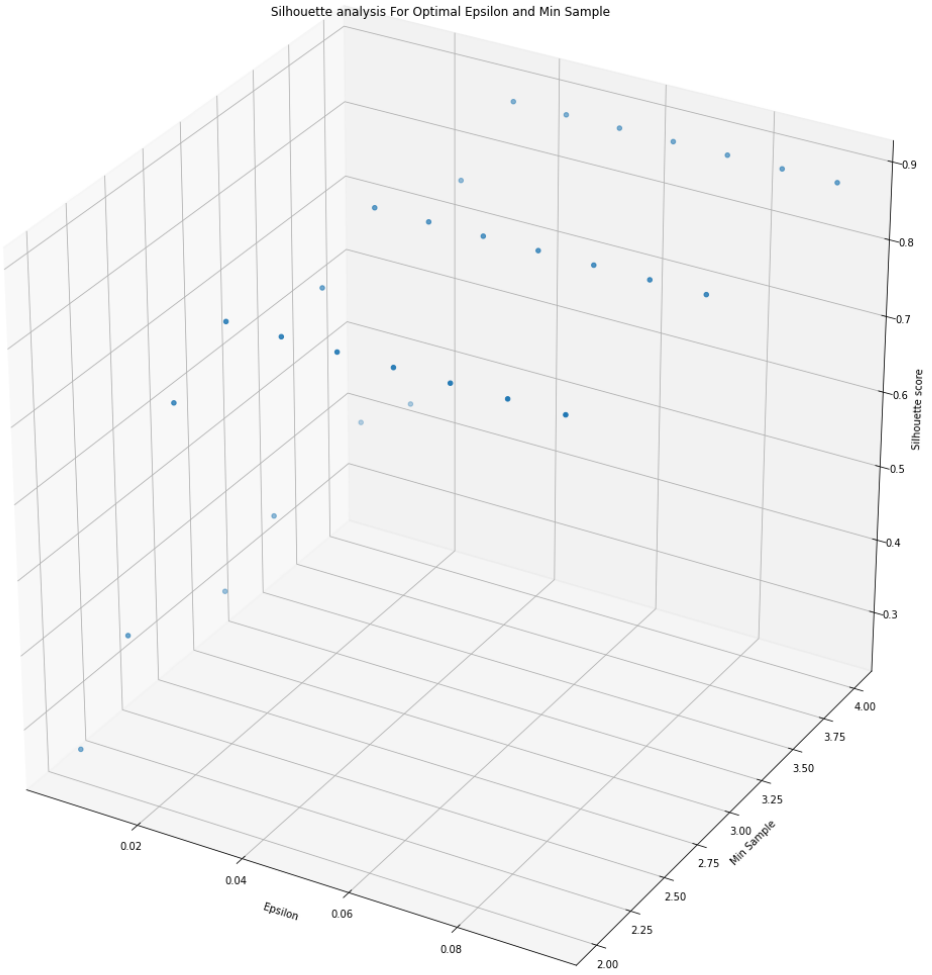
- The larger the data set, the larger the value of MinPts should be
- If the data set is noisier, choose a larger value of MinPts
- Generally, MinPts should be greater than or equal to the dimensionality of the data set
- For 2-dimensional data, use DBSCAN's default value of MinPts = 4 (Ester et al., 1996).
- If your data has more than 2 dimensions, choose MinPts = 2*dim, where dim= the dimensions of your data set (Sander et al., 1998).

After you select your MinPts value, you can move on to determining Ɛ. One technique to automatically determine the optimal Ɛ value. This technique calculates the average distance between each point and its k nearest neighbors, where k = the MinPts value you selected. The average k-distances are then plotted in ascending order on a k-distance graph. You'll find the optimal value for Ɛ at the point of maximum curvature (i.e. where the graph has the greatest slope).

According to these explanations and since we have 2 dimensions, we'll use KNN with k = 4 and find "crook of the elbow" in the plot to find the starting range of epsilon. Here is the elbow plot:



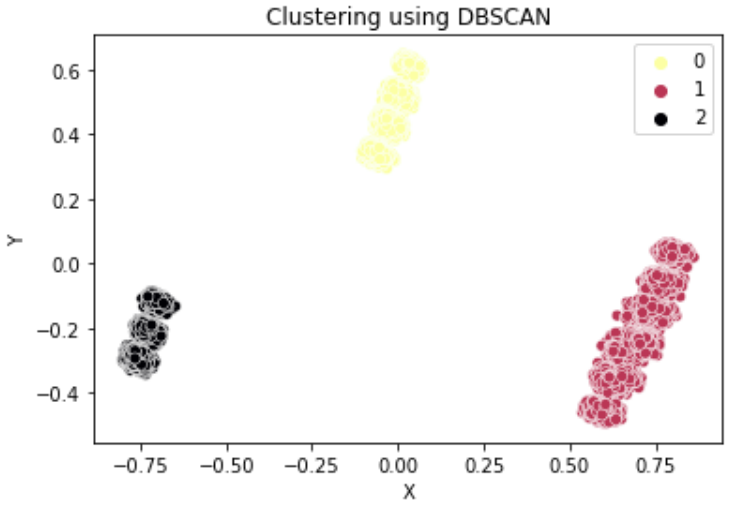
So we'll use 0.005 to 0.1 for epsilon and 2 to 5 for min points. Here is the results using these ranges:



Here are the best parameters for DBSCAN:

score	eps	min_sample
0.882074	0.055	2.0

And here is the result of clustering using DBSCAN:



Finally, we'll save the results of labels in a csv file.

Conclusion

- Kmeans clustering does not do a good job when the data is not spherically distributed. As the dimensionality increases, it gets worse.
- DBSCAN looks like a good solution. However, it is very sensitive to the minimum number of points required to form a dense region and the maximum radius of the neighborhood from a point.