COMP450 LAB WORK WEEK-10

Aga Saltikalp
041901048

TASK-10A:

```
[ ] import numpy as np
    import pandas as pd
    import statsmodels.api as sm
    import matplotlib.pyplot as plt
    import seaborn as sb
    from sklearn.cluster import KMeans
    from sklearn.preprocessing import StandardScaler
    %matplotlib inline
```

```
[ ] df = pd.read_csv('/content/ifood_df.csv')
```

```
[ ] df.head(50)
```

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFish
0	58138.0	0	0	58	635	88	546	
1	46344.0	1	1	38	11	1	6	
2	71613.0	0	0	26	426	49	127	
3	26646.0	1	0	26	11	4	20	
4	58293.0	1	0	94	173	43	118	
5	62513.0	0	1	16	520	42	98	
6	55635.0	0	1	34	235	65	164	
7	33454.0	1	0	32	76	10	56	
8	30351.0	1	0	19	14	0	24	

```
[ ] # standardizing features
    scaler = StandardScaler()
    scaled_features = scaler.fit_transform(df[['Income',  'Kidhome', 'Teenhome', 'Recency'
```

```
kmeans = KMeans(n_clusters=7, random_state=42)
kmeans.fit(scaled_features)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning:
  warnings.warn(
```

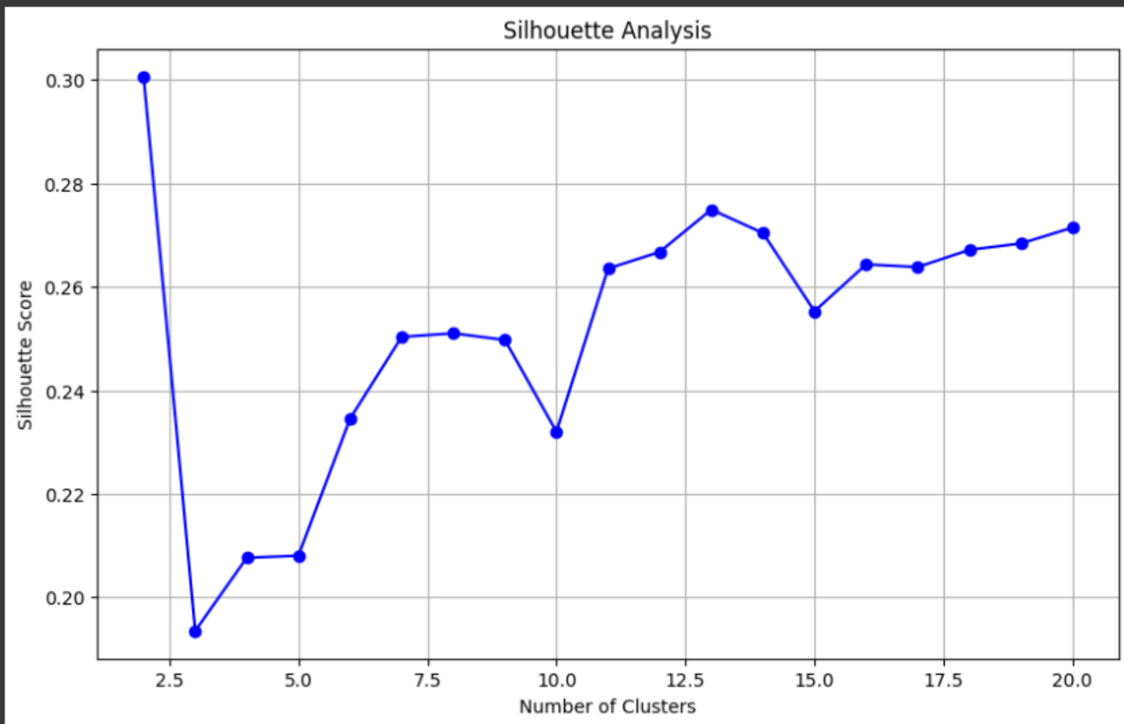
KMeans
KMeans(n_clusters=7, random_state=42)

```
[ ] from sklearn.metrics import silhouette_score

    n_clusters = list(range(2, 21))
```

```
[ ] silhouette_scores = []

    for n in n_clusters:
        kmeans = KMeans(n_clusters=n, random_state=42)
        cluster_labels = kmeans.fit_predict(scaled_features)
        silhouette_avg = silhouette_score(scaled_features, cluster_labels)
        silhouette_scores.append(silhouette_avg)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning:
```

```
[ ] plt.figure(figsize=(10, 6))
    plt.plot(n_clusters, silhouette_scores, marker='o', linestyle='-', color='b')
    plt.title('Silhouette Analysis')
    plt.xlabel('Number of Clusters')
    plt.ylabel('Silhouette Score')
    plt.grid(True)
    plt.show()
```



TASK-10B:

```
[1] import numpy as np
    import pandas as pd
    import statsmodels.api as sm
    import matplotlib.pyplot as plt
    import seaborn as sb
    from sklearn.cluster import KMeans
    from sklearn.preprocessing import StandardScaler
    %matplotlib inline
```

```
df = pd.read_csv('/content/ifood_df.csv')
```

```
[ ] df.head()
```

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishPro
0	58138.0	0	0	58	635	88	546	
1	46344.0	1	1	38	11	1	6	
2	71613.0	0	0	26	426	49	127	
3	26646.0	1	0	26	11	4	20	
4	58293.0	1	0	94	173	43	118	

5 rows × 39 columns

```python
def remove_outliers(df, column_names):
    for column in column_names:
        Q1 = df[column].quantile(0.25)
        Q3 = df[column].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR
        df = df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]
    return df
```

[7]
```python
columns = ['Income', 'Kidhome', 'Teenhome', 'Recency', 'MntFruits', 'MntMeatProducts',
           'NumDealsPurchases', 'AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3',
           'AcceptedCmp4', 'AcceptedCmp5']
```

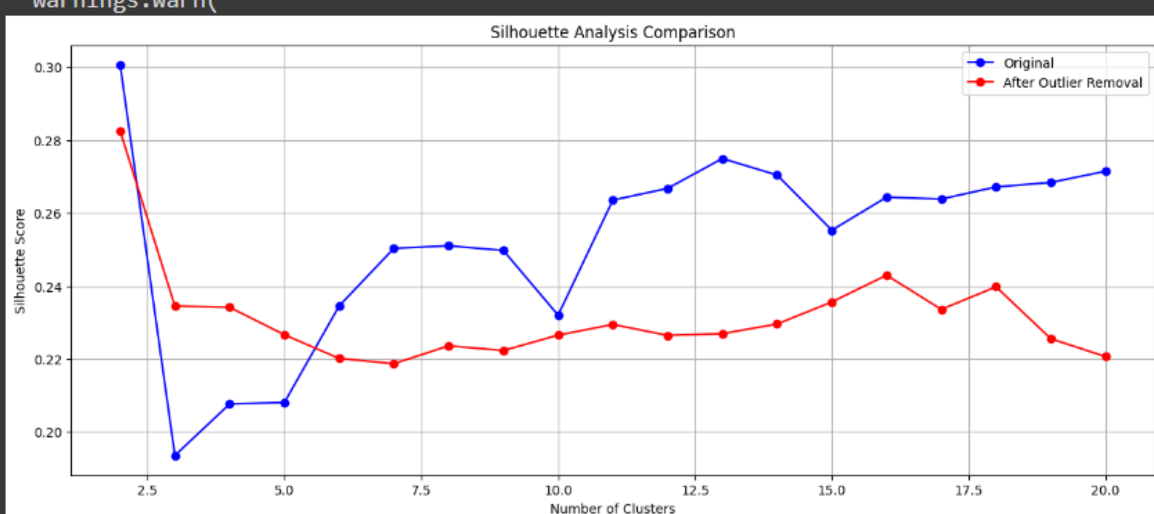
[8]
```python
df_no_outliers = remove_outliers(df, columns)
```

```python
# Scaling original features
scaler = StandardScaler()
scaled_features = scaler.fit_transform(df[columns])
```

[21]
```python
# Standardizing features after removing outliers
scaled_features_no_outliers = scaler.fit_transform(df_no_outliers[columns])
```

```python
# n_clusters for the original dataset
n_clusters_original = range(2, 21)
silhouette_scores_original = []

# silhouette scores for the original dataset
for n in n_clusters_original:
    kmeans_original = KMeans(n_clusters=n, random_state=42)
    cluster_labels_original = kmeans_original.fit_predict(scaled_features)
    silhouette_avg_original = silhouette_score(scaled_features, cluster_labels_origin
    silhouette_scores_original.append(silhouette_avg_original)

# Plotting the silhouette scores for comparison
plt.figure(figsize=(15, 6))

plt.plot(n_clusters_original, silhouette_scores_original, marker='o', linestyle='-', c

plt.plot(n_clusters_range, silhouette_scores_no_outliers, marker='o', linestyle='-', c

plt.title('Silhouette Analysis Comparison')
plt.xlabel('Number of Clusters')
plt.ylabel('Silhouette Score')
plt.grid(True)
plt.legend()
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning:
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning:
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning:
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning:
  warnings.warn(
```



Differences:

Outlier elimination and feature scaling are expected to significantly enhance the quality of KMeans clustering in your dataset. By removing extreme values, outlier elimination will help create more homogeneous clusters, reducing the skewness caused by these anomalies. Feature scaling ensures all variables contribute equally, regardless of their original scale, leading to a more balanced clustering process. This will not only stabilize cluster assignments by reducing the influence of outliers, but it may also result in different optimal cluster numbers, as indicated by changes in silhouette scores. Additionally, these preprocessing steps generally lead to cleaner, more distinct clusters, facilitating easier interpretation and clearer insights into customer segments.

TASK-10C:

[1]
```
import numpy as np
import pandas as pd
import statsmodels.api as sm
import matplotlib.pyplot as plt
import seaborn as sb
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
%matplotlib inline
```

[3]
```
df = pd.read_csv('/content/ifood_df.csv')
```

[ ]
```
df.head(50)
```

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFish
0	58138.0	0	0	58	635	88	546	
1	46344.0	1	1	38	11	1	6	
2	71613.0	0	0	26	426	49	127	
3	26646.0	1	0	26	11	4	20	
4	58293.0	1	0	94	173	43	118	
5	62513.0	0	1	16	520	42	98	
6	55635.0	0	1	34	235	65	164	
7	33454.0	1	0	32	76	10	56	
8	30351.0	1	0	19	14	0	24	

1s completed at 9:40 PM

```
[4] def remove_outliers(df, column_names):
        for column in column_names:
            Q1 = df[column].quantile(0.25)
            Q3 = df[column].quantile(0.75)
            IQR = Q3 - Q1
            lower_bound = Q1 - 1.5 * IQR
            upper_bound = Q3 + 1.5 * IQR
            df = df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]
        return df
```

```
[5] columns = ['Income', 'Kidhome', 'Teenhome', 'Recency', 'MntFruits', 'MntMeatProducts',
               'NumDealsPurchases', 'AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3',
               'AcceptedCmp4', 'AcceptedCmp5']
```

```
[6]
    df_no_outliers = remove_outliers(df, columns)
```

```
[7] # Standardizing features
    scaler = StandardScaler()
    scaled_features_no_outliers = scaler.fit_transform(df_no_outliers[columns])
```

```
[8] from sklearn.mixture import GaussianMixture
    # Rerun KMeans clustering and silhouette analysis
    n_components_range = range(2, 21)
    aic_scores = []
    bic_scores = []

    for n in n_components_range:
```

1s completed at 9:40 PM

```
plt.figure(figsize=(15, 6))

plt.plot(n_components_range, aic_scores, marker='o', linestyle='-', color='b', label='
plt.plot(n_components_range, bic_scores, marker='o', linestyle='-', color='r', label='

plt.title('GMM AIC and BIC Scores')
plt.xlabel('Number of Components (Clusters)')
plt.ylabel('Scores')
plt.grid(True)
plt.legend()
plt.show()
```