A20442087
Tiffany Wong

Assignment 2

CS484 Spring 2021
February 20, 2021

Question 1

- Answer

$Lift(\{B, D\} \rightarrow \{A, C, E\}) = 100\% = 1$

- Explanation

It's given that {A, B, C, D, E} has a Support value of 1, and it means that the probability of the entire itemset occurring is 100%. In the same vein, this means that the probability of any individual item occurring in a transaction is also 100% (=1). Since the probability of {B, D} occurring does not affect the probability of {A, C, E} happening, that means that the Lift value for the statement $\{B, D\} \rightarrow \{A, C, E\}$ is 1.

Question 2

- Answer

$Lift(\{Cheese,\ Wings\} \rightarrow \{Soda\}) = 1.125$

- Figure

Size	Count	Support	Food Item 1	Food Item 2
1	4	66.67%	Soda	
2	3	50%	Cheese	Wings

- Explanation

Support for {Cheese, Wings}:
Occurred in friends: Andrew, Carl, Frank
Equals 3/6 = 50%

$Support\ (Soda \mid Cheese, Wings)$:
Occurred in friends: Andrew, Carl, Frank
Equals 3/3 = 100%

Support for {Soda}:
Occurred in friends Andrew, Betty, Carl, Frank
Equals 4/6 = 66.67%

The Lift of this association rule {Cheese, Wings} ==> {Soda} is the Confidence divided by the Expected Confidence, where $Confidence = Support(Soda \mid Cheese, Wings)$ and $Expected\ Confidence = Support(Soda)$.

Solving for the Confidence and Expected Confidence gives me: $Confidence = 100\%$ and $Expected\ Confidence = 66.67\%$.
Then the Lift is $Confidence / Expected\ Confidence = 100\% / 66.67\% = 150\% = \frac{3}{2}$.

Question 3

- Answer

(C) Close to one

- Explanation

The formula for a Silhouette value is the average of the Silhouette width of all observations. The Silhouette width is the difference between the minimum of the average distance of an observation x_{ij} to its nearest neighboring cluster, b_{ij}, and the average distance between the observations x_{ij} and all other observations in the same cluster, a_{ij}, divided by the maximum of the previously mentioned quantities, $\max(a_{ij}, b_{ij})$.

With the given scatterplot, the three clusters are practically at the corners of an equilateral triangle, where each cluster consists of points that are about the same distance apart from the center of each cluster.

The big visual takeaway is that the three clusters are fairly far apart from each other, while the observations in each cluster are relatively close to each other. This means that b_{ij} is a big number compared to a_{ij}, so the difference between the two would result in a number close to b_{ij} and since the maximum is also b_{ij} (since it's bigger), the formula would essentially reduce down to b_{ij} dividing by itself. Therefore, the Silhouette value would be option (C) close to one.

Question 4a

- Answer

The Silhouette Width of the observation 2 (i.e., the value -1) in Cluster 0 is 0.6428571428571429.

- Code

```
#%% question 4a (for loop method)
# Calculate the Silhouette Width of the observation 2 (i.e., the value -1)
# in Cluster 0.

import math
import numpy as np

# a: The mean distance between a sample and all other points in the same class.
# b: The mean distance between a sample and all other points in the next nearest cluster.

X = [-2, -1, 1, 2, 3]
Y = [4, 5, 7, 8]

a_all = []
b_all= []

for i in X:
    a_all.append(abs(X[1]-i))

a = sum(a_all)/(len(a_all)-1)

for j in Y:
    b_all.append(abs(X[1]-j))

b = np.mean(b_all)

silhouette_value = (b-a)/(max(a,b))

print(silhouette_value)
```

- Explanation

I initially tried to implement the silhouette.score() function that is built into the sklearn module, but I wasn't about to figure out what the labels parameter would be in this situation because the clusters are already given to us. I wrote a for loop that subtracts the value -1 from each element of Cluster 0 and then did the same for Cluster 1. Then I used the mean() function from the numpy module to find the average distance in each array and used the amax() function to find the maximum between the two average distance quantities. Next, I used the formula for Silhouette Width, $\frac{(b-a)}{max(a, b)}$ and plugged in the values I found into it to find the Silhouette Width for the observation 2 in cluster 0.

Question 4b

- Answer

The cluster-wise Davies-Bouldin value of Cluster 0 is 1.68 and Cluster 1 is 1.5. The R-value is 0.5889.

- Code

```
#%% question 4b (cluster-wise Davies-Bouldin value)
# Calculate the cluster-wise Davies-Bouldin value of Cluster 0 (i.e., R_0) and
# Cluster 1 (i.e., R_1).

from sklearn.neighbors import NearestCentroid

# intra-cluster is distances between other pts in the same cluster: a_all
# inter-cluster is distances to other pts in diff cluster: b_all

# array of observations
X = [-2, -1, 1, 2, 3]
Y = [4, 5, 7, 8]

# array of distances
X_distances = []
Y_distances = []

# cluster centroids
# centroid for X is 1
centroid0 = sum(X)/len(X)
# centroid for Y is between 5 and 7
centroid1 = sum(Y)/len(Y)

# distances for X
for i in X:
    X_distances.append(abs(i-centroid0))

# distances for Y
for i in Y:
    Y_distances.append(abs(i-centroid1))

# intra cluster dist for X
s_k_x = (1/(len(X)))*(sum(X_distances))

# intra cluster dist for Y
s_k_Y = (1/(len(Y)))*(sum(Y_distances))

print('cluster 0:', s_k_x, 'cluster 1:', s_k_Y)
```

- Explanation

To find the cluster-wise Davies-Bouldin value, I went through the formula for Intra-Cluster Distance from the slides.

Question 4c

- Answer

The Davies-Bouldin Index of this two-cluster solution is 0.2944.

- Code

```
#%% question 4c (Davies-Bouldin index)
m_kl = abs(centroid0 - centroid1)

r_kl = (s_k_x + s_k_Y) / m_kl

print(r_kl / 2)
```

- Explanation

I continued the formula for Davies-Bouldin index by dividing the R-value I got previously from 4(b) by 2 because that's the number of clusters. That gives me the respective 0.2944.

Question 5a

- Answer

We can find a total of 524 itemsets and the largest k-value among our datasets is k=4.

- Code

```
#%% question 5a

import pandas as pd
from mlxtend.preprocessing import TransactionEncoder

GroceriesAssoc = pd.read_csv('/Users/tiffwong/Desktop/cs484/assignments/assignment 2/Groceries.csv',
                             delimiter=',')

customers = max(GroceriesAssoc['Customer'])

# Convert the Groceries data to the Item List format
ListItem = GroceriesAssoc.groupby(['Customer'])['Item'].apply(list).values.tolist()

# Convert the Item List format to the Item Indicator format
te = TransactionEncoder()
te_ary = te.fit(ListItem).transform(ListItem)
ItemIndicator = pd.DataFrame(te_ary, columns=te.columns_)

from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

# Find the frequent itemsets
frequent_itemsets = apriori(ItemIndicator, min_support = 75 / len(ListItem),
                            max_len = None, use_colnames = True)

itemsets = len(frequent_itemsets)
print(f"{itemsets} itemsets")
largest_k = len(frequent_itemsets['itemsets'][itemsets - 1])
print(f"Largest k: {largest_k}")
```

- Explanation

I start by reading in the Groceries.csv and then converting the data into Item List format. Then I convert the Item List format into the Item Indicator format. Next, I used the Apriori function from the mlxtend library to find the frequent itemsets and found the number of itemsets in the frequent_itemsets. That's how I found the amount of itemsets in total we are able to find and the largest k-value among those itemsets.

Question 5b

- Answer

There are 1228 association rules that can be found when Confidence metrics are greater than or equal to 1%.

- Code

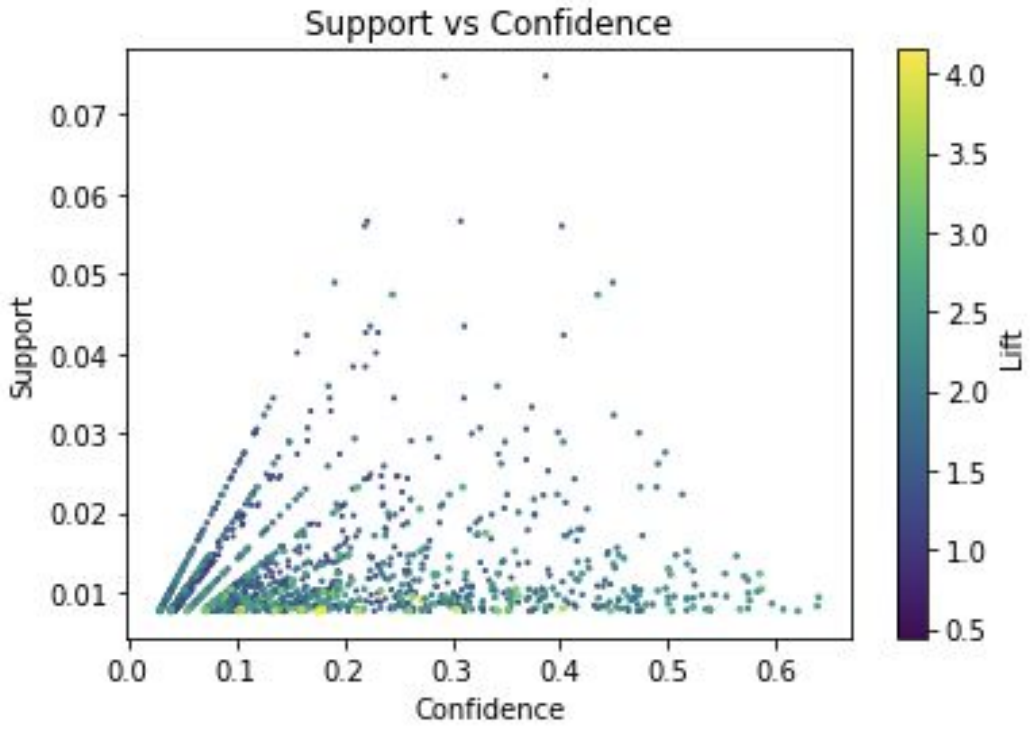
```
#%% question 5b
# Discover the association rules
assoc_rules = association_rules(frequent_itemsets, metric = "confidence", min_threshold = 0.01)
```

- Explanation

I used the association_rules function from the mlxtend.frequent_patterns library to put into the frequent_itemsets data I got from (5a), set the metric as ‘confidence’, and put the min_threshold as 0.01, because the problem asks for the the association rules when the Confidence metrics are greater than or equal to 1%.

Question 5c

- Answer/Figure



- Code

```
#%% question 5c
print("---QUESTION 5C---")

import matplotlib.pyplot as plt

plt.figure(figsize=(6,4))
plt.scatter(assoc_rules['confidence'], assoc_rules['support'], \
            c = assoc_rules['lift'], s = assoc_rules['lift'])
plt.grid(True)
plt.xlabel("Confidence")
plt.ylabel("Support")
plt.title("Support vs Confidence")
colorgradient = plt.colorbar()
colorgradient.set_label("Lift")
plt.show()
```

- Explanation

To plot the Support metrics against the Confidence metrics, I used the scatter function in the matplotlib library. I labeled the axes according to the instructions from the question (assoc_rules's confidence against assoc_rules's support) and included a color gradient legend for the Lift metrics.

Question 5d

- Answer/Figure

antecedent	consequent	Antecedent support	Consequent support	support	confidence	lift	leverage	conviction
(butter, root vegetables)	(whole milk)	0.012913	0.255516	0.008236	0.637795	2.496107	0.004936	2.055423
(yogurt, butter)	(whole milk)	0.014642	0.255516	0.009354	0.638889	2.500387	0.005613	2.061648
(yogurt, other vegetables, root vegetables)	(whole milk)	0.012913	0.255516	0.007829	0.606299	2.372842	0.004530	1.890989
(yogurt, other vegetables, tropical fruit)	(whole milk)	0.012303	0.255516	0.007626	0.619835	2.425816	0.004482	1.958317

- Code

```
#%% question 5d
print("---QUESTION 5D---")

assoc_rules2 = association_rules(frequent_itemsets, metric = "confidence", min_threshold = 0.6)
```

- Command Output

antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
(butter, root vegetables)	(whole milk)	0.012913	0.255516	0.008236	0.637795	2.496107	0.004936	2.055423
(yogurt, butter)	(whole milk)	0.014642	0.255516	0.009354	0.638889	2.500387	0.005613	2.061648
(yogurt, other vegetables, root vegetables)	(whole milk)	0.012913	0.255516	0.007829	0.606299	2.372842	0.004530	1.890989
(yogurt, other vegetables, tropical fruit)	(whole milk)	0.012303	0.255516	0.007626	0.619835	2.425816	0.004482	1.958317

- Explanation

I used the association_rules function in the mlxtend.frequent_patterns library to find the association rules when the Confidence metrics are greater than or equal to 60%.

Question 6a

- Answer/Command Prompt

N Clusters	Elbow	Silhouette	Calinski_Harabasz	Davies_Bouldin
2	6.7306	0.4635	454.6607	0.8405
3	7.5802	0.4219	465.6973	0.8275
4	9.4943	0.3932	436.9141	0.8836
5	9.3369	0.3290	411.1514	0.9710
6	9.5308	0.3332	402.2028	0.9744
7	9.5114	0.3486	412.7620	0.9410
8	9.6593	0.3262	397.4171	0.9491
9	10.2850	0.3305	379.2098	0.9603
10	10.1014	0.3301	365.3021	0.9415

- Explanation

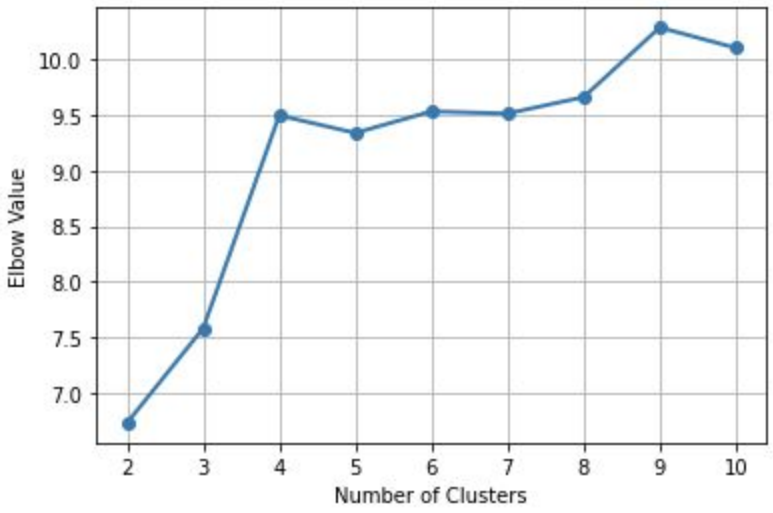
Using the sklearn module, I import the functions for the metrics: silhouette_score, calinski_harabasz_score, davies_bouldin_score. I used those functions to calculate the Silhouette Values, Calinski Harabasz Scores, and Davies Bouldin Indices for each of the number of clusters I had to find to, where the minimum is 2 clusters and the maximum is 10 clusters. Then to find the Elbow Values, I used a for loop to find the WCSS values and then calculated it according to the formula of Elbow Values.
For clarification, I'm going to

Question 6b

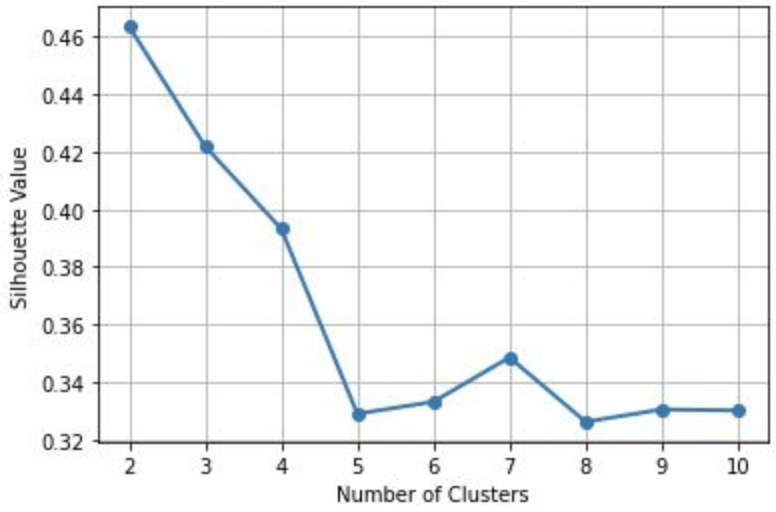
- Answer

Based on the values in (a), my suggested number of clusters is 3 clusters.

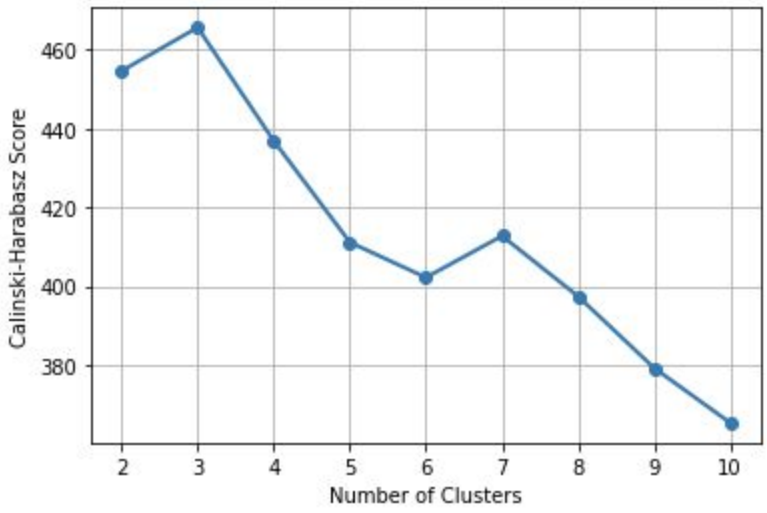
- Figure



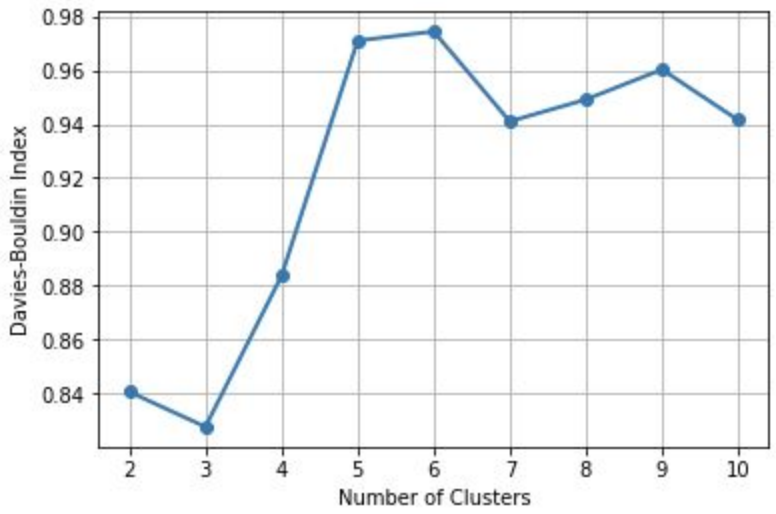
(1)



(2)



(3)



(4)

- Explanation

The way to find the recommended number of clusters from each graph is different, based on the value being used. Starting with the Elbow Value graph, the value that elbows the data visually would ideally be the number of clusters, in this case, there isn't a great Elbow Value, so it's either 2 or 3 clusters at this point. Then with the Silhouette Value graph, the value that peaks in the data visually would ideally be the number of clusters, in this case, the data decreases from the beginning, so it's either 2 or 3 clusters at this point. Next with the Calinski-Harabasz Score graph, the value that peaks in the data visually would ideally be the number of clusters, in this case, the data peaked when cluster=3 in the beginning, so it supports that the recommended number of clusters be 3. Finally with the Davies-Bouldin Index, the lowest value in the data would ideally be the number of clusters, in this case, the data was at its minimum when cluster=3 in the beginning, so it supports that the recommended number of clusters be 3.
Since the Elbow Value and the Silhouette Value recommends both either be 2 or 3 clusters and the Calinski-Harabasz Score and the Davies-Bouldin Index recommends 3 clusters, it makes sense to assume that the recommended number of clusters should be 3.

Question 6c

- Answer

Based on the values in (a), my suggested number of clusters is 3 clusters.

- Figure

```
Cluster  0
Centroid =  [1.81103977 1.98989899 2.96908939]
Unscaled Centroid =  [3635.36206897  108.89655172  188.81896552]
Size =  126.0
Within Sum of Squares =  217.32137947332734

Cluster  1
Centroid =  [3.34099117 3.51623377 4.74013158]
Unscaled Centroid =  [4874.36764706  118.97058824  201.42647059]
Size =  224.0
Within Sum of Squares =  318.2094316852603

Cluster  2
Centroid =  [5.23561894 5.7972028  6.63697706]
Unscaled Centroid =  [2785.1796875  101.0625     173.90625  ]
Size =  78.0
Within Sum of Squares =  345.9210146721626
```

- Explanation