PERFORMANCE ASSESSMENT

Task 1 | Clustering Techniques

KAILI HAMILTON

Masters of Science in Data Analytics, Western Governors University

Course: D212

Instructor: Dr. Kesselly Kamara

Program Mentor: Krissy Bryant

February, 2024

TABLE OF CONTENTS

A1

How can KMeans clustering analysis help determine the characteristics of this company's customer base relating to income and tenure so we can improve our marketing strategies?

A2

One goal of this data analysis is to use KMeans clustering to determine the characteristics of this company's customer base relating to income and tenure and to use this analysis to inform our marketing strategies.

B1

The clustering technique I chose to use to analyze this data set is KMeans, an unsupervised machine learning method. "The objective of K-means is [to] group similar data points together and discover underlying patterns. To achieve this objective, K-means looks for a fixed number (k) of clusters in a dataset. You'll define a target number k, which refers to the number of centroids you need in the dataset. A centroid is the imaginary or real location representing the center of the cluster. Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares. In other words, the K-means algorithm identifies k number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible. The 'means' in the K-means refers to averaging of the data; that is, finding the centroid." (Education Ecosystem, 2018).

B2

One important assumption with using the KMeans clustering technique is the clusters are approximately spherical and that the variance within each clusters is approximately the same. KMeans uses Euclidean distance from the centroid of the clusters and may not perform well if there are outliers or if the data set does not contain spherical clustering (Datacamp, n.d.).

B3

The packages and libraries used in this analysis are listed below, along with how they support the analysis.

Pandas	Import data into data frames and data manipulation
Numpy	Provides array objects for calculation

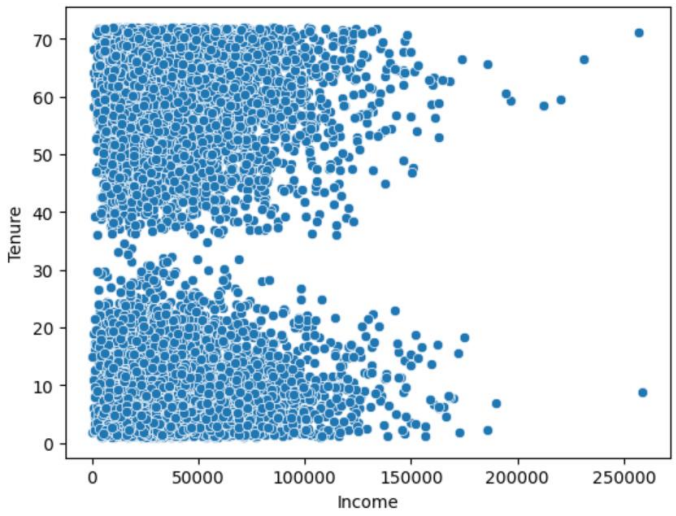
Seaborn	For visualizations
Matplotlib	For visualizations
StandardScaler from sklearn preprocessing	Standardize / Normalize values
KMeans from sklearn.cluster	Perform KMeans clustering analysis

C1

One data preprocessing goal relevant to KMeans clustering is to standardize, or Normalize, the variables used. The two variables of interest in this analysis are “Income” and “Tenure”, which have very different scales. They were Normalized to have a mean of 0 and a standard deviation of 1 so they can be compared.

C2

The initial data set contains these two continuous variables: “Income” and “Tenure”. They were chosen for KMeans cluster analysis because of the relatively spherical clusters.



Once KMeans was performed, I wanted to analyze some characteristics about the customers within each of the clusters determined by KMeans. The continuous variables included here are: “Income”, “Tenure”, “Children”, “Age”, and “MonthlyCharge”. The categorical variables included here are: “Marital”, “Gender”, “Churn”, and “Contract”.

C3

The data was prepared for analysis in the following way:

1. Initial libraries and packages were imported and the data was read in as a data frame.

```
import pandas as pd #dataframes
import numpy as np #arrays
import seaborn as sns #visualizations
```

```
from matplotlib import pyplot as plt #visualizations
```

```
df = pd.read_csv('churn_clean.csv')
df.head(3)
```

2. Exploratory data analysis
   a. Obtained some summary statistics and information on the data set.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 29 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Population            10000 non-null  int64
 1   Job                   10000 non-null  object
 2   Children              10000 non-null  int64
 3   Age                   10000 non-null  int64
 4   Income                10000 non-null  float64
 5   Marital               10000 non-null  object
 6   Gender                10000 non-null  object
 7   Churn                 10000 non-null  object
```

```
df.describe()
```

	Population	Children	Age
count	10000.000000	10000.0000	10000.000000
mean	9756.562400	2.0877	53.078400
std	14432.698671	2.1472	20.698882
min	0.000000	0.0000	18.000000
25%	738.000000	0.0000	35.000000
50%	2910.500000	1.0000	53.000000
75%	13168.000000	3.0000	71.000000
max	111850.000000	10.0000	89.000000

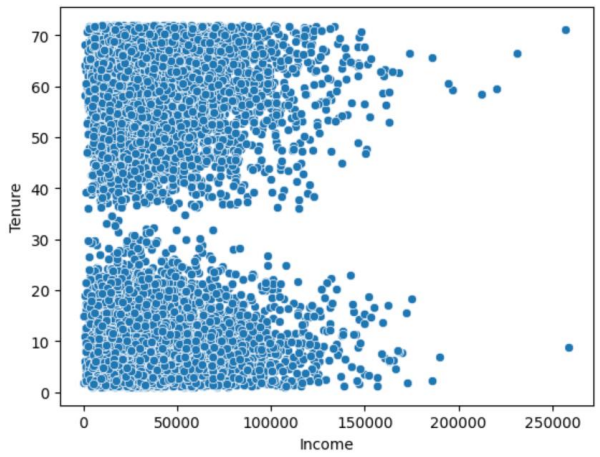
```
df.describe(include=object)
```

	Job	Marital	Gender	Churn
count	10000	10000	10000	10000
unique	639	5	3	2
top	Occupational psychologist	Divorced	Female	No
freq	30	2092	5025	7350

   b. Plotted scatter plots of continuous variables to help guide the framing of the research question expressed in part A.

```
sns.scatterplot(data=df, x="Income", y="Tenure")
```

```
<Axes: xlabel='Income', ylabel='Tenure'>
```



c. Chose variables related to the research question expressed in Part A.

```
# drop irrelavant columns

df.drop(columns=['CaseOrder',
                 'Customer_id',
                 'Interaction',
                 'UID',
                 'City',
                 'State',
                 'County',
                 'Zip',
                 'Lat',
                 'Lng',
                 'Area',
                 'TimeZone',
                 'Email',
                 'Item1',
                 'Item2',
                 'Item3',
                 'Item4',
                 'Item5',
                 'Item6',
                 'Item7',
                 'Item8'],
        inplace=True)
df.head()
```

```
df_2 = df[['Income',
     'Tenure',
     'Children',
     'Age',
     'MonthlyCharge',
     'Marital',
     'Gender',
     'Churn',
     'Contract']]

df_2.head(3)
```

3. Data preprocessing – prepared data for KMeans analysis by standardizing “Income” and “Tenure”.

```
df[['Income', 'Tenure']].head()
```

	Income	Tenure
0	28561.99	6.795513
1	21704.77	1.156681
2	9609.57	15.754144
3	18925.23	17.087227
4	40074.19	1.670972

```
# Normalize the data using StandardScaler from sklearn

from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
```

```
#scaled data frame

scaled_df_2 = scaler.fit_transform(df_2[['Income', 'Tenure']])
```

```
scaled_df_2 = pd.DataFrame(scaled_df_2, columns = ['Income', 'Tenure'])
scaled_df_2.head()
```

	Income	Tenure
count	10000.00	10000.00
mean	0.00	0.00
std	1.00	1.00

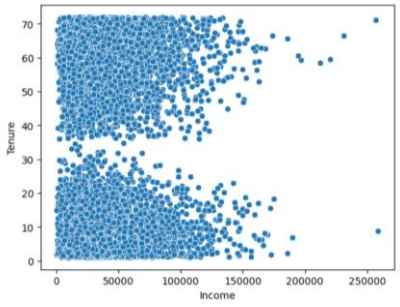
C4

A copy of the cleaned data set is provided with the submission of this Performance Assessment (PA).

D1

The optimal number of clusters, k, in the data set was determined to be $k = 4$. The method used to obtain this value is outlined as follows.

1. After the data was prepared for analysis (as outlined in part C3), I first fit the KMeans model with $k = 2$ clusters because of the two distinct groups found in the scatter plot of “income” vs “tenure”.



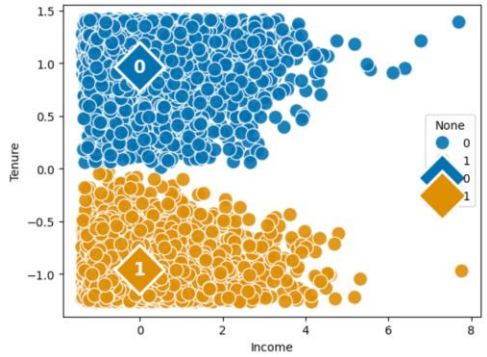

```
from sklearn.cluster import KMeans #package to perform KMeans clustering

k_model = KMeans(n_clusters = 2, n_init = 25, random_state = 496)
k_model.fit(scaled_df_2)
```

```
KMeans(n_clusters=2, n_init=25, random_state=496)
```

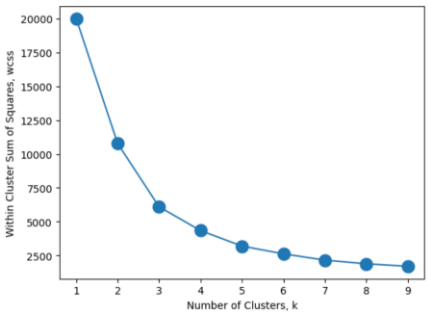
```
# evaluate the model

evaluate = pd.Series(k_model.labels_).value_counts()
evaluate
```

```
0    5001
1    4999
Name: count, dtype: int64
```

2. Next, I found the centroids of the $k = 2$ clusters and plotted the data



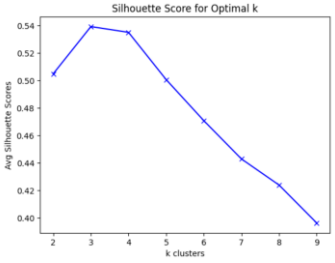
3. Now I looked for the optimal number of k values by using the elbow method with the plot of inertia values. From this plot, and using the distribution of the scatter plot of income vs tenure, we can see that $k = 4$ is an optimal number of clusters.



4. To further identify the optimal number of clusters, I calculated a silhouette score for a range of k clusters. $k = 3$ has a slightly higher silhouette score than $k = 4$, however, it's only slight and it makes more business sense to choose $k = 4$.

```
from sklearn.metrics import silhouette_score

for k in range(2, 10):
    kmeans = KMeans(n_clusters=k, n_init=25, random_state=496)
    preds = kmeans.fit_predict(scaled_df_2)
    score = silhouette_score(scaled_df_2, preds, metric = 'euclidean')
    print("For n_clusters = {}, silhouette score is {}".format(k, score))
```

```
For n_clusters = 2, silhouette score is 0.5048985830412945
For n_clusters = 3, silhouette score is 0.5393122118417948
For n_clusters = 4, silhouette score is 0.5351332735842662
For n_clusters = 5, silhouette score is 0.500685102774065
For n_clusters = 6, silhouette score is 0.47078261741147126
For n_clusters = 7, silhouette score is 0.44286344424766255
For n_clusters = 8, silhouette score is 0.42381992266922097
For n_clusters = 9, silhouette score is 0.39616976037201385
```



5. Refit the KMeans on the data using $k = 4$, find the centroids of the clusters, and plot.

```
k_model = KMeans(n_clusters = 4, n_init = 25, random_state = 496)
k_model.fit(scaled_df_2)
```

```
KMeans(n_clusters=4, n_init=25, random_state=496)
```

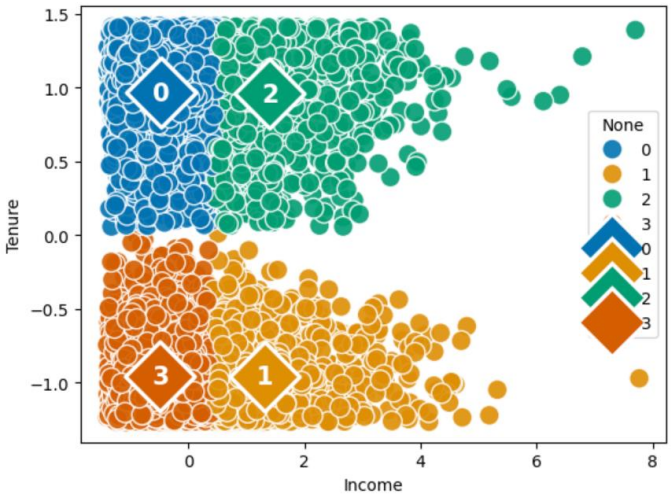
```
# evaluate the model

evaluate = pd.Series(k_model.labels_).value_counts()
evaluate
```

```
0    3759
3    3671
1    1329
2    1241
Name: count, dtype: int64
```

```
# Centroids with k=4

centeroid = pd.DataFrame(k_model.cluster_centers_, columns = ['Income', 'Tenure'])
centeroid
```

	Income	Tenure
0	-0.460490	0.962210
1	1.324107	-0.961721
2	1.406138	0.955437
3	-0.482152	-0.960103

```
plt.figure()

ax = sns.scatterplot(data = scaled_df_2,
                     x = 'Income',
                     y = 'Tenure',
                     hue = k_model.labels_,
                     palette = 'colorblind',
                     alpha = 0.9,
                     s = 150,
                     legend = True)

ax = sns.scatterplot(data = centeroid,
                     x = 'Income',
                     y = 'Tenure',
                     hue = centeroid.index,
                     palette = 'colorblind',
                     s = 900,
                     marker = 'D')

for i in range(len(centeroid)):
    plt.text(x = centeroid.Income[i],
             y = centeroid.Tenure[i],
             s = i,
             horizontalalignment = 'center',
             verticalalignment = 'center',
             size = 15,
             weight = 'bold',
             color = 'white')
```



6. Lastly, I put the cluster label back into the data frame to use for finding further insight into the characteristics of the customers within each of the four clusters.

```
df_2['Cluster'] = k_model.labels_.tolist()
df_2.head()
```

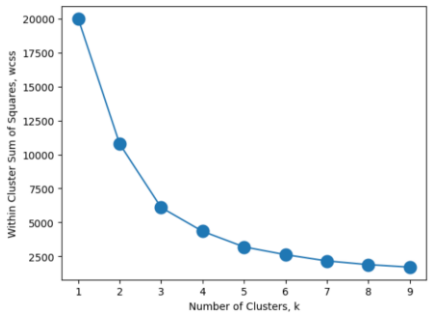
	Income	Tenure	Children	Age	MonthlyCharge	Marital	Gender	Churn	Contract	Cluster
0	28561.99	6.795513	0	68	172.455519	Widowed	Male	No	One year	3
1	21704.77	1.156681	1	27	242.632554	Married	Female	Yes	Month-to-month	3
2	9609.57	15.754144	4	50	159.947583	Widowed	Female	No	Two Year	3
3	18925.23	17.087227	1	48	119.956840	Married	Male	No	Two Year	3
4	40074.19	1.670972	0	83	149.948316	Separated	Male	Yes	Month-to-month	3

D2

A copy of the code used to perform the KMeans clustering analysis set is provided with the submission of this PA.

E1

The quality of the clusters created for $k = 4$ is a good fit. The optimal number of k clusters by using the elbow method is $k = 3$ or $k = 4$, which is better that the first chosen value of $k = 2$.



What's more, the silhouette scores indicate that $k = 2$ is not the optimal number of clusters. They indicate that $k = 3$ produces the maximum silhouette score. Even though $k = 3$ has a slightly higher silhouette score (0.539) than $k = 4$ (0.535), it's only a slight difference, and it makes more business sense to choose $k = 4$.

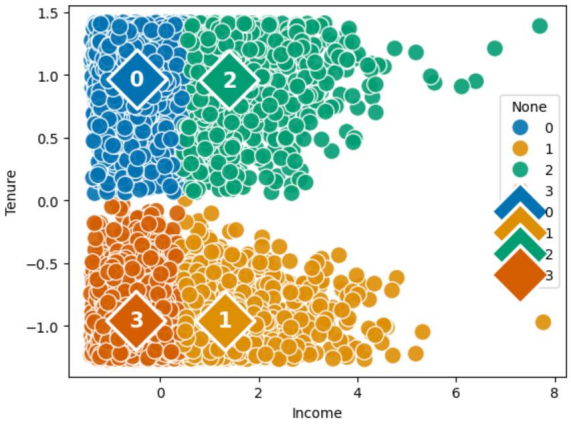
```
from sklearn.metrics import silhouette_score

for k in range(2, 10):
    kmeans = KMeans(n_clusters=k, n_init=25, random_state=496)
    preds = kmeans.fit_predict(scaled_df_2)
    score = silhouette_score(scaled_df_2, preds, metric = 'euclidean')
    print("For n_clusters = {}, silhouette score is {}".format(k, score))
```

```
For n_clusters = 2, silhouette score is 0.5048985830412945
For n_clusters = 3, silhouette score is 0.5393122118417948
For n_clusters = 4, silhouette score is 0.5351332735842662
For n_clusters = 5, silhouette score is 0.500685102774065
For n_clusters = 6, silhouette score is 0.47078261741147126
For n_clusters = 7, silhouette score is 0.44286344424766255
For n_clusters = 8, silhouette score is 0.42381992266922097
For n_clusters = 9, silhouette score is 0.39616976037201385
```

It makes business sense to have four clusters here because of distribution of the scatter plot. We can name these labels as follows: low income & low tenure, low income & high tenure, low income & high tenure, high income & high tenure.

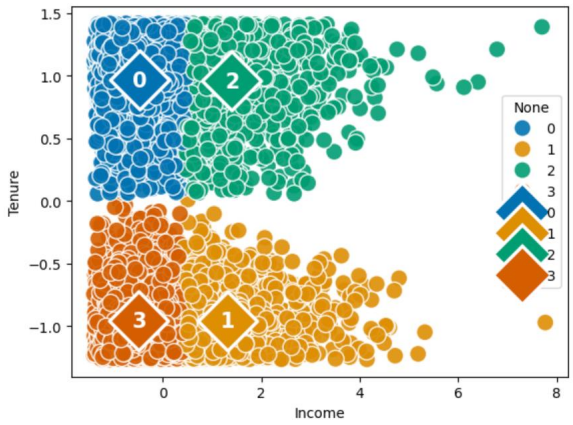
```
evaluate = pd.Series(k_model.labels_).value_counts()
evaluate
```

```
0    3759
3    3671
1    1329
2    1241
```



E2

The results and implications of this KMeans clustering are discussed here. There are four clusters, or groups relating to income and tenure, that we can segment our customer base. I will give names to the clusters:



Cluster 0 = Low Income & High Tenure	Cluster 2 = High Income & High Tenure
Cluster 3 = Low Income & Low Tenure	Cluster 1 = High Income & Low Tenure

I aggregated the data on the median and mean for the features and grouped by cluster to gain insights into each of the clusters.

Here are the median values of each feature by cluster. The median age and monthly charge is roughly the same across clusters, but Cluster 3 does have the highest monthly charge. Cluster 1 (High Income & Low Tenure) has more children on average than the other clusters.

Cluster	Income	Tenure	Children	Age	MonthlyCharge
0	25834.35	61.510350	1.0	53.0	167.456400
1	70321.71	8.000279	2.0	52.0	167.456400
2	72518.47	61.293580	1.0	55.0	164.967000
3	25560.03	7.895618	1.0	53.0	169.937833

The marital status of the customers is uniformly distributed amongst clusters

Cluster	Income	Tenure	Marital_Divorced	Marital_Married	Marital_Never Married	Marital_Separated	Marital_Widowed
0	25834.35	61.510350	0.208034	0.187018	0.196861	0.208300	0.199787
1	70321.71	8.000279	0.217457	0.193378	0.188111	0.198646	0.202408
2	72518.47	61.293580	0.199839	0.209508	0.195810	0.189363	0.205479
3	25560.03	7.895618	0.210569	0.188232	0.196949	0.199401	0.204849

Cluster 3 is the only cluster where there are more males than females, but there isn't a cluster that has a significant amount more of one gender over the other.

Cluster	Income	Tenure	Gender_Female	Gender_Male	Gender_Nonbinary
0	25834.35	61.510350	0.505720	0.475126	0.019154
1	70321.71	8.000279	0.528969	0.445448	0.025583
2	72518.47	61.293580	0.522965	0.452055	0.024980
3	25560.03	7.895618	0.482702	0.491692	0.025606

Churn rates are roughly the same for each of the low tenure clusters and for the high tenure clusters

Cluster	Income	Tenure	Churn_No	Churn_Yes
0	25834.35	61.510350	0.944400	0.055600
1	70321.71	8.000279	0.515425	0.484575
2	72518.47	61.293580	0.940371	0.059629
3	25560.03	7.895618	0.530646	0.469354

Contract types are also uniformly distributed across clusters.

Cluster	Income	Tenure	Contract_Month-to-month	Contract_One year	Contract_Two Year
0	25834.35	61.510350	0.540303	0.206970	0.252727
1	70321.71	8.000279	0.558315	0.212942	0.228743
2	72518.47	61.293580	0.536664	0.211926	0.251410
3	25560.03	7.895618	0.549442	0.211931	0.238627

The implications are that of these features included, there isn't an outstanding characteristic of each group. Overall the customers are fairly similar within each cluster. There is a clear separation of the two low tenure clusters and the two high tenure clusters at around 30 months.

E3

One limitation of my data analysis is that KMeans clustering might not lend itself well in using "Income" and "Tenure". There are outliers in the data set (KMeans is sensitive to outliers) and the shape of the clusters isn't very spherical, so assignment of datapoints to clusters could be off because of the calculation of the centroids.

E4

A course of action to aid in the goal described in Part A is to use insight into our customers' characteristics to guide our marketing strategies. Our analysis shows that, out the features chosen for analysis, there isn't a standout characteristic difference amongst the four clusters. Customers are more similar than they are different within income and tenure groups. Marketing and product efforts should be spent on providing quality products and services that are accessible to everyone.

That being said, there is a big difference between the median tenure of a customer in the two high tenure clusters (61 months) and that of the two low tenure clusters (8 months). The customer retention and sales department should focus efforts on obtaining longer contract lengths to keep the customers with our company longer.

F

A Panopto video recording of the functionality of the code used in this PA is included with the submission.

G

Kamara, Kesselly. "D212 Recommended Study Plan". D212 Western Governors University, n.d., srm.file.force.com/servlet/fileField?id=0BE3x000000c9YS.

Wilson, Benjamin. "Unsupervised Learning in Python". n.d., app.datacamp.com/learn/courses/unsupervised-learning-in-python

Stuart, Lisa. "Clustering analysis: choosing the opitmal number of clusters". n.d., campus.datacamp.com/courses/practicing-machine-learning-interview-questions-in-python/unsupervised-learning-467e974f-beb6-47c3-bfbe-a71d5a36b323?ex=13

H

Education Ecosystem. "Understanding K-Means Clustering in Machine Learning." Towards Data Science, Medium, September 12, 2018, https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1.

I

Professional communication is demonstrated throughout the content and presentation of this PA.