This project involves applying K-means clustering to segment customer data from the Mall Customer Segmentation Data. The dataset includes attributes such as customer ID, gender, age, annual income, and spending score. The analysis aims to discover distinct customer segments based on shopping behavior and profile these segments for business insights.

WEEK 3

Unsupervised Learning & Feature Engineering

ML-WEEK 3.1

Noor Ul Ain

TASK 3.1: CLUSTERING WITH K-MEANS

OBJECTIVE

Apply K-means clustering to segment customer data and analyze patterns without predefined labels.

DATASET

Mall Customer Segmentation Data from Kaggle. This dataset includes information like customer ID, gender, age, annual income, and spending score, which are used to segment customers based on shopping behaviour.

DATASET URL

https://www.kaggle.com/vjchoudhary7/customer-segmentation tutorial-in-python

ACTIVITIES

1. Data Exploration:
- Load the dataset and conduct basic exploratory data analysis to understand the features.

2. Clustering Implementation:
- Use the K-means algorithm from scikit-learn to find clusters among customers.
- Determine the optimal number of clusters using the elbow method.

3. Analysis of Clusters:
- Analyze and profile the characteristics of each customer cluster.
- Visualize the clusters using scatter plots by choosing appropriate feature pairs.

CODE

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import silhouette_score, davies_bouldin_score, calinski_harabasz_score,
adjusted_rand_score, adjusted_mutual_info_score
import warnings
warnings.filterwarnings('ignore', category=FutureWarning)
warnings.filterwarnings('ignore', message="When grouping with a length-1 list-like")

# Load the dataset
df = pd.read_csv(r'E:\A&D_Intership\ML-WEEK 3.1\pythonProject1\mall_customers.csv')

# Display the first few rows of the dataset
print("First few rows of the dataset:")
print(df.head())

# Display the last few rows of the dataset
print("\nLast few rows of the dataset:")
print(df.tail())

# Display the shape of the dataset
print("\nShape of the dataset:")
print(df.shape)

# Display the information about the dataset
print("\nInformation about the dataset:")
print(df.info())

# Display the statistical summary of the dataset
print("\nStatistical summary of the dataset:")
print(df.describe())


# Data Exploration
print("\nData Exploration")
fig = px.histogram(df, x='Age', nbins=10, title='Distribution of Age')
fig.show()
```

```
fig = px.histogram(df, x='Annual Income (k$)', nbins=10, title='Distribution of Annual Income')
fig.show()

fig = px.histogram(df, x='Spending Score (1-100)', nbins=10, title='Distribution of Spending
Score')
fig.show()

fig = px.scatter(df, x='Age', y='Annual Income (k$)', color='Gender',
hover_data=['CustomerID'])
fig.update_layout(title='Age vs Annual Income')
fig.show()

fig = px.scatter(df, x='Annual Income (k$)', y='Spending Score (1-100)', color='Gender',
hover_data=['CustomerID'])
fig.update_layout(title='Annual Income vs Spending Score')
fig.show()

fig = px.box(df, x='Gender', y='Annual Income (k$)', points='all', title='Annual Income by
Gender')
fig.show()

fig = px.box(df, x='Gender', y='Spending Score (1-100)', points='all', title='Spending Score by
Gender')
fig.show()

# Check for missing values
print("\nChecking for missing values:")
print(df.isnull().sum())

# Encode the 'Gender' column
label_encoder = LabelEncoder()
df['Gender'] = label_encoder.fit_transform(df['Gender'])

# Select features for clustering
X = df[['Annual Income (k$)', 'Spending Score (1-100)']]

# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Determine the optimal number of clusters using the Elbow Method
wcss = []
for i in range(1, 11):
```

```
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10,
random_state=42)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)

plt.figure(figsize=(10, 5))
plt.plot(range(1, 11), wcss, marker='o')
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()

# Apply K-means algorithm
kmeans = KMeans(n_clusters=5, init='k-means++', max_iter=300, n_init=10, random_state=42)
y_kmeans = kmeans.fit_predict(X_scaled)

# Add the cluster labels to the dataset
df['Cluster'] = y_kmeans

# Visualize the clusters
plt.figure(figsize=(10, 7))
sns.scatterplot(x='Annual Income (k$)', y='Spending Score (1-100)', hue='Cluster', data=df,
palette='viridis', s=100, alpha=0.7)
plt.title('Clusters of Customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()

# Evaluate the clustering performance
print("\nClustering Performance Evaluation")

inertia = kmeans.inertia_
print(f"Inertia (WCSS): {inertia}")

silhouette_avg = silhouette_score(X_scaled, y_kmeans)
print(f"Silhouette Score: {silhouette_avg}")

davies_bouldin = davies_bouldin_score(X_scaled, y_kmeans)
print(f"Davies-Bouldin Index: {davies_bouldin}")

calinski_harabasz = calinski_harabasz_score(X_scaled, y_kmeans)
print(f"Calinski-Harabasz Index: {calinski_harabasz}")
```

```
# Analyze and profile the characteristics of each cluster
print("\nCluster Analysis")

for cluster in df['Cluster'].unique():
    print(f"\nCluster {cluster} Characteristics:")
    print(df[df['Cluster'] == cluster].describe())
```

EXPLANATION

Pandas (pd):

- **Description:** Pandas is a powerful data manipulation and analysis library for Python.
- **Use in Project:** It is used to load, manipulate, and analyze the dataset. Functions like read_csv, head, tail, shape, info, and describe help in understanding the dataset structure and its statistical properties.

NumPy (np):

- **Description:** NumPy is a library for numerical computations in Python, providing support for arrays and matrices along with a collection of mathematical functions.
- **Use in Project:** It is used for numerical operations, such as creating arrays and performing computations on them.

Matplotlib (plt):

- **Description:** Matplotlib is a plotting library for creating static, animated, and interactive visualizations in Python.
- **Use in Project:** It is used to create plots like the Elbow Method graph to determine the optimal number of clusters.

Seaborn (sns):

- **Description:** Seaborn is a statistical data visualization library based on Matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- **Use in Project:** It is used to create scatter plots for visualizing customer clusters.

Plotly (px):

- **Description:** Plotly is a graphing library that makes interactive, publication-quality graphs online.
- **Use in Project:** It is used for creating interactive histograms, scatter plots, and box plots to explore the dataset visually.

Scikit-learn (sklearn):

- **Description:** Scikit-learn is a machine learning library in Python that provides simple and efficient tools for data mining and data analysis.
- **Use in Project:** It is used for applying the K-means clustering algorithm, standardizing features with StandardScaler, and encoding categorical data with LabelEncoder. Metrics like silhouette_score, davies_bouldin_score, and calinski_harabasz_score are used to evaluate the clustering performance.

Warnings (warnings):

- **Description:** This library is used to control the display of warning messages in Python.
- **Use in Project:** It is used to suppress specific warnings to keep the output clean and focused on the relevant results.

The project is divided into several parts:

1. Importing Libraries and Loading Data

We start by importing necessary libraries, including Pandas for data manipulation, NumPy for numerical operations, Matplotlib and Seaborn for visualization, Plotly for interactive plots, and scikit-learn for machine learning tasks. The dataset is then loaded into a Pandas DataFrame.

2. Initial Data Exploration

We display the first and last few rows of the dataset, its shape, and basic information to understand the data structure. A statistical summary provides insights into the distribution and range of values in each column.

3. Data Exploration with Visualizations

We generate various plots to explore the data:

- Histograms of age, annual income, and spending score to understand their distributions.
- Scatter plots to visualize relationships between age and annual income, and annual income and spending score, color-coded by gender.
- Box plots to compare annual income and spending score distributions across genders.

4. Data Preprocessing

We check for missing values and encode the 'Gender' column using LabelEncoder to convert categorical data into numerical format. The features 'Annual Income (k$)' and 'Spending Score (1-100)' are selected for clustering and standardized using StandardScaler.

5. Determining the Optimal Number of Clusters

The Elbow Method is used to find the optimal number of clusters by plotting the within-cluster sum of squares (WCSS) for different cluster numbers. The plot helps identify the point where increasing the number of clusters does not significantly reduce WCSS, indicating the optimal cluster count.

6. Applying K-Means Algorithm

We apply the K-means algorithm with the determined optimal number of clusters. The cluster labels are added to the dataset.

7. Visualizing Clusters

We create a scatter plot using Seaborn to visualize the clusters formed based on annual income and spending score. Each cluster is color-coded for clear differentiation.

8. Evaluating Clustering Performance

Various metrics such as inertia (WCSS), silhouette score, Davies-Bouldin index, and Calinski-Harabasz index are calculated to evaluate the clustering performance.

9. Cluster Analysis

We analyze and profile each cluster by printing the statistical summary for the data points belonging to each cluster. This helps in understanding the characteristics and behavior of customers in each segment.