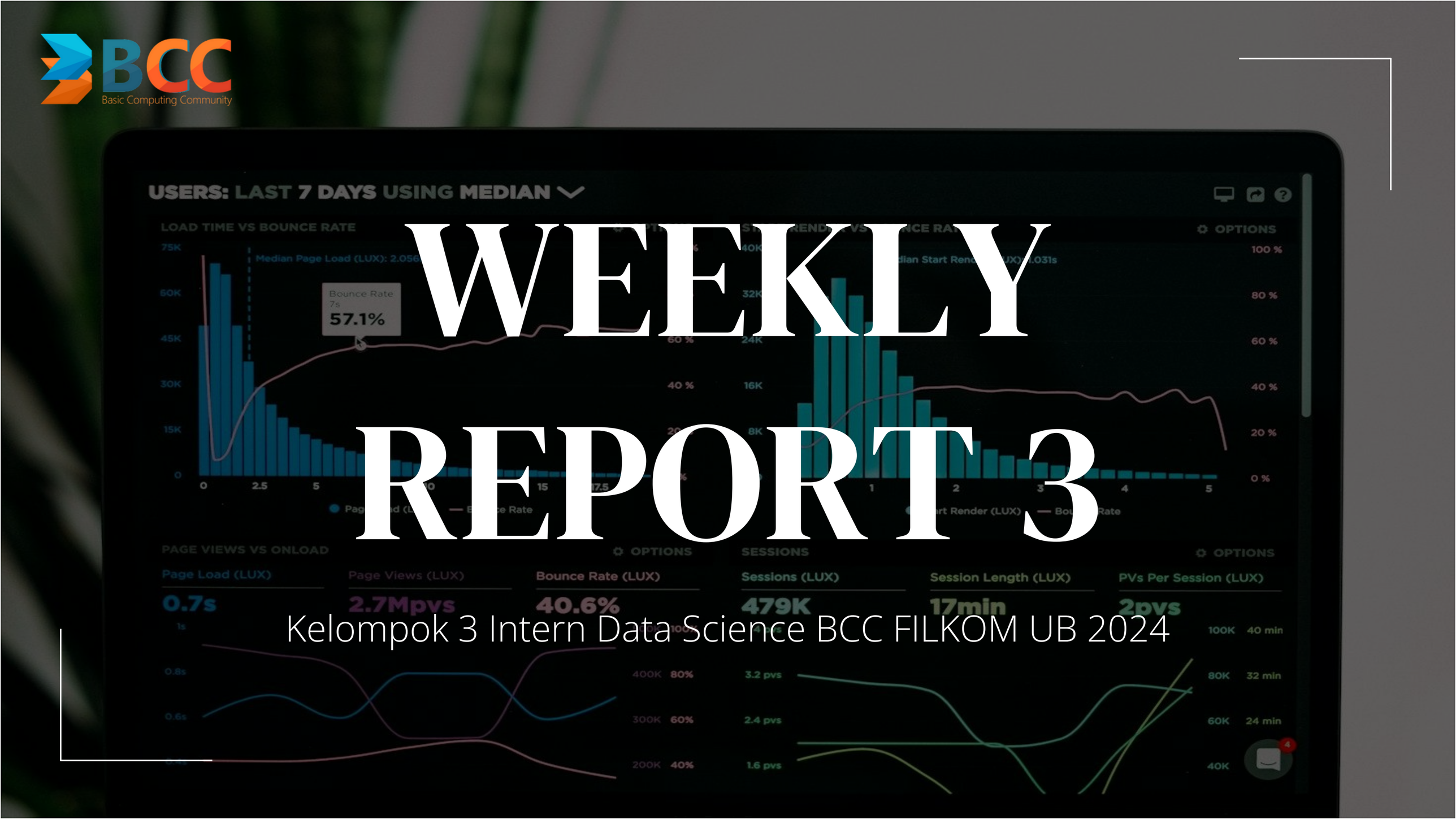


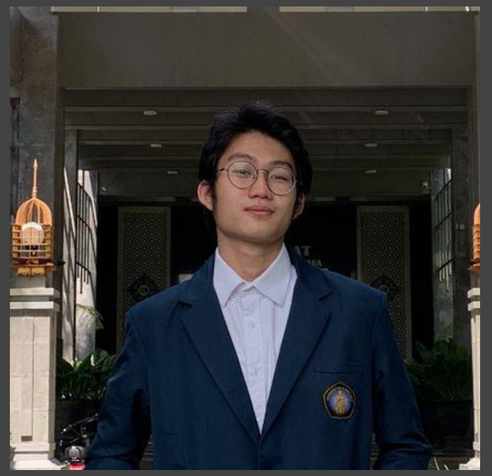
WEEKLY REPORT 3

Kelompok 3 Intern Data Science BCC FILKOM UB 2024

OUR TEAM

Johanes Paulus Bernard Purek

Sistem Informasi ‘22



Pieter Christy Yan Yudhistira

Teknik Informatika ‘23

Contents

Data Preprocessing

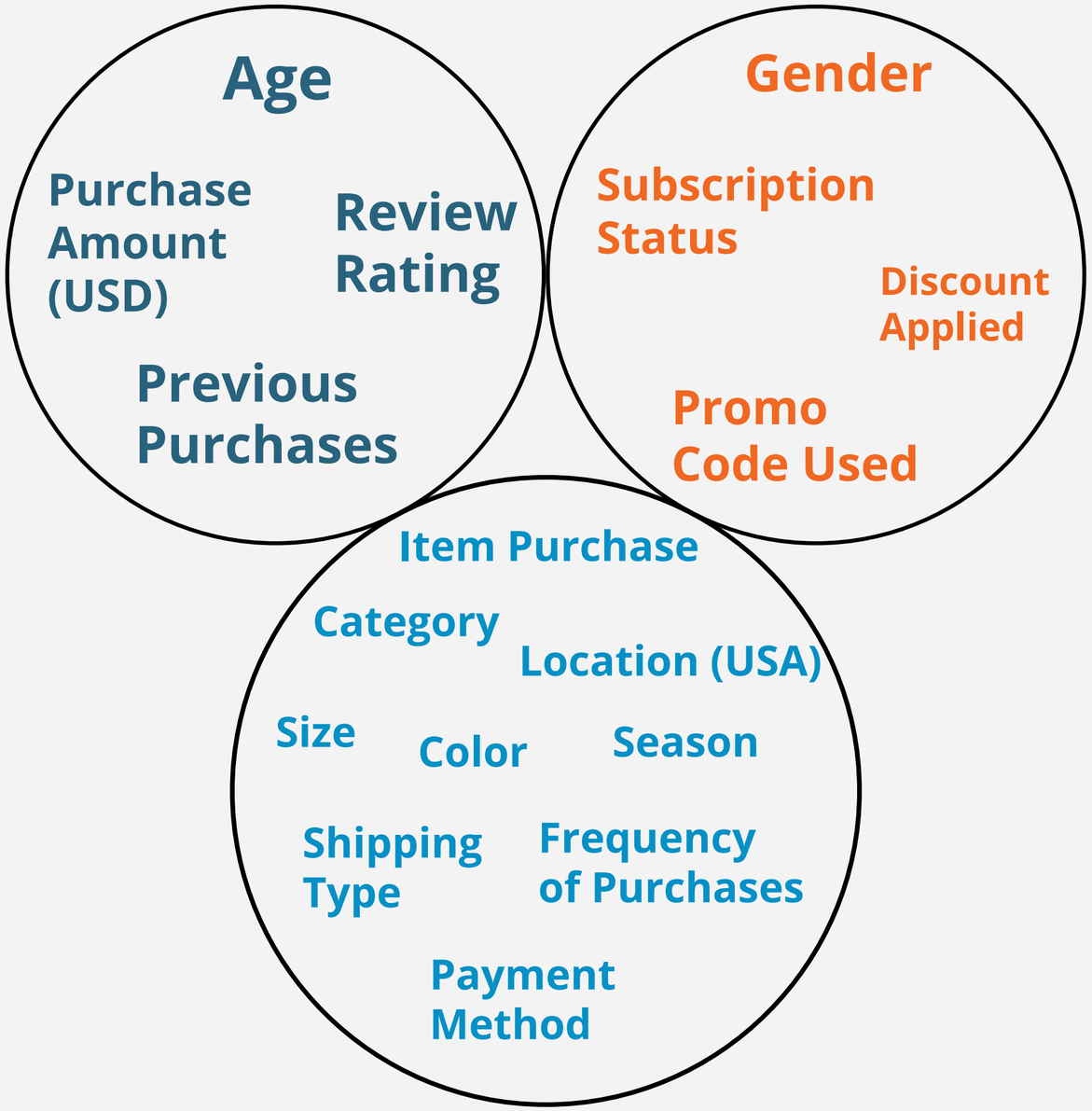
Machine Learning Implementation

Evaluation

BCC
Basic Computing Community

DATA PREPROCESSING

DATASET COLUMNS




```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3900 entries, 0 to 3899
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Customer ID             3900 non-null   int64
 1   Age                     3900 non-null   int64
 2   Gender                  3900 non-null   object
 3   Item Purchased          3900 non-null   object
 4   Category                3900 non-null   object
 5   Purchase Amount (USD)   3900 non-null   int64
 6   Location                3900 non-null   object
 7   Size                    3900 non-null   object
 8   Color                   3900 non-null   object
 9   Season                  3900 non-null   object
 10  Review Rating           3900 non-null   float64
 11  Subscription Status     3900 non-null   object
 12  Shipping Type           3900 non-null   object
 13  Discount Applied        3900 non-null   object
 14  Promo Code Used         3900 non-null   object
 15  Previous Purchases      3900 non-null   int64
 16  Payment Method          3900 non-null   object
 17  Frequency of Purchases  3900 non-null   object
dtypes: float64(1), int64(4), object(13)
memory usage: 548.6+ KB
```

MISSING VALUE OR DUPLICATE?

```
Customer ID                0
Age                        0
Gender                     0
Item Purchased             0
Category                   0
Purchase Amount (USD)      0
Location                   0
Size                       0
Color                      0
Season                     0
Review Rating              0
Subscription Status        0
Shipping Type              0
Discount Applied           0
Promo Code Used            0
Previous Purchases         0
Payment Method             0
Frequency of Purchases     0
dtype: int64
```

Any Missing Value?

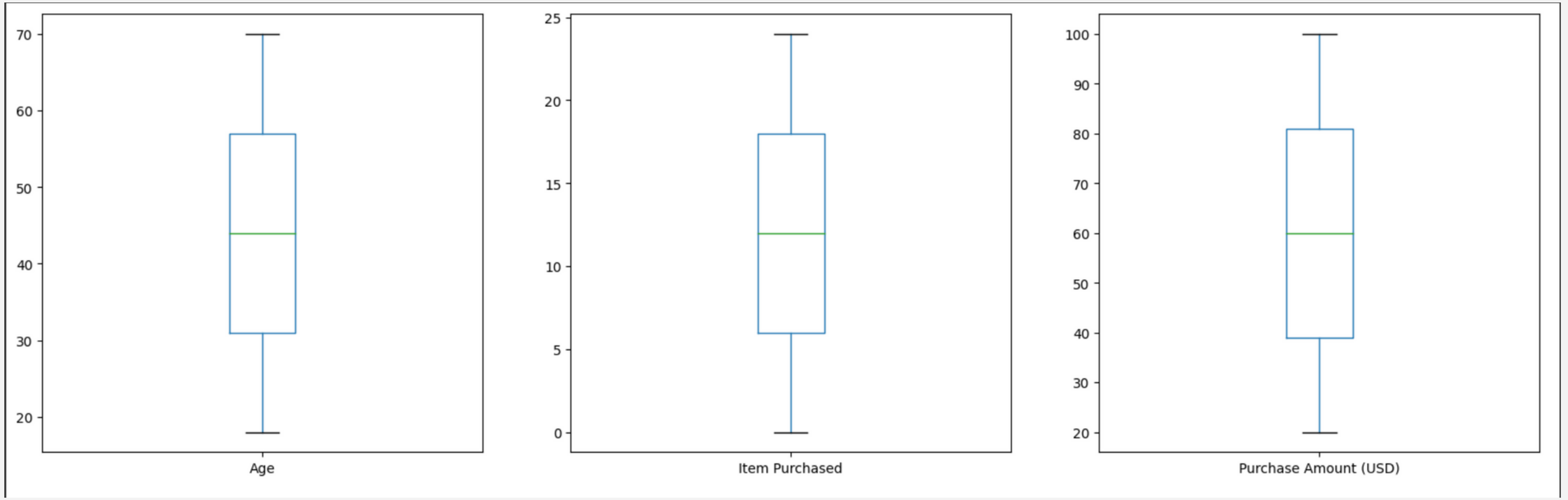
Any Duplicated Value?

```
[19] data.duplicated().sum()

     0
```

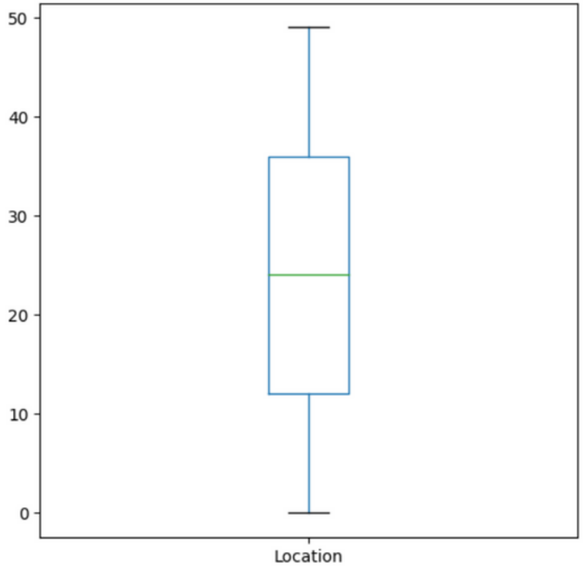
Even When Removed the Customer ID?

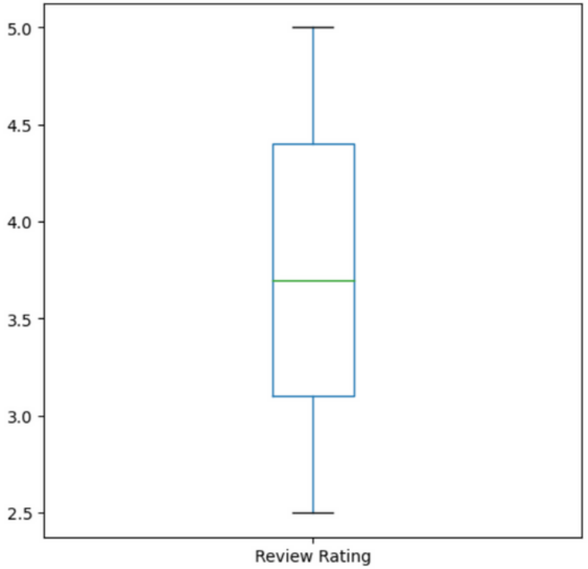
```
[20] duplicate_without_id = (data.drop(columns=["Customer ID"]))
     duplicate_without_id.duplicated().sum()

     0
```

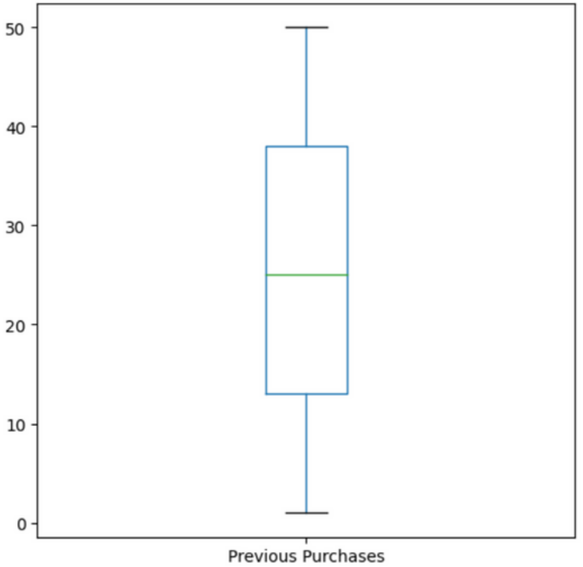
OUTLIER

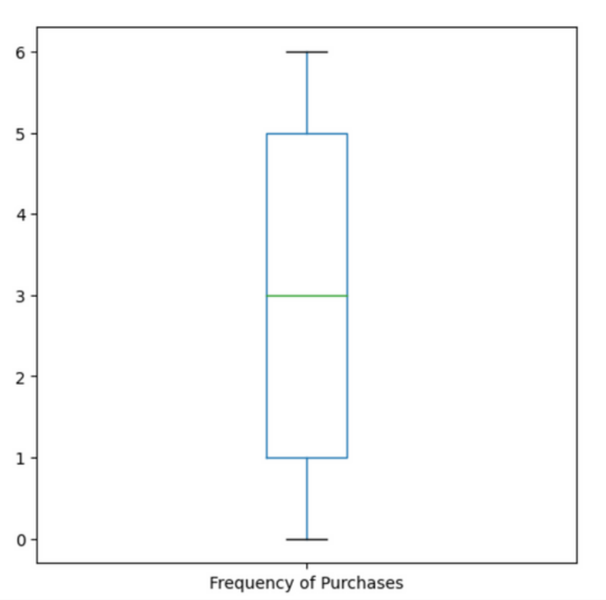

70
60
50
40
30
20
Age
25
20
15
10
5
0
Item Purchased
100
90
80
70
60
50
40
30
20
Purchase Amount (USD)

OUTLIER


50
40
30
20
10
0
Location


5.0
4.5
4.0
3.5
3.0
2.5
Review Rating


50
40
30
20
10
0
Previous Purchases


6
5
4
3
2
1
0
Frequency of Purchases

Age Column Grouping

Sumber : https://libguides.usc.edu/busdem/age

- The Greatest Generation – born 1901-1924.
- The Silent Generation – born 1925-1945.
- The Baby Boomer Generation – born 1946-1964.
- Generation X – born 1965-1979.
- Millennials – born 1980-1994.
- Generation Z – born 1995-2012.
- Gen Alpha – born 2013 – 2025.

Our Age Grouping (5 Unique Value)

```
[ ]  # Define a function to classify age groups
     def classify_age_group(age):
         if age <= 19:
             return 'Teenager'
         elif 20 <= age <= 30:
             return 'Young_Adult'
         elif 31 <= age <= 44:
             return 'Adult'
         elif 45 <= age <= 59:
             return 'Middle_Age'
         else:
             return 'Elder'

     # Apply the function to create a new column 'Age_group'
     data['Age Group'] = data['Age'].apply(classify_age_group)
```

“Usia pertengahan (middle age) atau disebut juga sebagai usia paruh baya yaitu kelompok usia dari 45-59 tahun.”
Sumber : http://repo.unand.ac.id/397/3/bab%25201.pdf

“A teenager, or teen, is someone who is 13 to 19 years old.”
Sumber : https://simple.wikipedia.org/wiki/Teenager

“In addition, while the older shoppers are more influenced by social support to adopt healthy shopping habits, the younger shoppers are more influenced by the relative price of products.”
Sumber : https://dl.acm.org/doi/abs/10.1145/3209219.3209253

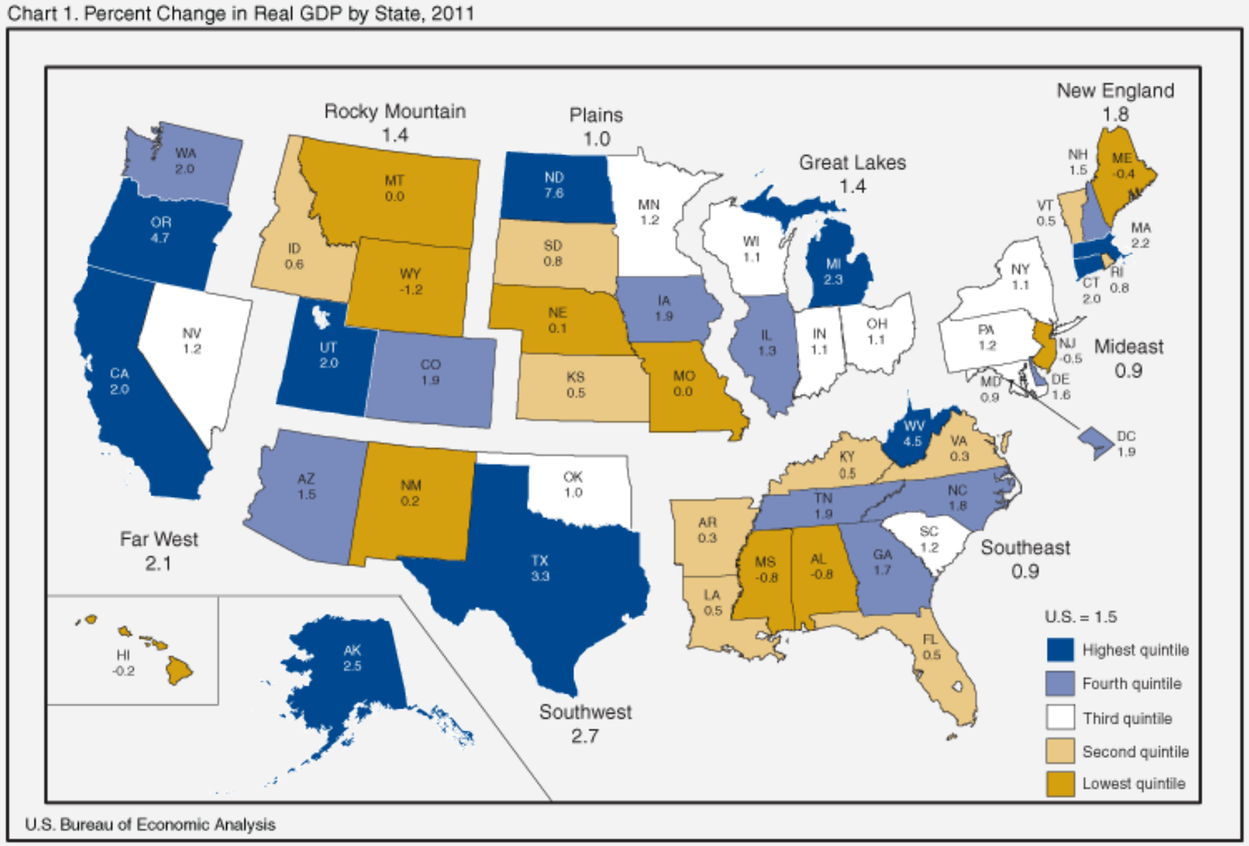
Sumber : Classification of Age Groups Based on Facial Features

Age Group	Age Intervals
Baby	0~2
Young Adults	3~12
	13~19
	20~29
	30~39
Middle-aged Adults	40~49
	50~59
Old Adults	60~69
	70~79
	80~89
	90~99

Value Counts

```
Middle_Age     1142
Adult          1021
Young_Adult     799
Elder           788
Teenager        150
Name: Age Group, dtype: int64
```

Sumber : http://www.bea.gov/newsreleases/regional/gdp_state/2012/_images/gsp_0612.png



1. **New England:** Connecticut, Maine, Massachusetts, New Hampshire, Rhode Island, and Vermont
2. **Mideast:** Delaware, District of Columbia, Maryland, New Jersey, New York, and Pennsylvania
3. **Great Lakes:** Illinois, Indiana, Michigan, Ohio, and Wisconsin
4. **Plains:** Iowa, Kansas, Minnesota, Missouri, Nebraska, North Dakota, and South Dakota
5. **Southeast:** Alabama, Arkansas, Florida, Georgia, Kentucky, Louisiana, Mississippi, North Carolina, South Carolina, Tennessee, Virginia, and West Virginia
6. **Southwest:** Arizona, New Mexico, Oklahoma, and Texas
7. **Rocky Mountain:** Colorado, Idaho, Montana, Utah, and Wyoming
8. **Far West:** Alaska, California, Hawaii, Nevada, Oregon, and Washington

LOCATION COLUMN GROUPING

Based on Bureau of Economic Analysis (BEA) regions

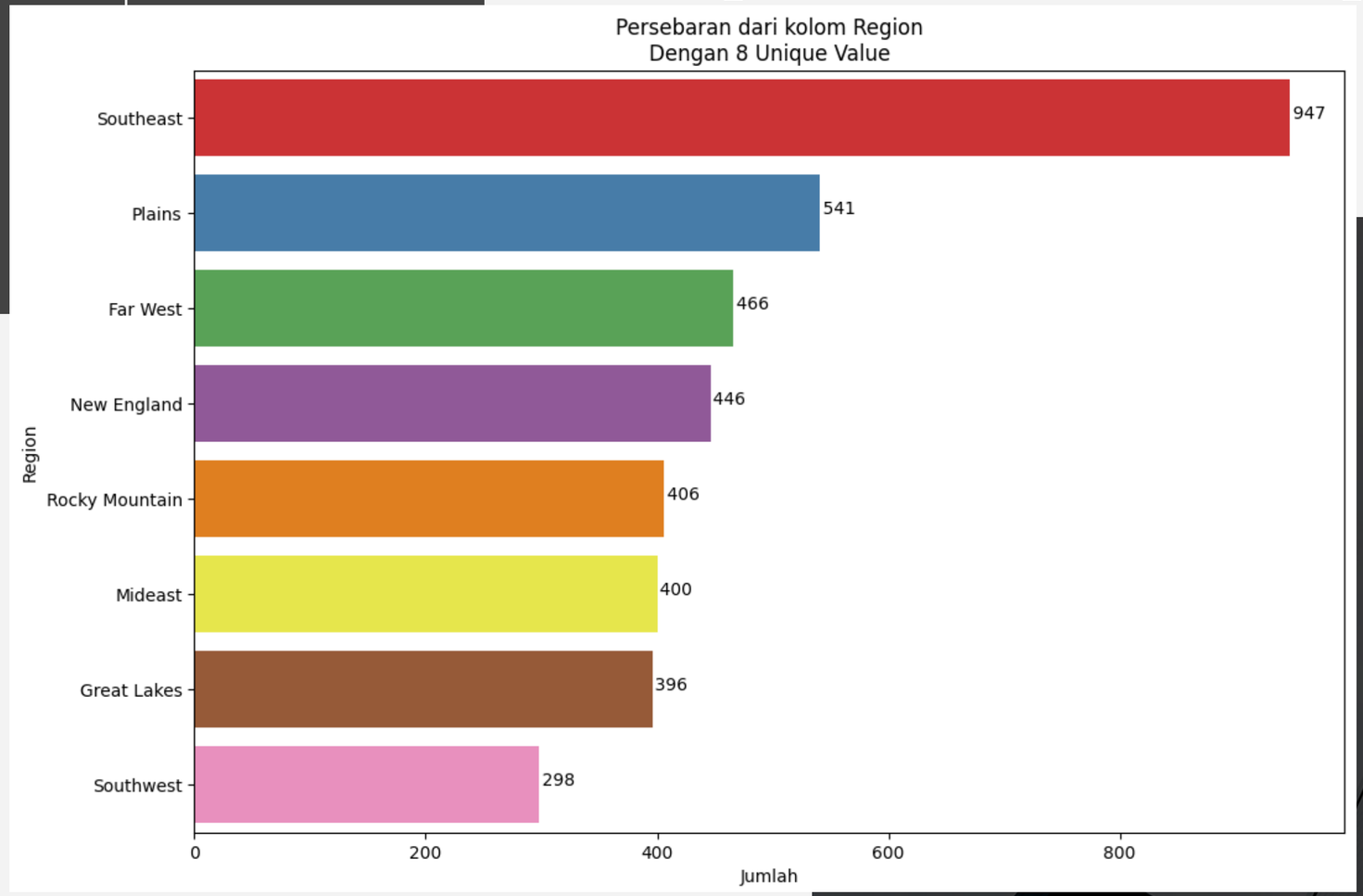
https://apps.bea.gov/regional/docs/msalist.cfm?mlist=2

Rank	Mexican Produce type	$/Week	Puerto Rican Produce type	$/Week
1	Chili Jalapeno (Chili pepper) *Capsicum species*	2.76	Batata (Sweet potato) *Ipomoea batatas*	1.74
2	Tomatillo (Husk tomato) *Physalis philadelphica Lam*	1.73	Cilantro (Coriander) *Coriandrum sativum*	1.68
3	Calabaza (Pumpkin) *Cucurbita moschata*	1.49	Aji Dulce (Capsicum) *Capsicum species*	1.58
4	Chili Poblano (Poblano pepper) *Capsicum species*	1.28	Calabaza (Pumpkin) *Cucurbita moschata*	0.96
5	Calabacita (Little squash) *Cucurbita pepo*	1.28	Pepinillo (Bitter gourd) *Momordica charantia*	0.70
6	Cilantro (Coriander) *Coriandrum sativum*	1.24	Fava Beans (Broad Beans) *Vicia faba*	0.63
7	Chili Serrano (Serrano pepper) *Capsicum species*	0.92	Chili Caribe (Chili pepper) *Capsicum species*	0.56
8	Anaheim Pepper (California chili) *Capsicum species*	0.83	Berenjena (Eggplant) *Solanum melongena*	0.51
9	Chili Habanaro (habanero chili) *Capsicum Chinese*	0.24	Calabacita (Little squash) *Cucurbita pepo*	0.43
10	Tutuma (Mexican squash) *Cucurbita species*	0.10	Verdolaga (Pigweed) *Portulaca oleracea*	0.10

Table V. Average weekly expenditure of the selected ethnic specialty produce types

Sumber : "Preference for ethnic specialty produce by the Hispanics in the east coast of the USA"

50 Unique Value => 8 Unique Value



LOCATION COLUMN GROUPING

Table 2
Descriptive statistics by commodity.

Commodities	Value per metric ton (Valueton) ($/metric ton)				Freight Charges (FRT) ($/shipment)				Transit Time (Transit) (Days)				East Coast (%)
	Min	Max	Mean	STD	Min	Max	Mean	STD	Min	Max	Mean	STD	
Animal & Animal Products	410	12,739	2,931	1,986	11	55,868	6,854	7,660	18.70	39.43	28.76	4.50	0.63
Chemicals & Allied Industries	218	46,819	6,484	7,203	17	182,230	7,426	14,681	19.18	35.92	27.80	5.31	0.47
Foodstuffs	202	7,830	2,825	960	4	92,519	20,623	22,816	18.70	35.63	20.95	4.76	0.08
Footwear/Headgear	1,909	69,917	11,734	8,194	6	76,162	6,518	11,337	21.55	35.75	26.92	4.35	0.25
Machinery/Electrical	211	86,205	18,702	16,963	4	153,162	5,303	10,204	18.70	35.92	26.92	5.70	0.38
Metals	150	76,786	5,283	5,921	7	215,453	11,911	24,444	18.70	42.80	28.26	5.25	0.44
Miscellaneous	1,040	89,116	8,968	10,176	2	218,630	5,565	11,189	19.98	35.92	29.09	5.19	0.48
Other	1	34,028	4,413	3,449	13	76,191	3,490	6,653	19.52	41.08	29.08	5.28	0.55
Plastics/Rubbers	330	34,948	4,897	3,865	5	44,948	3,932	4,460	19.18	36.57	27.75	5.95	0.44
Raw Hides, Skins, Leather, & Furs	1,484	91,372	10,558	12,010	2	127,356	2,684	7,093	18.70	35.75	29.71	4.67	0.55
Stone/Glass	284	72,752	5,582	11,565	12	138,856	8,572	11,570	19.16	35.92	26.24	5.21	0.31
Textiles	1,111	100,615	19,038	10,890	3	177,450	7,306	13,055	19.18	36.23	26.85	3.17	0.11
Transportation	713	82,158	8,394	8,920	3	52,641	3,465	4,682	19.52	35.72	29.82	5.53	0.60
Wood & Wood Products	402	7,545	2,332	1,298	6	56,320	7,574	9,100	21.57	35.75	29.35	4.75	0.52

Sumber : "East Coast vs. West Coast: The impact of the Panama Canal's expansion on the routing of Asian imports into the United States"

BCC
Basic Computing Community

LOCATION COLUMN GROUPING

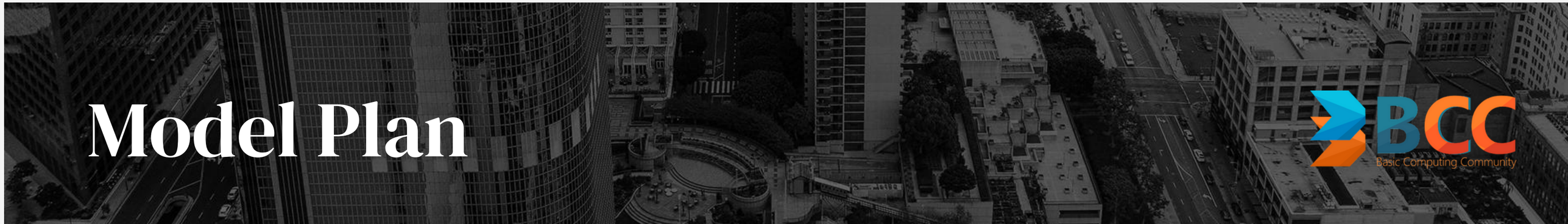
```
df['Region'] = df['Location'].apply(get_region)

df[['Age', 'Region']].head(10)
```

	Age	Region
0	55	Southeast
1	19	New England
2	50	New England
3	21	New England
4	45	Far West
5	46	Rocky Mountain
6	63	Rocky Mountain
7	27	Southeast
8	26	Southeast
9	57	Plains

Model 1
Data with Minimal Preprocessing

Model 2
Data Without One Hot Encoding Column

Model 3
Full Preprocess Data



Model Plan

BCC
Basic Computing Community

Categorical Column Handling

REMOVE USELESS FEATURE

```
nan_process_data = drop_column(data.copy(), ["Customer ID"])
nan_process_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3900 entries, 0 to 3899
Data columns (total 19 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Age                     3900 non-null   int64
 1   Gender                  3900 non-null   object
 2   Item Purchased          3900 non-null   object
 3   Category                3900 non-null   object
 4   Purchase Amount (USD)   3900 non-null   int64
 5   Location                3900 non-null   object
 6   Size                    3900 non-null   object
 7   Color                   3900 non-null   object
 8   Season                  3900 non-null   object
 9   Review Rating           3900 non-null   float64
 10  Subscription Status     3900 non-null   object
 11  Shipping Type           3900 non-null   object
 12  Discount Applied        3900 non-null   object
 13  Promo Code Used         3900 non-null   object
 14  Previous Purchases      3900 non-null   int64
 15  Payment Method          3900 non-null   object
 16  Frequency of Purchases  3900 non-null   object
 17  Region                  3900 non-null   object
 18  Age Group               3900 non-null   object
dtypes: float64(1), int64(3), object(15)
memory usage: 579.0+ KB
```

Model 1

just removed the Customer ID

```
nan_encoding_data = processed_data = drop_column(data.copy(), ["Customer ID", "Item Purchased", "Color", "Location"])
processed_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3900 entries, 0 to 3899
Data columns (total 16 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Age                     3900 non-null   int64
 1   Gender                  3900 non-null   object
 2   Category                3900 non-null   object
 3   Purchase Amount (USD)   3900 non-null   int64
 4   Size                    3900 non-null   object
 5   Season                  3900 non-null   object
 6   Review Rating           3900 non-null   float64
 7   Subscription Status     3900 non-null   object
 8   Shipping Type           3900 non-null   object
 9   Discount Applied        3900 non-null   object
 10  Promo Code Used         3900 non-null   object
 11  Previous Purchases      3900 non-null   int64
 12  Payment Method          3900 non-null   object
 13  Frequency of Purchases  3900 non-null   object
 14  Region                  3900 non-null   object
 15  Age Group               3900 non-null   object
dtypes: float64(1), int64(3), object(12)
memory usage: 487.6+ KB
```

Model 2 and Model 3

Removed Unique Value Heavy Column

BCC Basic Computing Community

chase mount (USD)	Size	Season	Review Rating	Subscription Status	Shipping Type	Discount Applied	...	New England	Plains	Rocky Mountain	Southeast	Southwest	Adult	Elder	Middle_Age	Teenager	Young_Adult
53	0	3	3.1	1	1	1	...	0	0	0	1	0	0	0	1	0	0
64	0	3	3.1	1	1	1	...	1	0	0	0	0	0	0	0	1	0
73	2	1	3.1	1	2	1	...	1	0	0	0	0	0	0	1	0	0
90	1	1	3.5	1	3	1	...	1	0	0	0	0	0	0	0	0	1
49	1	1	2.7	1	2	1	...	0	0	0	0	0	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
28	0	2	4.2	0	0	0	...	0	0	0	1	0	1	0	0	0	0
49	0	1	4.5	0	5	0	...	0	1	0	0	0	0	0	1	0	0
33	0	1	2.9	0	4	0	...	0	0	0	0	0	0	0	1	0	0
77	2	2	3.8	0	1	0	...	0	1	0	0	0	1	0	0	0	0
81	1	1	3.1	0	5	0	...	0	0	0	0	0	0	0	1	0	0

(3900, 33)

Categorical Column Handling

Model 3 One Hot Encoding

- Frequency of Purchases
- Region
- Age Group

BCC
Basic Computing Community

Categorical Column Handling

Using LabelEncoder For Object Columns for all Models

Model 1

	Age	Gender	Item Purchased	Category	Purchase Amount (USD)	Location	Size	Color	Season	Review Rating	Subscription Status	Shipping Type	Discount Applied	Promo Code Used	Previous Purchases	Payment Method	Frequency of Purchases	Region	Age Group
0	55	1	2	1	53	16	0	7	3	3.1	1	1	1	1	14	5	3	6	2
1	19	1	23	1	64	18	0	12	3	3.1	1	1	1	1	2	1	3	3	3
2	50	1	11	1	73	20	2	12	1	3.1	1	2	1	1	23	2	6	3	2
3	21	1	14	2	90	38	1	12	1	3.5	1	3	1	1	49	4	6	3	4
4	45	1	2	1	49	36	1	21	1	2.7	1	2	1	1	31	4	0	0	2

Model 2

	Age	Gender	Category	Purchase Amount (USD)	Size	Season	Review Rating	Subscription Status	Shipping Type	Discount Applied	Promo Code Used	Previous Purchases	Payment Method	Frequency of Purchases	Region	Age Group
0	55	1	1	53	0	3	3.1	1	1	1	1	14	5	3	6	2
1	19	1	1	64	0	3	3.1	1	1	1	1	2	1	3	3	3
2	50	1	1	73	2	1	3.1	1	2	1	1	23	2	6	3	2
3	21	1	2	90	1	1	3.5	1	3	1	1	49	4	6	3	4
4	45	1	1	49	1	1	2.7	1	2	1	1	31	4	0	0	2

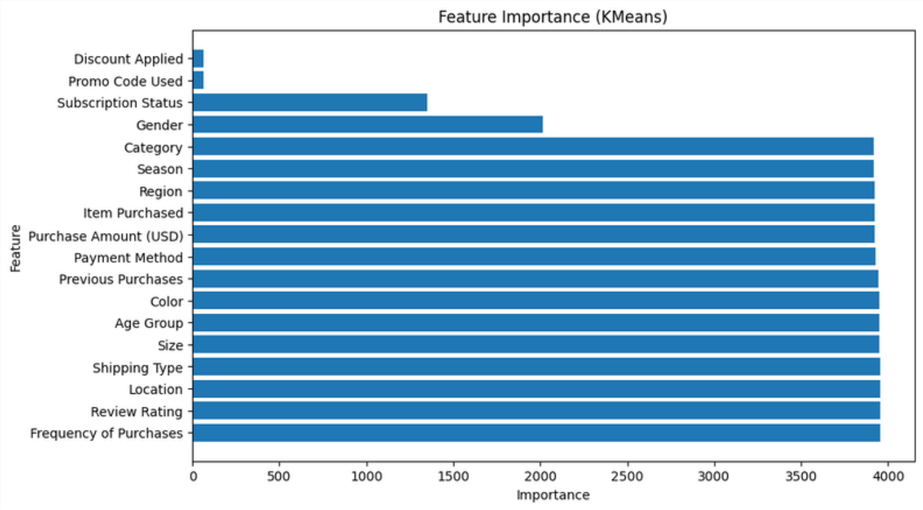
Model 3

	Age	Gender	Category	Purchase Amount (USD)	Size	Season	Review Rating	Subscription Status	Shipping Type	Discount Applied	...	New England	Plains	Rocky Mountain	Southeast	Southwest	Adult	Elder	Middle_Age	Teenager	Yo
0	55	1	1	53	0	3	3.1	1	1	1	...	0	0	0	1	0	0	0	1	0	
1	19	1	1	64	0	3	3.1	1	1	1	...	1	0	0	0	0	0	0	0	1	
2	50	1	1	73	2	1	3.1	1	2	1	...	1	0	0	0	0	0	0	1	0	
3	21	1	2	90	1	1	3.5	1	3	1	...	1	0	0	0	0	0	0	0	0	
4	45	1	1	49	1	1	2.7	1	2	1	...	0	0	0	0	0	0	0	1	0	

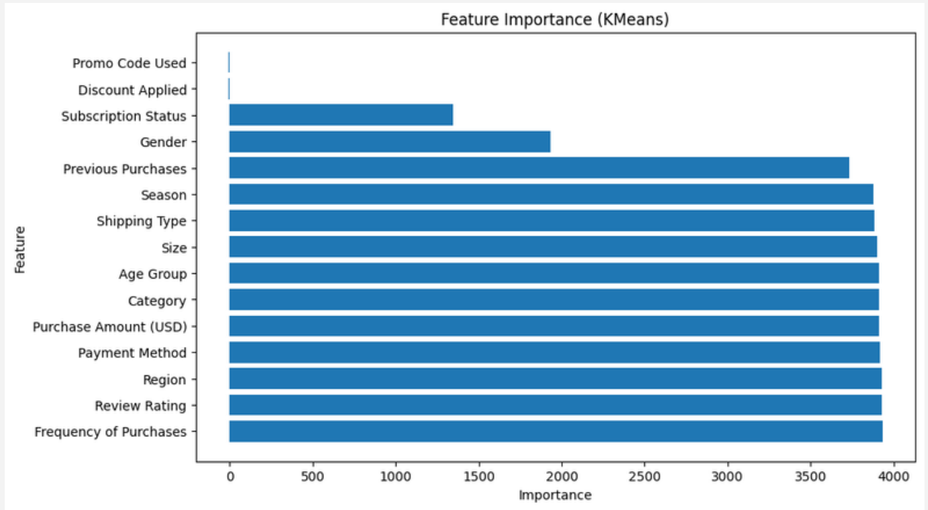
5 rows × 33 columns

BCC Basic Computing Community

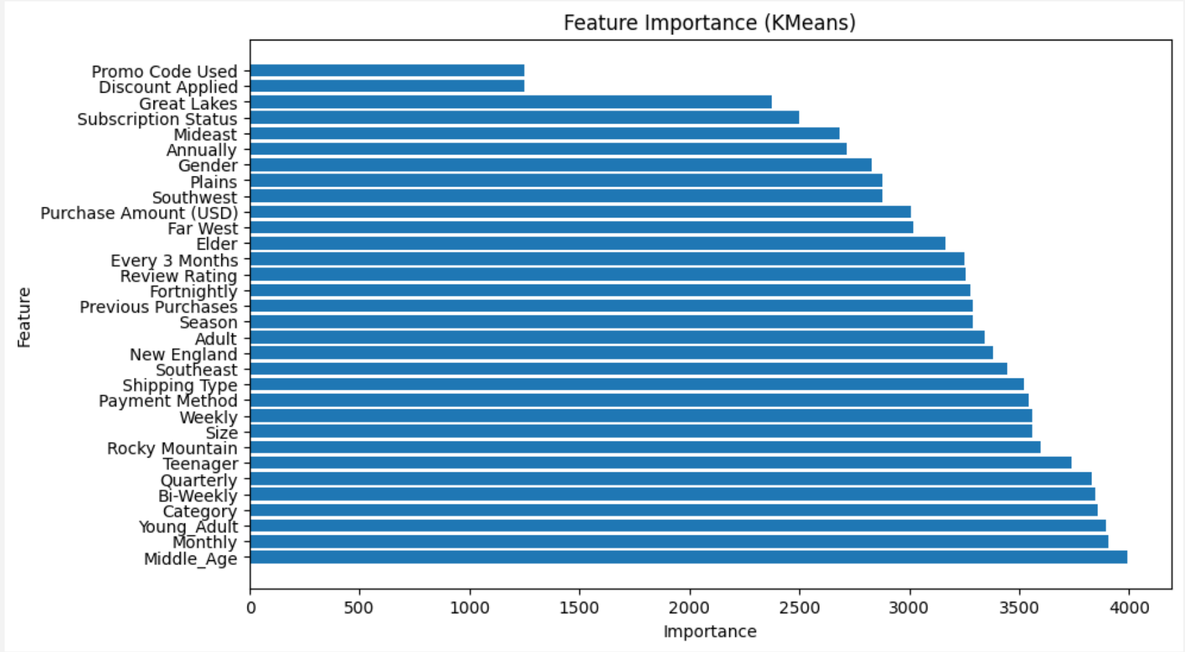
Model 1



Model 2



Model 3



Feature Selection using KMeans

BCC
Basic Computing Community

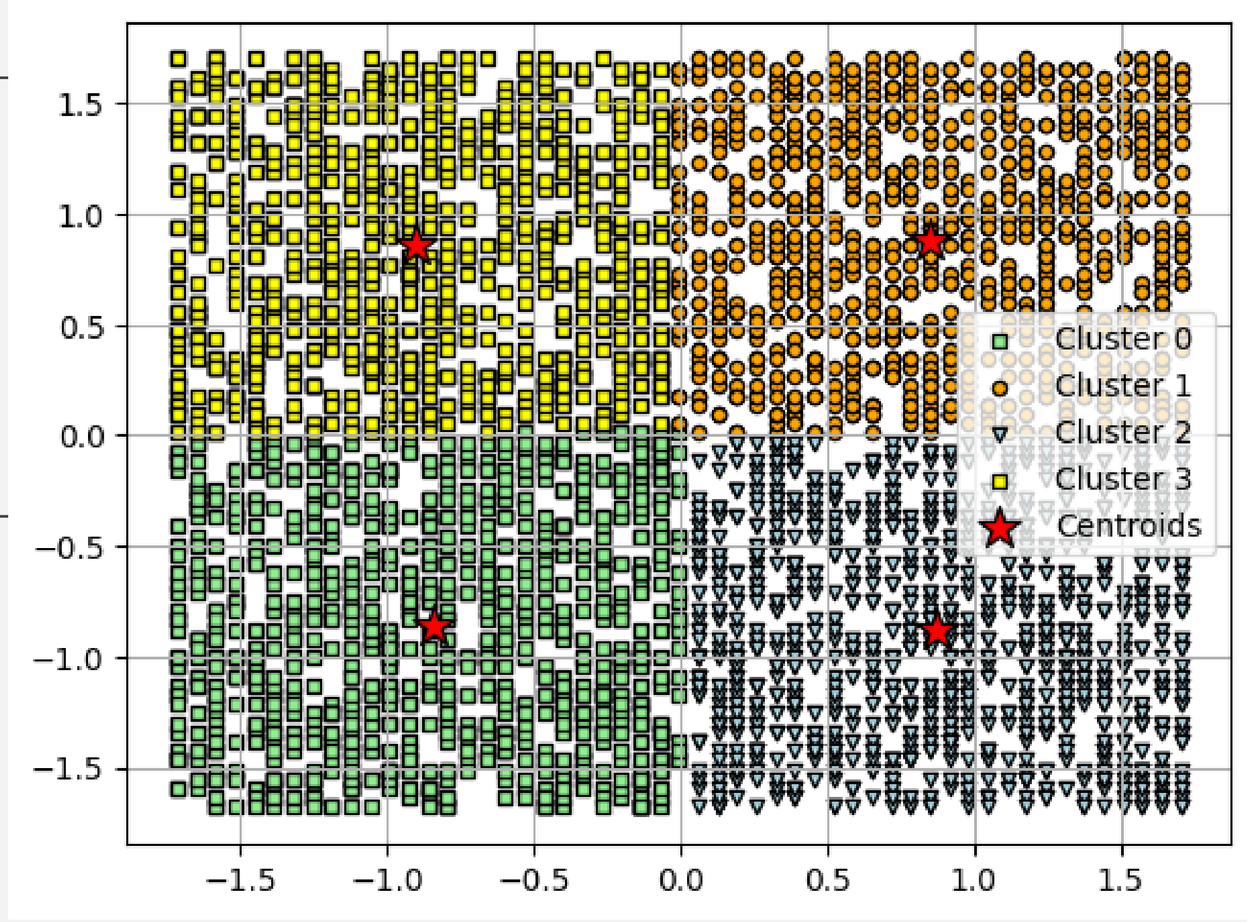
BCC
Basic Computing Community

MACHINE LEARNING IMPLEMENTATION

K-Means Clustering

BCC
Basic Computing Community

```
# Plot clusters
plt.scatter(X[X_column], X[y_column], c=y_km, s=40, cmap='viridis')
plt.title(f"Scatter Plot {X_column} Vs {y_column}")
plt.xlabel(X_column, fontsize=12)
plt.ylabel(y_column, fontsize=12)

plt.legend(y_km)
plt.show()
```

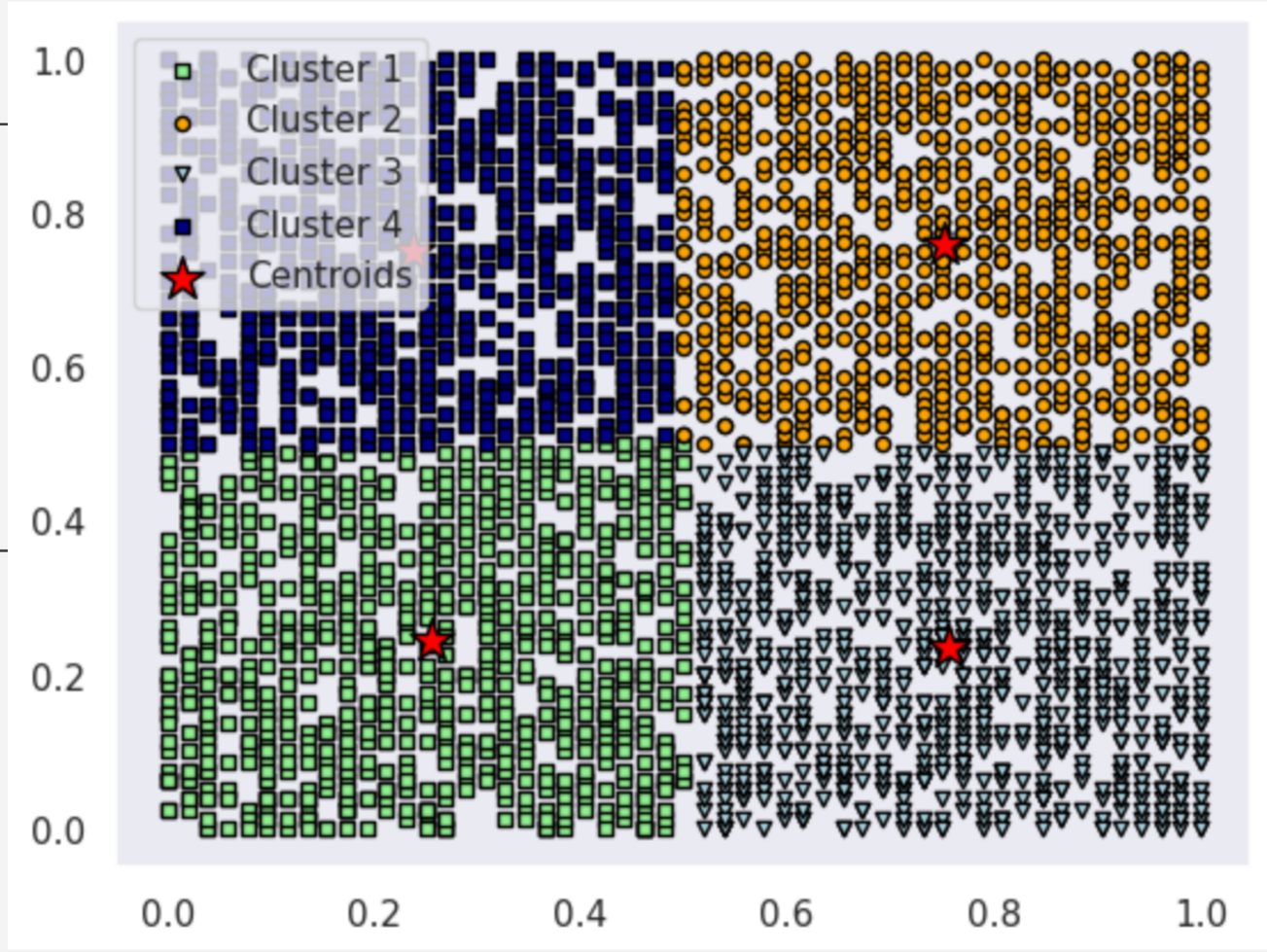


Using Standard Scaler

K-Means Clustering

```
km_mm = KMeans(n_clusters=4, random_state=45, n_init="auto")
km_mm.fit(X_scaled_mm)

y_km_mm = km_mm.predict(X_scaled_mm)
```

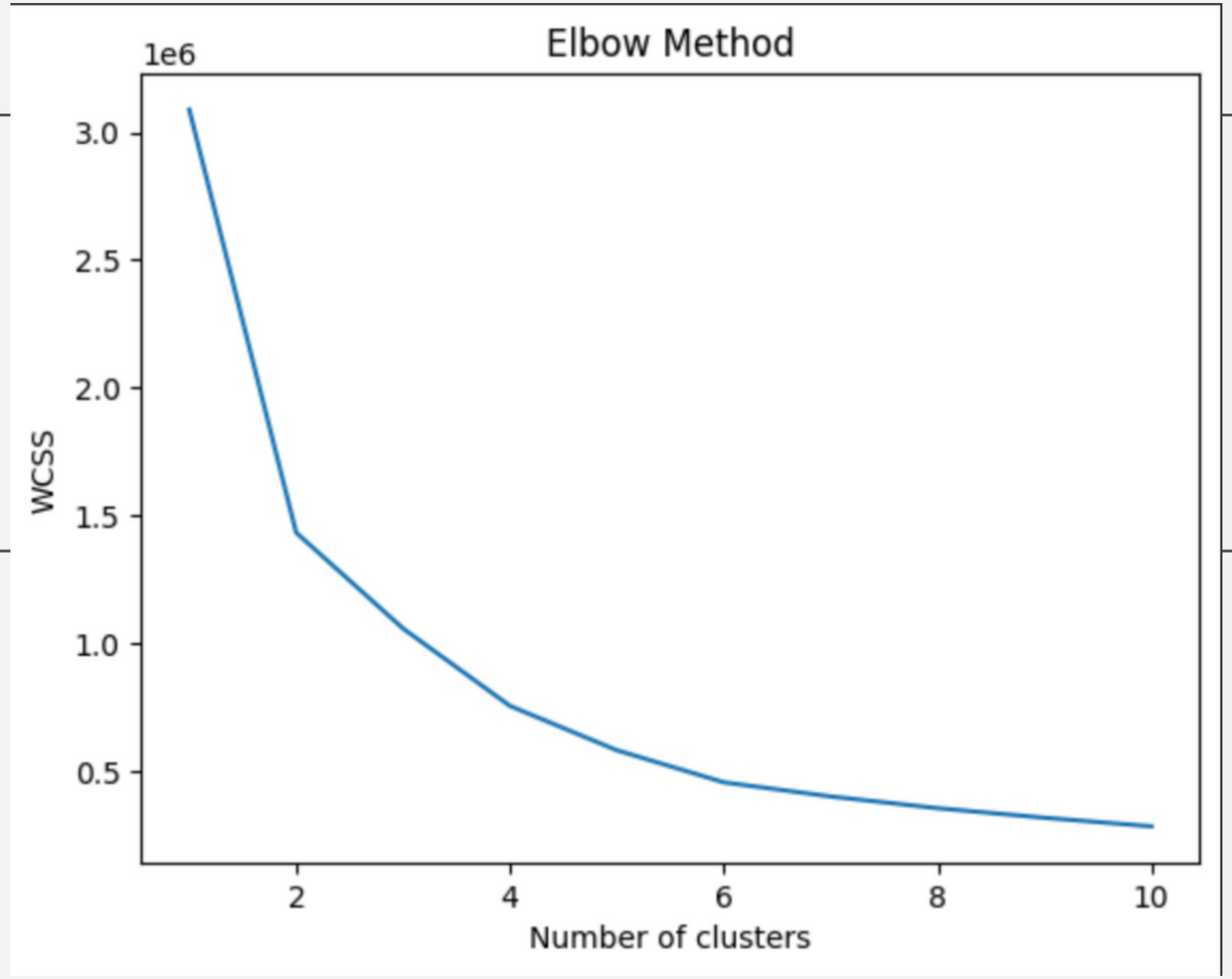
Using Min-Max Scaler



K-Means Clustering

```python
from sklearn.cluster import KMeans

features = process_data[[X_column, y_column]]

wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=500, n_init=10, random_state=0)
    kmeans.fit(features)
    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```

Elbow Method



METRIC EVALUATION

With Standard Scaler

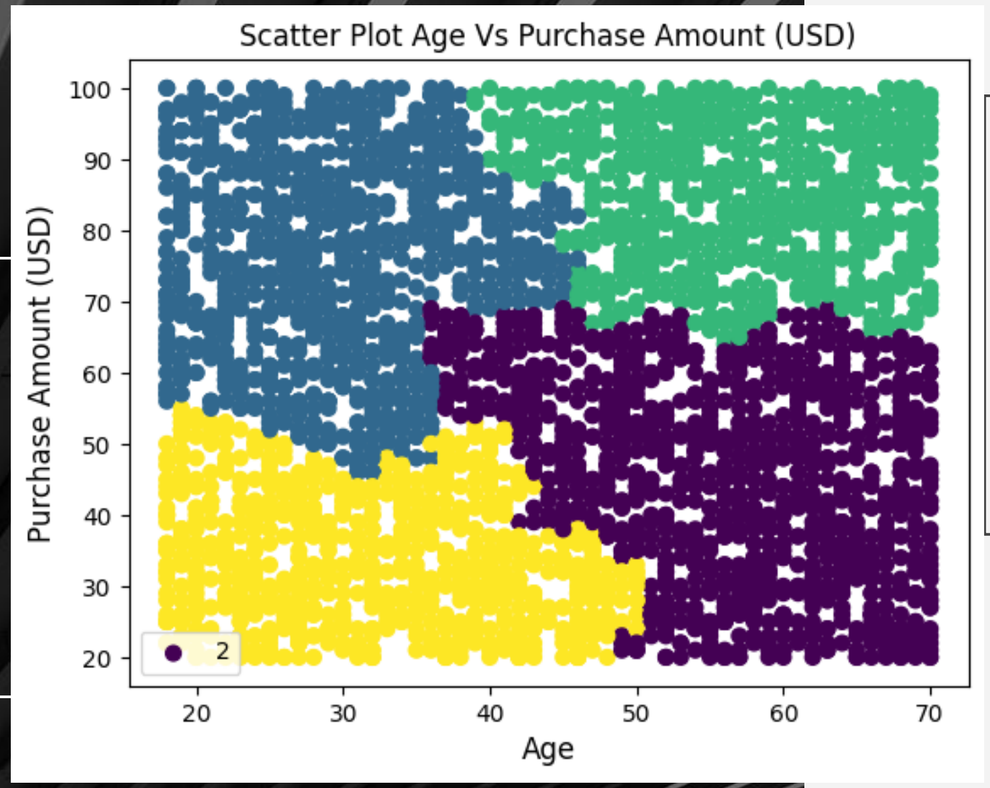
```
score = silhouette_score(X_scaled, y_km)

new_row = {"Model": f"K-Means {X_column} {y_column} SS", "Silhouette Score": score}
clus_score = pd.concat([clus_score, pd.DataFrame([new_row])], ignore_index=True)

clus_score.sort_values(by="Silhouette Score", ascending=False)
```

	Model	Silhouette Score
0	K-Means Age Purchase Amount (USD) SS	0.413762

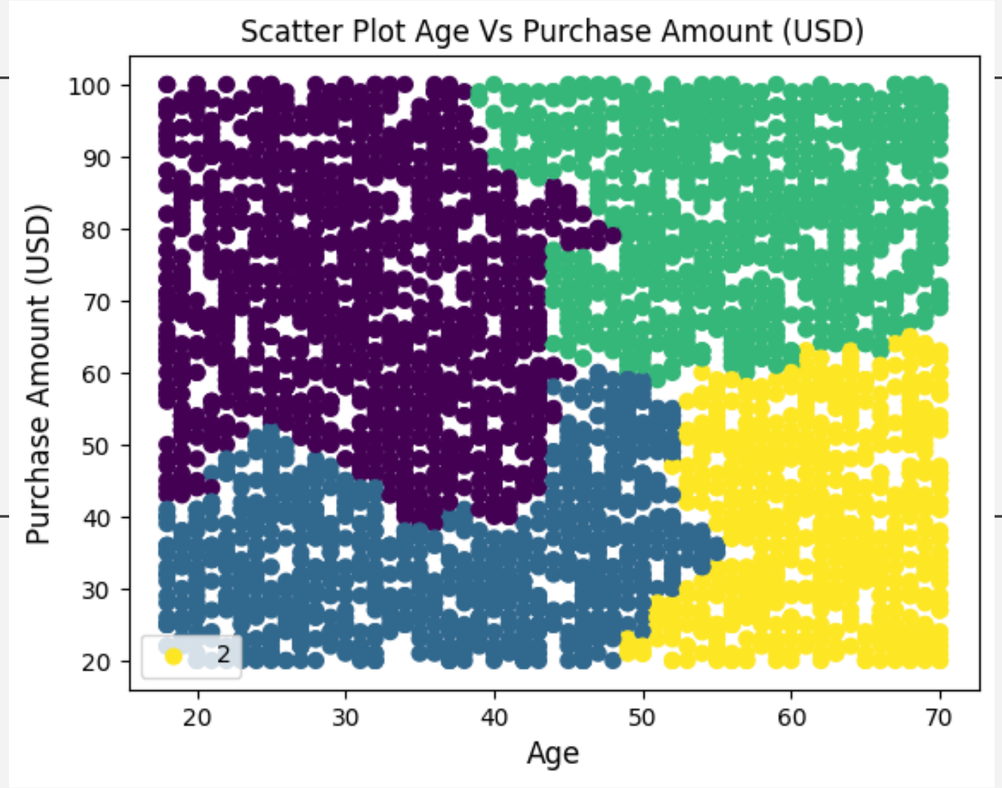
With Min-Max Scaler

	Model	Silhouette Score
0	K-Means Age Purchase Amount (USD) SS	0.413762
1	K-Means Age Purchase Amount (USD) MM	0.413725

Agglomerative (AHC) Clustering



Standard Scaler



MinMax Scaler

METRIC EVALUATION

With Standard Scaler

```
score = silhouette_score(X_scaled, y_pred)

new_row = {"Model": f"AHC {X_column} {y_column} SS", "Silhouette Score": score}
clus_score = pd.concat([clus_score, pd.DataFrame([new_row])], ignore_index=True)

clus_score.sort_values(by="Silhouette Score", ascending=False)
```

	Model	Silhouette Score
0	K-Means Age Purchase Amount (USD) SS	0.413762
1	K-Means Age Purchase Amount (USD) MM	0.413725
2	AHC Age Purchase Amount (USD) SS	0.372396

With Min-Max Scaler

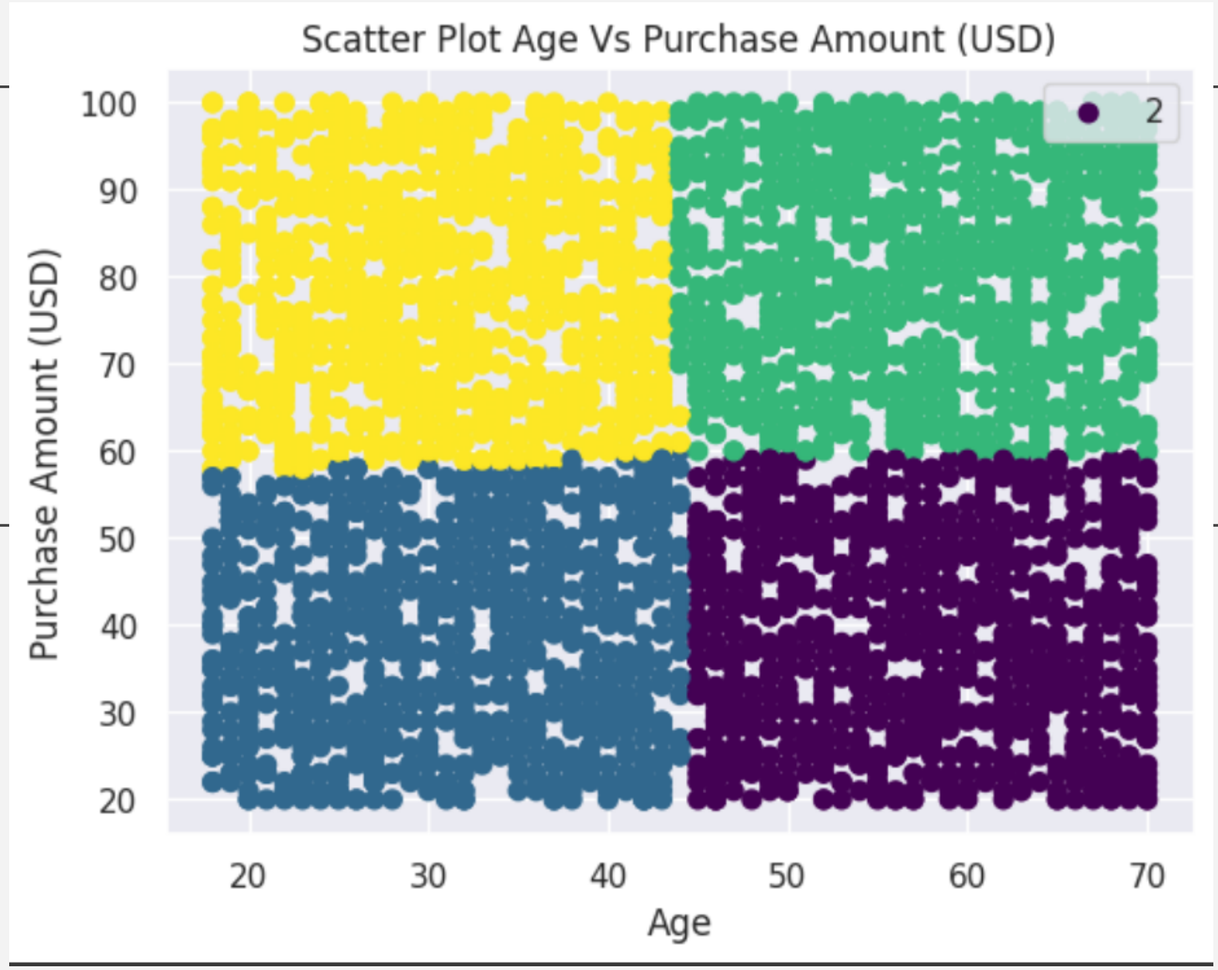
	Model	Silhouette Score
0	K-Means Age Purchase Amount (USD) SS	0.413762
1	K-Means Age Purchase Amount (USD) MM	0.413725
2	AHC Age Purchase Amount (USD) SS	0.372396
3	AHC Age Purchase Amount (USD) MM	0.350095

Gaussian Mixture Model Clustering

BCC
Basic Computing Community

```
# Create an instance of Gaussian Mixture Model
gmm = GaussianMixture(n_components=4, random_state=42)

# Fit the model to the data
gmm.fit(X_scaled)

# Predict cluster labels
labels = gmm.predict(X_scaled)

# Plot clusters
plt.scatter(X[X_column], X[y_column], c=labels, s=40, cmap='viridis')
plt.title(f"Scatter Plot {X_column} Vs {y_column}")
plt.xlabel(X_column, fontsize=12)
plt.ylabel(y_column, fontsize=12)

plt.legend(y_km)
plt.show()
```

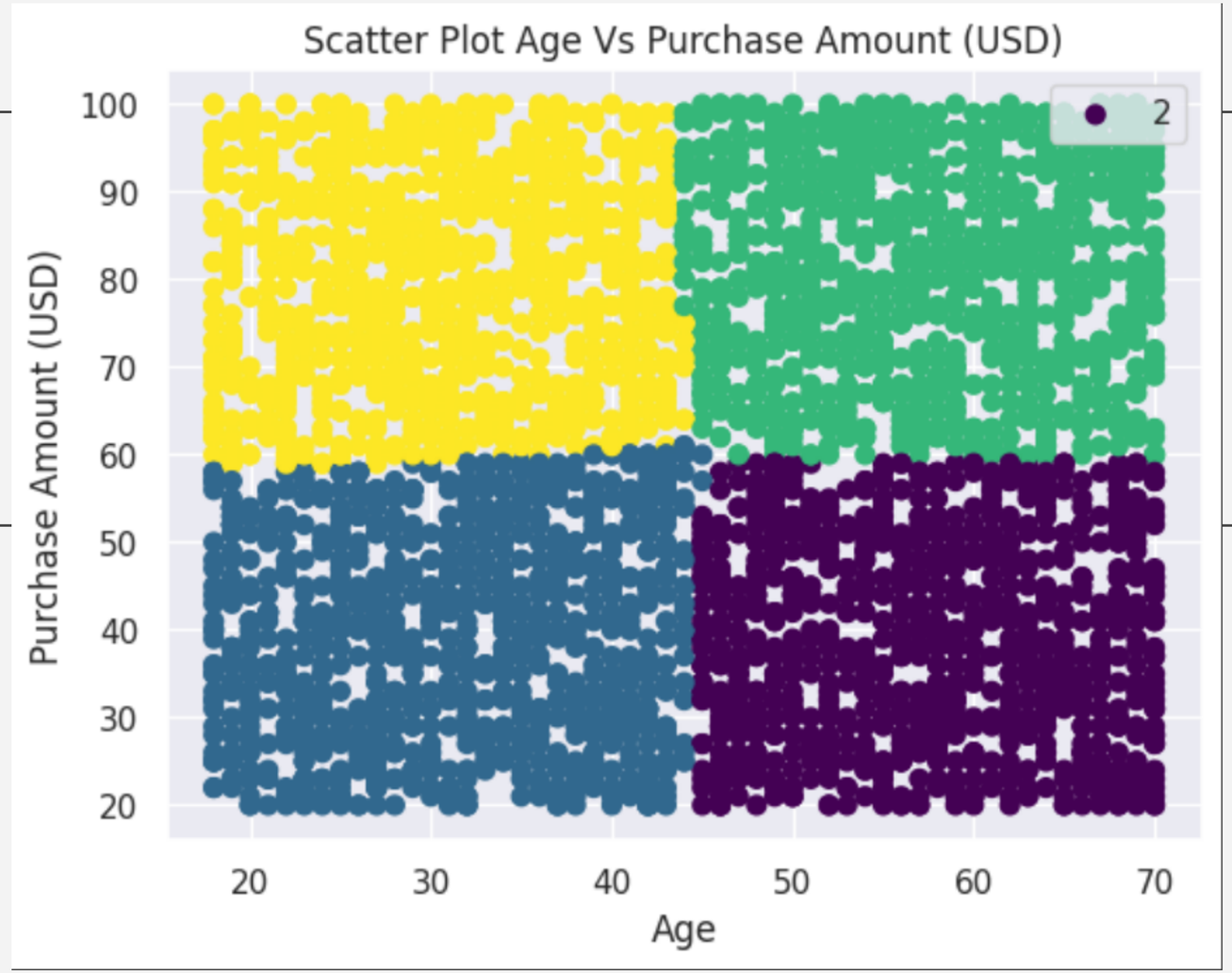
Using Standard Scaler



Gaussian Mixture Model Clustering

```
gmm = GaussianMixture(n_components=4, random_state=42)

# Fit the model to the data
gmm.fit(X_scaled_mm)

# Predict cluster labels
labels_mm = gmm.predict(X_scaled_mm)

# Plot clusters
plt.scatter(X[X_column], X[y_column], c=labels_mm, s=40, cmap='viridis')
plt.title(f"Scatter Plot {X_column} Vs {y_column}")
plt.xlabel(X_column, fontsize=12)
plt.ylabel(y_column, fontsize=12)
```

Using Min-Max Scaler



METRIC EVALUATION

With Standard Scaler

```
silhouette_avg = silhouette_score(X_scaled, labels)
print("Silhouette Score:", silhouette_avg)
```

```
Silhouette Score: 0.41362385876442215
```

With Min-Max Scaler

```
[126] silhouette_avg = silhouette_score(X_scaled_mm, labels_mm)
      print("Silhouette Score:", silhouette_avg)
```

```
Silhouette Score: 0.41361499873161905
```

CLUSTERING WITH ALL COLUMN

index	Model	Silhouette Score
13	(Nan Encoding Data) K-Means Age Purchase Amount (USD) MM	0.18108622717425715
17	(Nan Encoding Data) Gaussian Age Purchase Amount (USD) MM	0.14438121566990897
15	(Nan Encoding Data) AHC Age Purchase Amount (USD) MM	0.14024832660495704
7	(Nan Process Data) K-Means Age Purchase Amount (USD) MM	0.1398516774071249 5
11	(Nan Process Data) Gaussian Age Purchase Amount (USD) MM	0.1398516774071249 5
9	(Nan Process Data) AHC Age Purchase Amount (USD) MM	0.1398516774071249 5
12	(Nan Encoding Data) K-Means Age Purchase Amount (USD) SS	0.11644489929812246
14	(Nan Encoding Data) AHC Age Purchase Amount (USD) SS	0.11332625361246117
3	AHC Age Purchase Amount (USD) MM	0.10393431501712494
16	(Nan Encoding Data) Gaussian Age Purchase Amount (USD) SS	0.10116824307314544
6	(Nan Process Data) K-Means Age Purchase Amount (USD) SS	0.0952208825949715
10	(Nan Process Data) Gaussian Age Purchase Amount (USD) SS	0.08815866375583821
8	(Nan Process Data) AHC Age Purchase Amount (USD) SS	0.08779767459237334
1	K-Means Age Purchase Amount (USD) MM	0.08067032165953238
5	Gaussian Age Purchase Amount (USD) MM	0.07831641993729559
0	K-Means Age Purchase Amount (USD) SS	0.07110544092481591
2	AHC Age Purchase Amount (USD) SS	0.05697828500236625
4	Gaussian Age Purchase Amount (USD) SS	0.05567523670020354

- **Nan Process Data (Model 1) : KMeans MinMax (0,139851677)**
- **Nan Encoding Data (Model 2) : KMeans MinMax (0,181086227)**
- **Process Data (Model 3) : AHC MinMax (0,103934)**

CLUSTERING 5 FEATURE SELECTION

index	Model	Silhouette Score
5	Gaussian Age Purchase Amount (USD) MM	0.38351937855911594
3	AHC Age Purchase Amount (USD) MM	0.34936552860693065
2	AHC Age Purchase Amount (USD) SS	0.31054216772224486
4	Gaussian Age Purchase Amount (USD) SS	0.279555022594238
1	K-Means Age Purchase Amount (USD) MM	0.26255164661723746
0	K-Means Age Purchase Amount (USD) SS	0.2302898888290238
17	(Nan Encoding Data) Gaussian Age Purchase Amount (USD) MM	0.13031039842186648
13	(Nan Encoding Data) K-Means Age Purchase Amount (USD) MM	0.12259522262309908
12	(Nan Encoding Data) K-Means Age Purchase Amount (USD) SS	0.1224045386584888
7	(Nan Process Data) K-Means Age Purchase Amount (USD) MM	0.11611496649685828
11	(Nan Process Data) Gaussian Age Purchase Amount (USD) MM	0.11147024466443575
16	(Nan Encoding Data) Gaussian Age Purchase Amount (USD) SS	0.10490328300530806
6	(Nan Process Data) K-Means Age Purchase Amount (USD) SS	0.09744135648669207
10	(Nan Process Data) Gaussian Age Purchase Amount (USD) SS	0.09541149986272633
15	(Nan Encoding Data) AHC Age Purchase Amount (USD) MM	0.08566164239264461
9	(Nan Process Data) AHC Age Purchase Amount (USD) MM	0.07017308434592164
14	(Nan Encoding Data) AHC Age Purchase Amount (USD) SS	0.06850392391454273
8	(Nan Process Data) AHC Age Purchase Amount (USD) SS	0.05321860058175387

- **Nan Process Data (Model 1) : KMeans MinMax (0.11611496649)**
- **Nan Encoding Data (Model 2) : Gaussian MinMax (0.1303103984)**
- **Process Data (Model 3) : Gaussian MinMax (0.38351937)**

Modellling (Location)

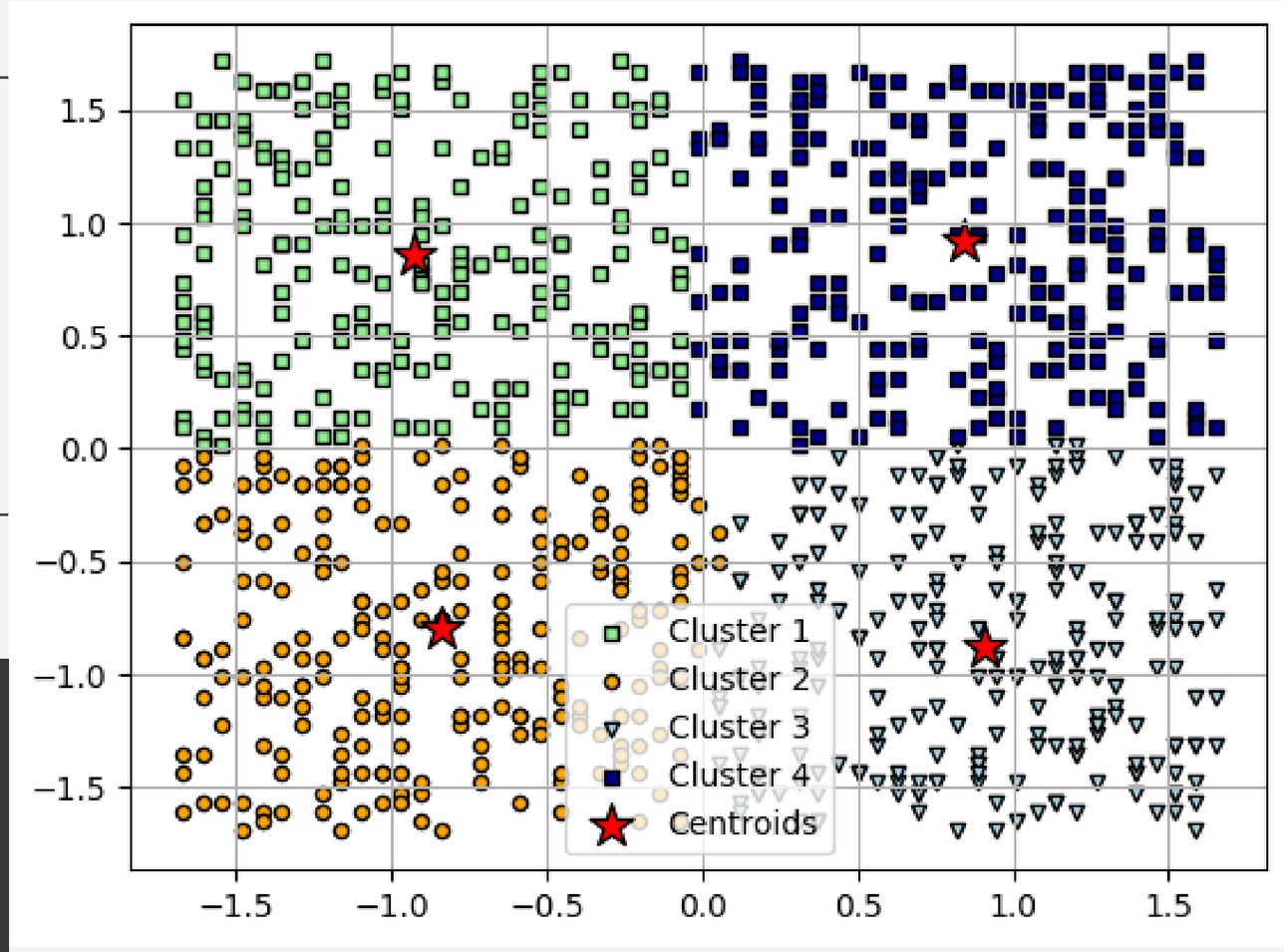
Model 1

Southeast

k-Means Clustering

Using Standard Scaler

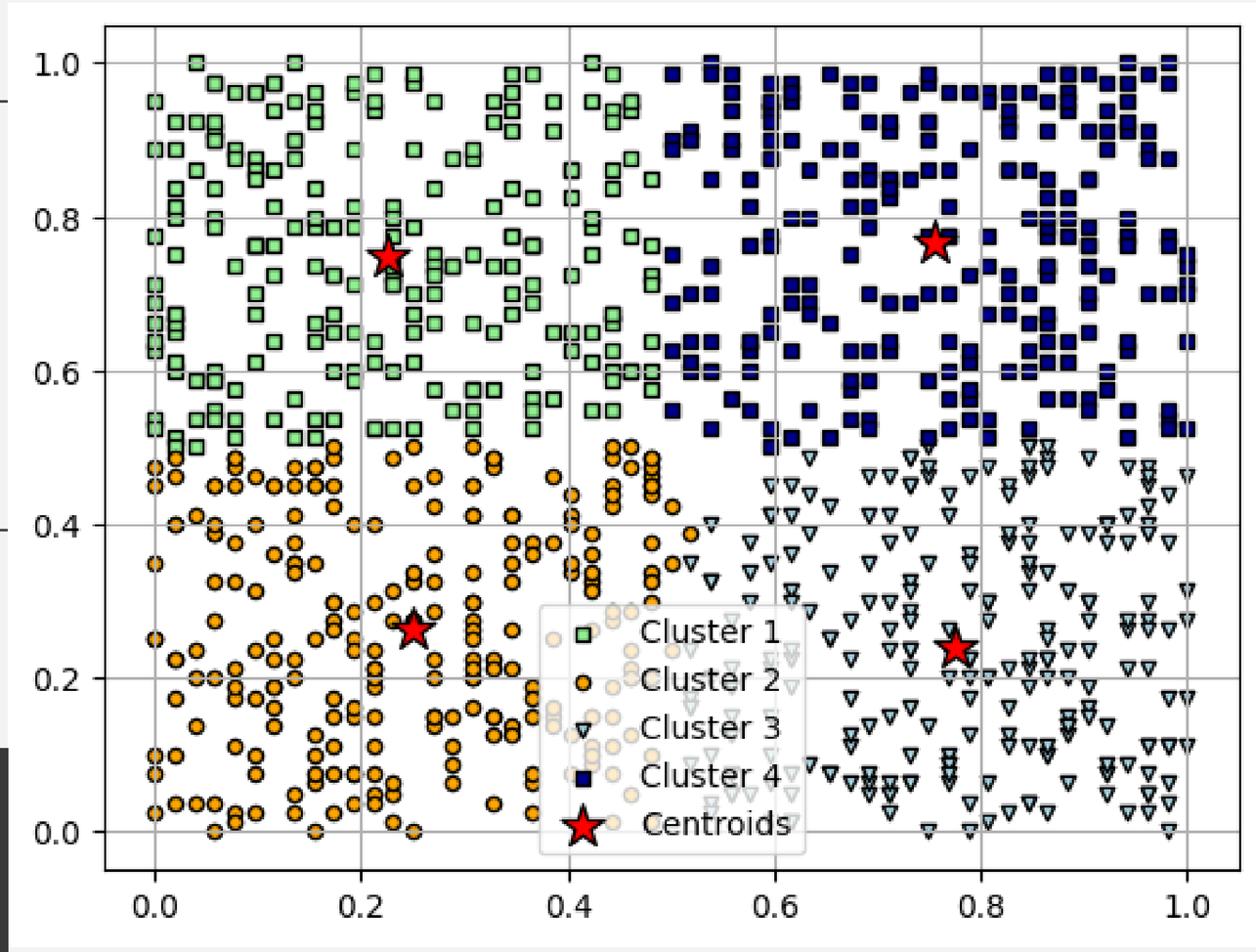
```
1    251
3    238
2    236
0    222
dtype: int64
```



k-Means Clustering

Using Min-Max Scaler

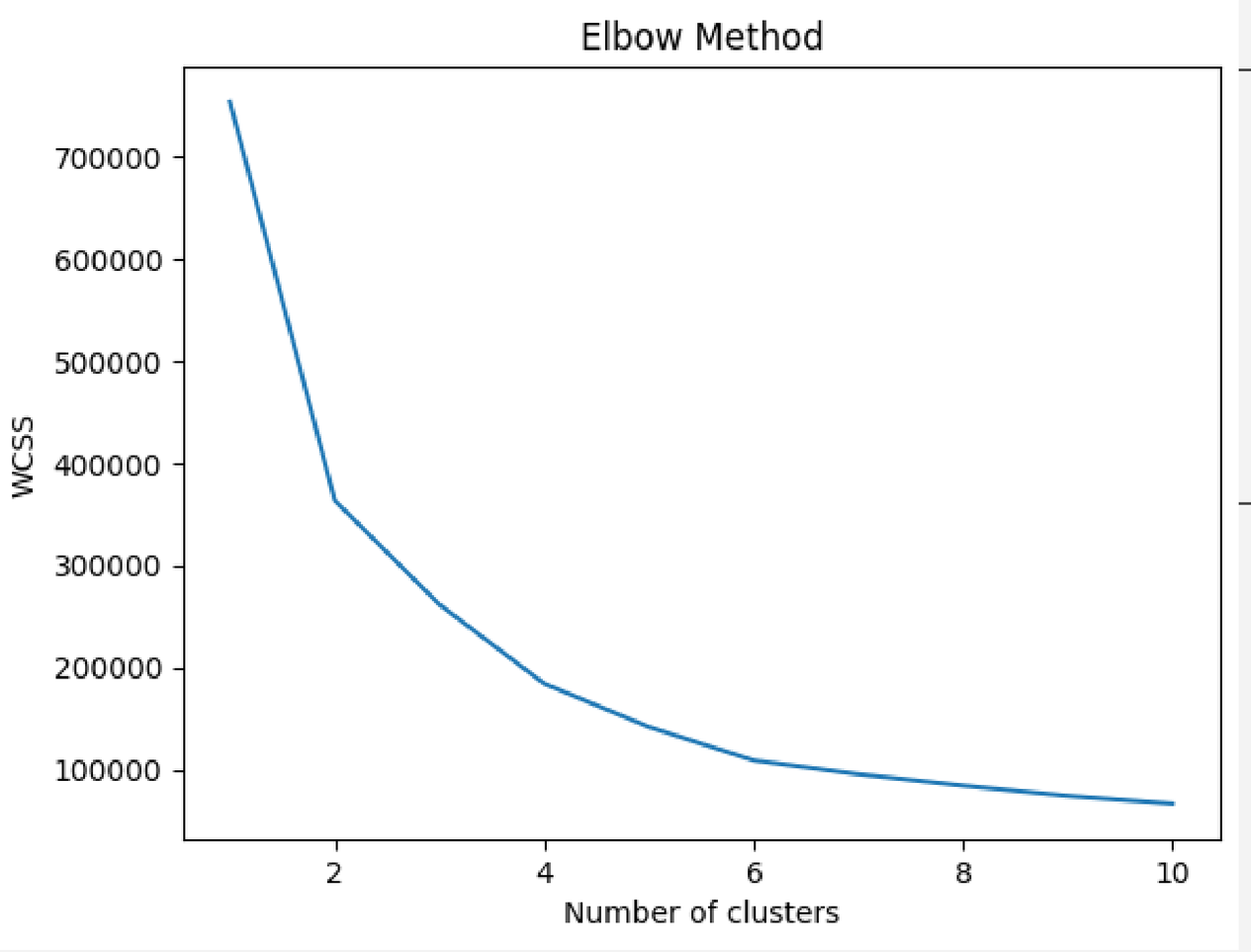
```
1    250
3    238
2    237
0    222
dtype: int64
```



k-Means Clustering

```python
from sklearn.cluster import KMeans

features = process_data[[X_column, y_column]]

wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=500, n_init=10, random_state=0)
    kmeans.fit(features)
    wcss.append(kmeans.inertia_)

plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```

Elbow Method



METRIC EVALUATION

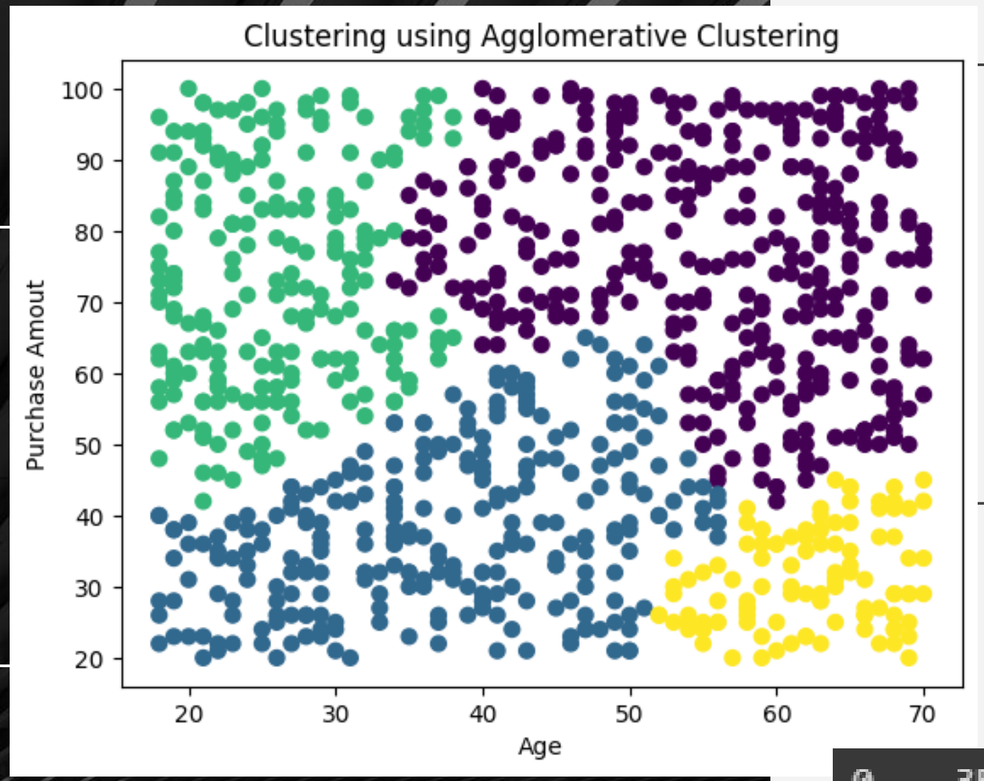
```
score = silhouette_score(X_scaled, y_km)

clus_score = pd.DataFrame(columns=["Model", "Silhouette Score"])

new_row = {"Model": "K-Means SS", "Silhouette Score": score}
clus_score = pd.concat([clus_score, pd.DataFrame([new_row])], ignore_index=True)

clus_score.sort_values(by="Silhouette Score", ascending=True)
```

```
score = silhouette_score(X_scaled_mm, y_km_mm)


new_row = {"Model": "K-Means MM", "Silhouette Score": score}
clus_score = pd.concat([clus_score, pd.DataFrame([new_row])], ignore_index=True)

clus_score.sort_values(by="Silhouette Score", ascending=True)
```

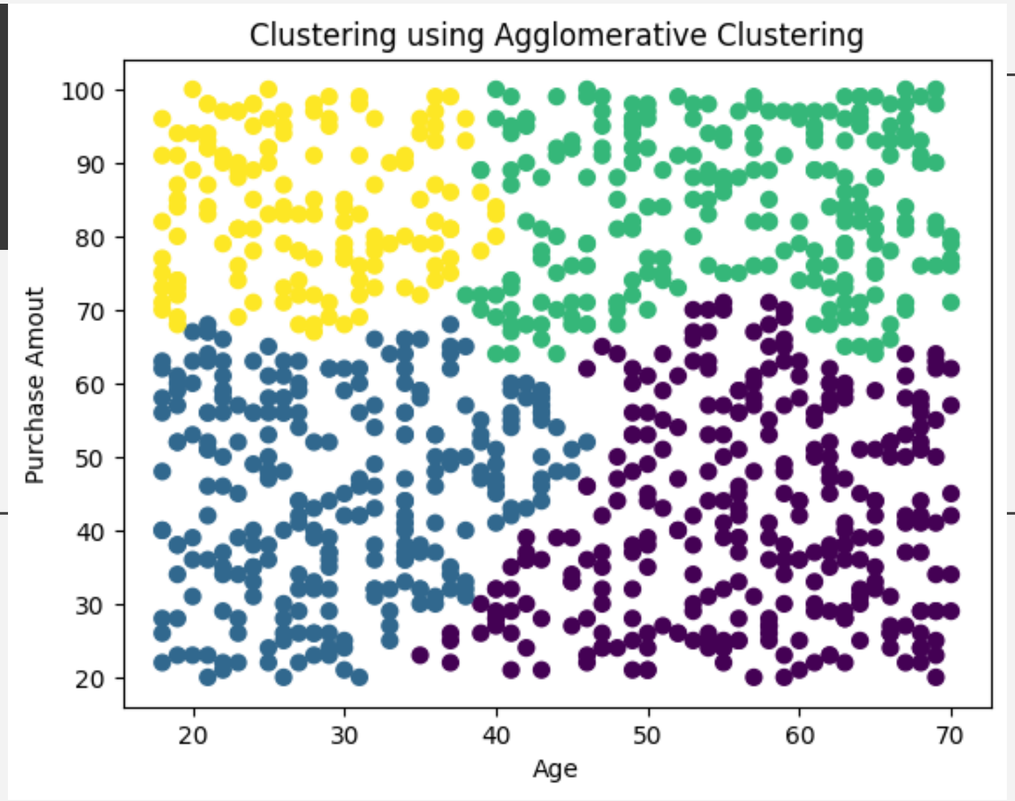
Silhouette Score

	Model	Silhouette Score
1	K-Means MM	0.417472
0	K-Means SS	0.417528

Agglomerative (AHC) Clustering




```
0    300
1    262
2    236
3    149
dtype: int64
```



Standard Scaler

```
0    350
1    270
2    221
3    106
dtype: int64
```

MinMax Scaler

METRIC EVALUATION

```
score = silhouette_score(X_scaled, y_km)

clus_score = pd.DataFrame(columns=["Model", "Silhouette Score"])

new_row = {"Model": "K-Means SS", "Silhouette Score": score}
clus_score = pd.concat([clus_score, pd.DataFrame([new_row])], ignore_index=True)

clus_score.sort_values(by="Silhouette Score", ascending=True)
```

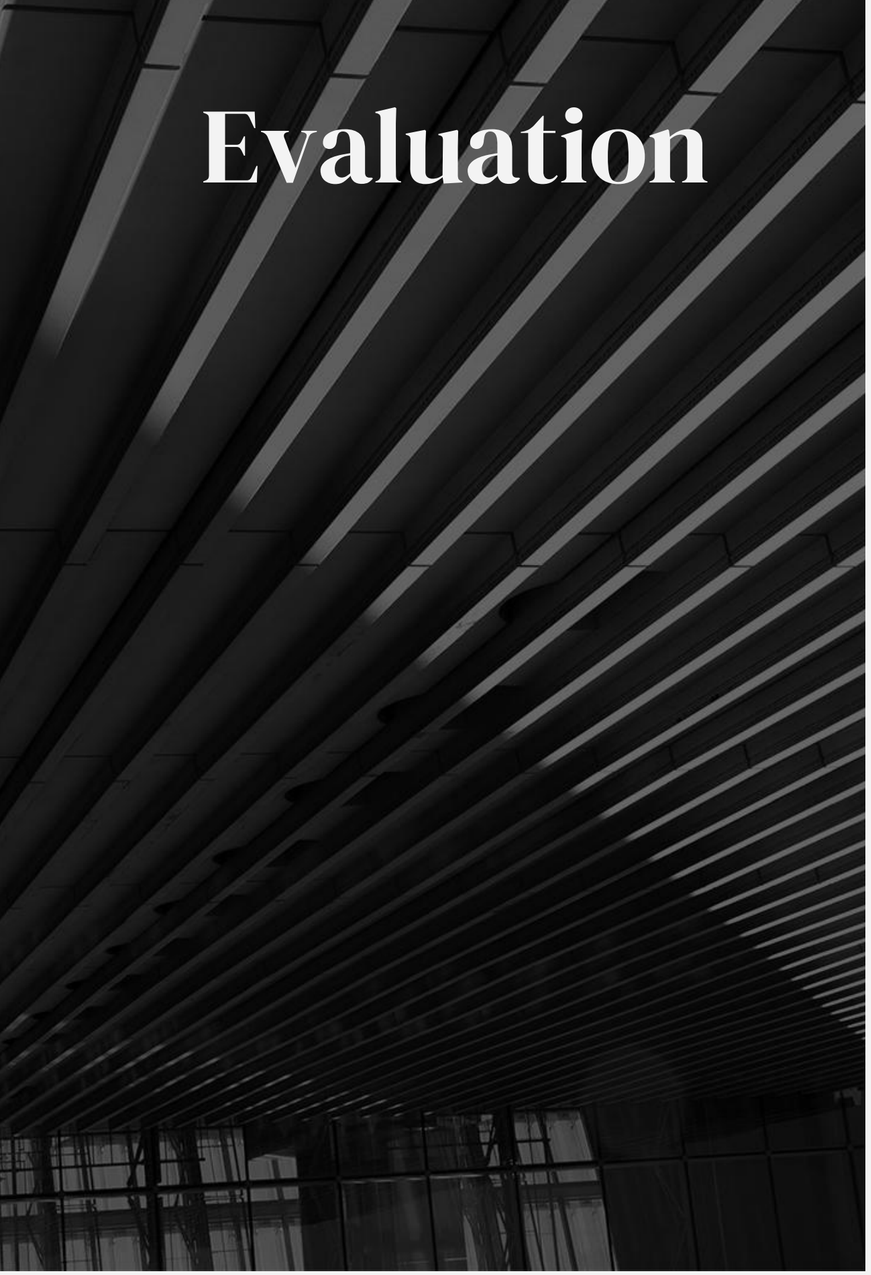
```
score = silhouette_score(X_scaled_mm, y_km_mm)


new_row = {"Model": "K-Means MM", "Silhouette Score": score}
clus_score = pd.concat([clus_score, pd.DataFrame([new_row])], ignore_index=True)

clus_score.sort_values(by="Silhouette Score", ascending=True)
```

Silhouette Score

	Model	Silhouette Score
0	K-Means SS	0.417528
1	K-Means MM	0.417472
3	AHC MM	0.383216
2	AHC SS	0.339801

BCC
Basic Computing Community

EVALUATION



Evaluation

- Memerlukan proses modelling yang sangat mendalam dengan membaginya berdasarkan method preprocessing (All preprocessing, Nan Process Data, dan Nan Encoding Data) serta berdasarkan lokasi (Southeast, Far-West, dan Southwest)
- Kebutuhan dalam melakukan preprocessing tersebut dikarenakan ingin berfokus sesuai dengan business understanding dimana melakukan segmentasi terhadap konsumer akan kecenderungan mereka dengan melihat dari usia serta bagaimana mereka melakukan kegiatan pengeluaran uang untuk pembelian produk dengan melihat pada beberapa lokasi
- Perlu dilakukan proses menggali lagi informasi yang bernilai dengan mungkin melakukan feature engineering sehingga mampu mendapatkan akurasi yang baik

Conclusion

- Tidak adanya missing values maupun outlier sehingga proses preparation data cukup mudah untuk dilakukan
- Menghandle numerical dan beberapa categorical data untuk dilakukan proses encoder yang sesuai
- Melakukan proses scaling dan mengimplementasikannya dalam proses pemodelan Machine Learning Clustering serta mendapatkan nilai skornya

Things to Improve

Feature Engineering terhadap beberapa kolom

BCC
Basic Computing Community

THANK YOU

Feel free if you have any questions