APPLIED DATA SCIENCE

**CUSTOMER SEGMENTATION USING DATA SCIENCE-PHASE 4**

**DEVELOPMENT PART 2**

Continue building the customer segmentation model by:

* Feature engineering
* Applying clustering algorithms
* Visualization
* Interpretation.

CODING:

**import** pandas **as** pd

**import** numpy **as** np

**import** seaborn **as** sns

**import** matplotlib.pyplot **as** plt

**from** sklearn.decomposition **import** PCA

**from** sklearn.preprocessing **import** StandardScaler

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn.linear\_model **import** LinearRegression

**from** sklearn.metrics **import** mean\_squared\_error, r2\_score

**from** sklearn.tree **import** DecisionTreeRegressor

**from** sklearn.metrics **import** mean\_squared\_error

**from** sklearn.svm **import** SVR

**from** sklearn.linear\_model **import** LinearRegression

dataset**=** pd**.**read\_csv("C:\\Users\\sowen\\Downloads\\Mall\_Customers.csv")

dataset

OUTPUT:

| **CustomerID** | **Genre** | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
| --- | --- | --- | --- | --- |
| **0** | 1 | Male | 19 | 15 | 39 |
| **1** | 2 | Male | 21 | 15 | 81 |
| **2** | 3 | Female | 20 | 16 | 6 |
| **3** | 4 | Female | 23 | 16 | 77 |
| **4** | 5 | Female | 31 | 17 | 40 |
| **...** | ... | ... | ... | ... | ... |
| **195** | 196 | Female | 35 | 120 | 79 |
| **196** | 197 | Female | 45 | 126 | 28 |
| **197** | 198 | Male | 32 | 126 | 74 |
| **198** | 199 | Male | 32 | 137 | 18 |
| **199** | 200 | Male | 30 | 137 | 83 |

200 rows × 5 columns

***Data Exploration***

Dataset

**OUTPUT**:

|  | **CustomerID** | **Genre** | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
| --- | --- | --- | --- | --- | --- |
| **0** | 1 | Male | 19 | 15 | 39 |
| **1** | 2 | Male | 21 | 15 | 81 |
| **2** | 3 | Female | 20 | 16 | 6 |
| **3** | 4 | Female | 23 | 16 | 77 |
| **4** | 5 | Female | 31 | 17 | 40 |
| **...** | ... | ... | ... | ... | ... |
| **195** | 196 | Female | 35 | 120 | 79 |
| **196** | 197 | Female | 45 | 126 | 28 |
| **197** | 198 | Male | 32 | 126 | 74 |
| **198** | 199 | Male | 32 | 137 | 18 |
| **199** | 200 | Male | 30 | 137 | 83 |

200 rows × 5 columns

dataset**.**info()

**OUTPUT:**

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 200 entries, 0 to 199

Data columns (total 5 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 CustomerID 200 non-null int64

1 Genre 200 non-null object

2 Age 200 non-null int64

3 Annual Income (k$) 200 non-null int64

4 Spending Score (1-100) 200 non-null int64

dtypes: int64(4), object(1)

memory usage: 7.9+ KB

dataset**.**drop('CustomerID', axis**=**1, inplace**=True**)

dataset**.**drop('Genre', axis**=**1, inplace**=True**)

dataset

OUTPUT:

|  | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
| --- | --- | --- | --- |
| **0** | 19 | 15 | 39 |
| **1** | 21 | 15 | 81 |
| **2** | 20 | 16 | 6 |
| **3** | 23 | 16 | 77 |
| **4** | 31 | 17 | 40 |
| **...** | ... | ... | ... |
| **195** | 35 | 120 | 79 |
| **196** | 45 | 126 | 28 |
| **197** | 32 | 126 | 74 |
| **198** | 32 | 137 | 18 |
| **199** | 30 | 137 | 83 |

200 rows × 3 columns

dataset**.**describe()

**OUTPUT:**

|  | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
| --- | --- | --- | --- |
| **count** | 200.000000 | 200.000000 | 200.000000 |
| **mean** | 38.850000 | 60.560000 | 50.200000 |
| **std** | 13.969007 | 26.264721 | 25.823522 |
| **min** | 18.000000 | 15.000000 | 1.000000 |
| **25%** | 28.750000 | 41.500000 | 34.750000 |
| **50%** | 36.000000 | 61.500000 | 50.000000 |
| **75%** | 49.000000 | 78.000000 | 73.000000 |
| **max** | 70.000000 | 137.000000 | 99.000000 |

dataset**.**columns

OUTPUT:

Index(['Age', 'Annual Income (k$)', 'Spending Score (1-100)'], dtype='object')

**Data Visualization**

**1.Histogram**

plt**.**figure(figsize**=**(10, 6))

plt**.**hist(dataset['Age'], bins**=**20, color**=**'blue', alpha**=**0.7)

plt**.**title('Age based customer segmentation')

plt**.**xlabel('Age')

plt**.**ylabel('Gender')

plt**.**show()

**2. Bar chart**

plt**.**figure(figsize**=**(12,6))

sns**.**pairplot(dataset)

dataset**.**hist(figsize**=**(10,8))

**Visualising Correlation**

dataset**.**corr()

OUTPUT:

|  | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
| --- | --- | --- | --- |
| **Age** | 1.000000 | -0.012398 | -0.327227 |
| **Annual Income (k$)** | -0.012398 | 1.000000 | 0.009903 |
| **Spending Score (1-100)** | -0.327227 | 0.009903 | 1.000000 |

plt**.**figure(figsize**=**(10,5))

sns**.**heatmap(dataset**.**corr(), annot**=True**)

**By using PCA Algorithm**

X **=** dataset**.**drop('Spending Score (1-100)', axis**=**1)

y **=** dataset['Spending Score (1-100)']

scaler **=** StandardScaler()

X\_scaled **=** scaler**.**fit\_transform(X)

n\_components **=** 2

pca **=** PCA(n\_components**=**n\_components)

X\_pca **=** pca**.**fit\_transform(X\_scaled)

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X\_pca, y, test\_size**=**0.2, random\_state**=**42)

model **=** LinearRegression()

model**.**fit(X\_train, y\_train)

y\_pred **=** model**.**predict(X\_test)

mse **=** mean\_squared\_error(y\_test, y\_pred)

print(f"Mean Squared Error: {mse}")

Mean Squared Error: 483.55682175408344

**Training model:1\_ Decision Tree Regressor**

In [19]:

decision\_tree **=** DecisionTreeRegressor()

decision\_tree**.**fit(X\_train, y\_train)

y\_pred **=** decision\_tree**.**predict(X\_test)

mse **=** mean\_squared\_error(y\_test, y\_pred)

print("Mean Squared Error:", mse)

Mean Squared Error: 1073.35

**Training model:2\_Support vector machine algorithm**

svm **=** SVR(kernel**=**'rbf', C**=**1.0, gamma**=**'scale')

svm**.**fit(X\_train, y\_train)

SVR()

y\_pred **=** svm**.**predict(X\_test)

mse **=** mean\_squared\_error(y\_test, y\_pred)

print(f"Mean Squared Error: {mse}")

Mean Squared Error: 524.8207652112172

**Training model:3\_Linear Regression**

regression\_model **=** LinearRegression()

regression\_model**.**fit(X\_train, y\_train)

y\_pred **=** regression\_model**.**predict(X\_test)

mse **=** mean\_squared\_error(y\_test, y\_pred)

r2 **=** r2\_score(y\_test, y\_pred)

print(f"Mean Squared Error (MSE): {mse}")

print(f"R-squared (R2): {r2}")

Mean Squared Error (MSE): 483.55682175408344

R-squared (R2): 0.01963177813218009