SOMAIYA
VIDYAVIHAR
K J Somaiya Institute of Technology
An Autonomous Institute Permanently Affiliated to the University of Mumbai

DEPARTMENT OF INFORMATION TECHNOLOGY

Course: Data Mining & Business Intelligence Lab (ITL601)

B.Tech. (Information Technology) – Semester VI

Academic Year: 2022-23 (Even Semester)

PRACTICAL 10

Aim: Consider a real-world application and perform KDD process on it and extract BI from it.

Lab Objective: To identify and compare the performance of business.

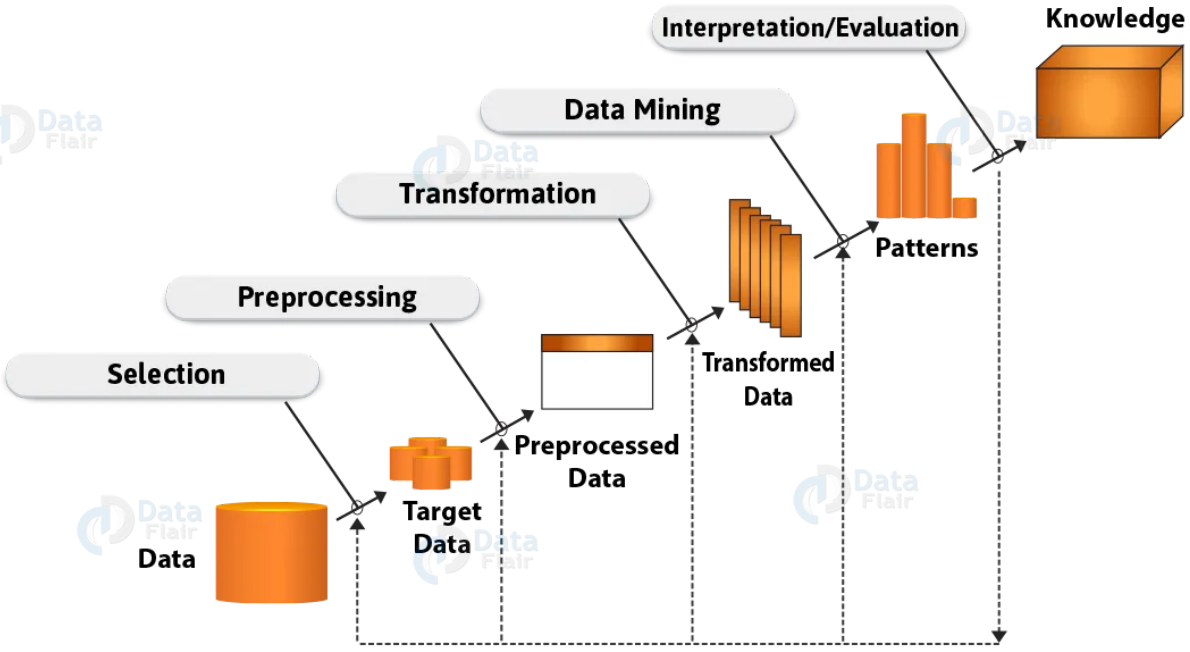
Problem Statement:

Customer segmentation: The dataset contains customer demographic and transaction data, where the goal is to segment customers based on their behavior and preferences.

Mall Customer Segmentation Data Set: This dataset contains customer demographic data (age, gender, income) and their spending scores (on a scale of 1-100) at a mall.

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

Theory:



Here are the steps we will follow to perform KDD for customer segmentation:

- Data Understanding: In this step, we'll try to understand the structure and nature of the dataset. We'll look at the different attributes and their meanings, the range and type of values for each attribute, and the relationship between the attributes. We'll also look for missing or inconsistent data.
- Data Cleaning: In this step, we'll clean the dataset by removing duplicates, correcting errors, filling in missing values, and removing outliers.
- Data Transformation: In this step, we'll transform the dataset to make it suitable for segmentation. We may need to transform categorical variables to numerical ones, normalize the data to remove scale differences, and reduce the dimensionality of the data.
- Data Reduction: In this step, we'll reduce the dimensionality of the data by applying techniques such as principal component analysis (PCA) or factor analysis. This will help us identify the most important attributes that contribute to the variability in the data.
- Data Mining: In this step, we'll apply clustering algorithms to segment the customers based on their spending behavior. We'll evaluate the results of the clustering and interpret the clusters to gain insights about the different customer segments.
- Knowledge Representation: In this step, we'll represent the knowledge gained from the clustering process in a meaningful way. This could be in the form of visualizations or summaries that highlight the key characteristics of each customer segment.

Step1: Data Understanding

- Load the dataset into a data analysis tool such as Python's Pandas library.
- Use the "head" function to display the first few rows of the dataset and get an idea of the different attributes.
- Use the "info" function to display information about the dataset, such as the number of rows, the data types of each attribute, and whether there are any missing values.
- Use the "describe" function to get summary statistics for each attribute, such as the mean, standard deviation, and quartiles.
- Use visualization tools such as histograms, scatterplots, and boxplots to explore the distribution of the different attributes and identify any outliers.

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset into a Pandas dataframe
df = pd.read_csv('/content/drive/MyDrive/Mall_Customers.csv')
```

```
# Display the first few rows of the dataset
print(df.head())
```

```
   CustomerID  Gender  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
```

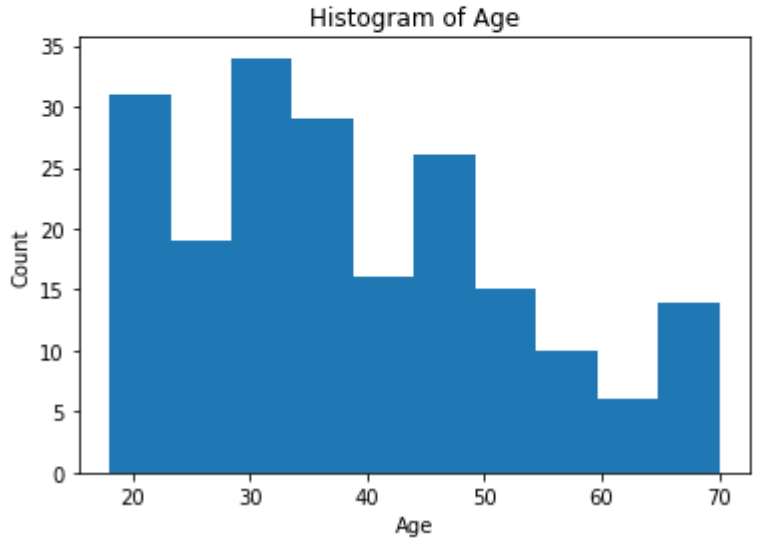
```
# Display information about the dataset
print(df.info())
```

```
<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   Gender                  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
None
```

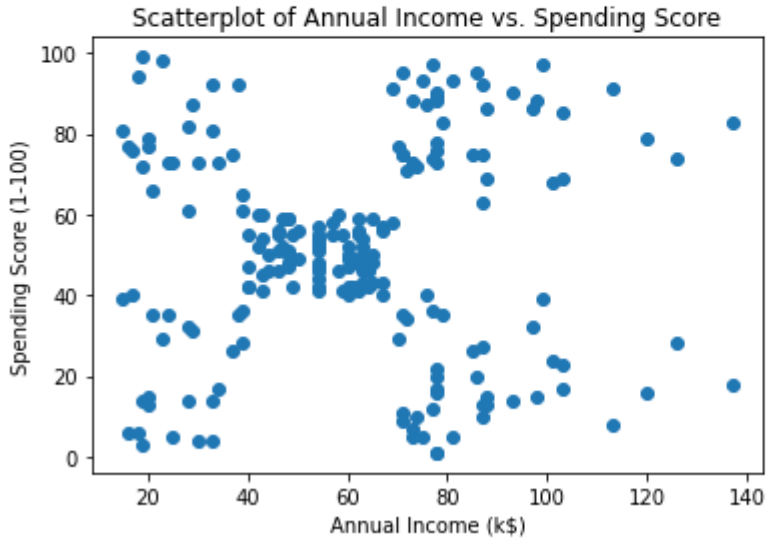
```
# Get summary statistics for each attribute
print(df.describe())
```

```
       CustomerID         Age  Annual Income (k$)  Spending Score (1-100)
count  200.000000  200.000000          200.000000              200.000000
mean   100.500000   38.850000           60.560000               50.200000
std     57.879185   13.969007           26.264721               25.823522
min      1.000000   18.000000           15.000000                1.000000
25%     50.750000   28.750000           41.500000               34.750000
50%    100.500000   36.000000           61.500000               50.000000
75%    150.250000   49.000000           78.000000               73.000000
max    200.000000   70.000000          137.000000               99.000000
```

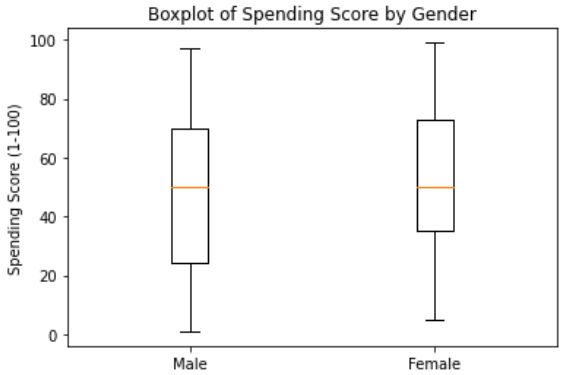
```
# Create a histogram of the Age attribute
plt.hist(df['Age'])
plt.title('Histogram of Age')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



```
# Create a scatterplot of Annual Income vs. Spending Score
plt.scatter(df['Annual Income (k$)'], df['Spending Score (1-100)'])
plt.title('Scatterplot of Annual Income vs. Spending Score')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.show()
```




```
# Create a boxplot of the Spending Score by Gender
plt.boxplot([df[df['Gender']=='Male']['Spending Score (1-100)'], df[df['Gender']=='Female']['Spending Score (1-100)']])
plt.title('Boxplot of Spending Score by Gender')
plt.xticks([1, 2], ['Male', 'Female'])
plt.ylabel('Spending Score (1-100)')
plt.show()
```



Step 2: Data Cleaning

In this step, we will:

- Remove any duplicate rows in the dataset using the "drop_duplicates" function.
- Check for missing values in the dataset using the "isnull" function and either remove rows with missing values using the "dropna" function or fill in missing values using the "fillna" function.

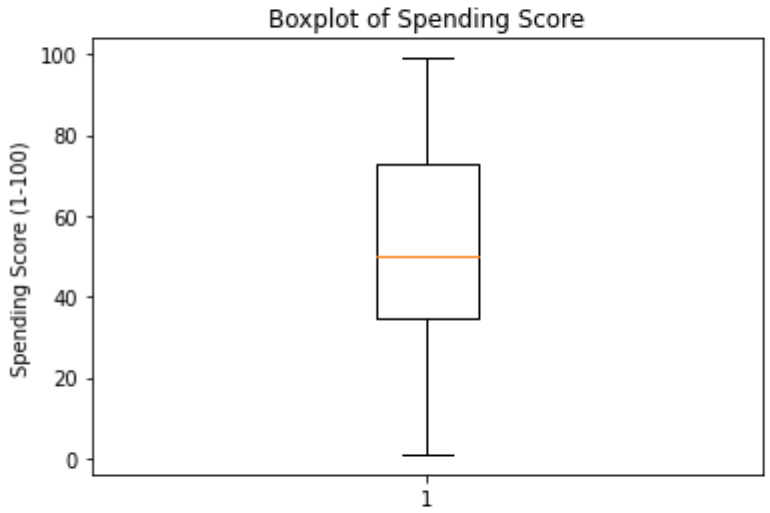
- Check for any outliers in the dataset using visualization tools such as boxplots or scatterplots. Remove any outliers using the "drop" function.

```
# Remove duplicate rows in the dataset
df = df.drop_duplicates()

# Check for missing values in the dataset
print(df.isnull().sum())
```

```
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

```
# Fill in missing values in the Income attribute with the mean income
mean_income = df['Annual Income (k$)'].mean()
df['Annual Income (k$)'] = df['Annual Income (k$)'].fillna(mean_income)
```

```
# Check for outliers in the Spending Score attribute using a boxplot
plt.boxplot(df['Spending Score (1-100)'])
plt.title('Boxplot of Spending Score')
plt.ylabel('Spending Score (1-100)')
plt.show()
```




```
# Remove outliers in the Spending Score attribute with a value above 80
df = df[df['Spending Score (1-100)'] <= 80]
```

Step 3: Data Transformation

In this step we will,

- Normalize or standardize the dataset to ensure that each attribute has a similar scale. We can use the "MinMaxScaler" or "StandardScaler" functions from the scikit-learn library to do this.
- Convert categorical variables such as Gender to numerical values using one-hot encoding or label encoding.
- Drop any unnecessary attributes that are not relevant to the customer segmentation task.
- Create new attributes that may be useful for customer segmentation, such as the ratio of Spending Score to Annual Income.

```
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import LabelEncoder
import numpy as np

# Normalize the dataset using MinMaxScaler
scaler = MinMaxScaler()
df_normalized = pd.DataFrame(scaler.fit_transform(df[['Age', 'Annual Income (k$)', 'Spending Score (1-100)']]), columns=['Age', 'Annual Income (k$)', 'Spending
```

```
# Convert the Gender attribute to numerical values using LabelEncoder
le = LabelEncoder()
df_normalized['Gender'] = le.fit_transform(df['Gender'])
```

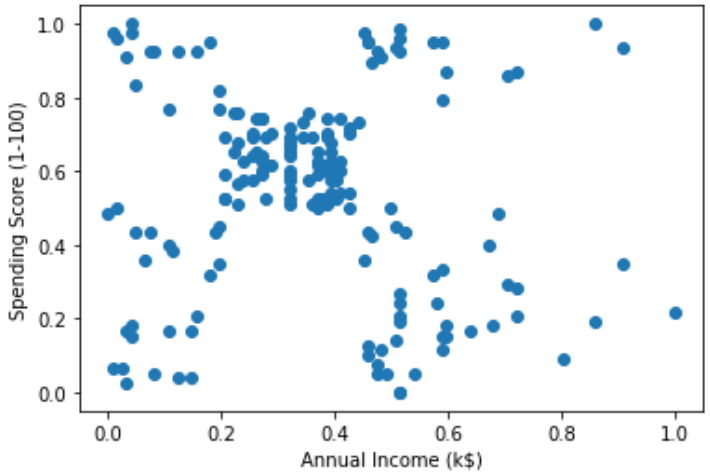
```
# Create a new attribute for the ratio of Spending Score to Annual Income
df_normalized['Score/Income'] = np.round(df_normalized['Spending Score (1-100)'] / (df_normalized['Annual Income (k$)'] + 1), 2)
```

Step 4: Data Mining

For this step, we will:

- Perform exploratory data analysis (EDA) to understand the relationships between attributes and identify patterns in the data.
- Choose a suitable clustering algorithm such as K-Means, Hierarchical Clustering, or DBSCAN to group customers into different segments based on their attributes.
- Evaluate the performance of the clustering algorithm using metrics such as Silhouette score or Calinski-Harabasz index.
- Visualize the clusters to gain insights and interpret the results.

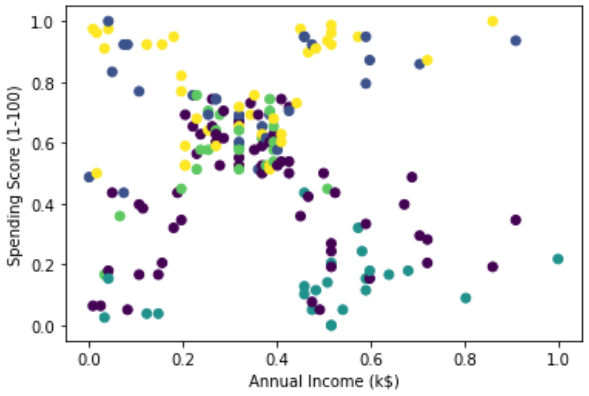
```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import matplotlib.pyplot as plt

# Perform exploratory data analysis
plt.scatter(df_normalized['Annual Income (k$)'], df_normalized['Spending Score (1-100)'])
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.show()
```




```
# Choose a suitable clustering algorithm
kmeans = KMeans(n_clusters=5, random_state=0, n_init=10)
kmeans.fit(df_normalized)

# Evaluate the performance of the clustering algorithm
score = silhouette_score(df_normalized, kmeans.labels_)
print('Silhouette score:', score)
```

```
Silhouette score: 0.3624654986471616
```

```
# Visualize the clusters
plt.scatter(df_normalized['Annual Income (k$)'], df_normalized['Spending Score (1-100)'], c=kmeans.labels_)
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.show()
```



Step 5:Data Interpretation

In this step we perform the following steps.

- Examine the cluster assignments to understand the characteristics of each cluster.
- Analyze the distribution of customers within each cluster in terms of the attributes used in clustering.
- Identify common patterns among customers in each cluster and use this information to define marketing strategies.
- Evaluate the effectiveness of the clustering solution and consider refining the model if necessary.

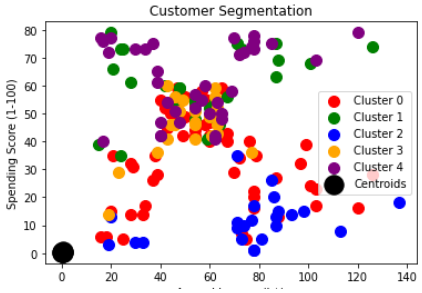
```
[ ]  # Add the cluster labels to the original dataframe
     df['Cluster'] = kmeans.labels_

     # Examine the cluster assignments
     print(df.groupby('Cluster').mean())
```

```
             CustomerID        Age  Annual Income (k$)  Spending Score (1-100)
Cluster
0             96.796610  46.677966           59.542373               34.881356
1             91.000000  27.500000           56.428571               60.250000
2            136.130435  41.565217           75.130435               11.695652
3             78.608696  58.347826           51.739130               45.652174
4             87.945946  27.837838           54.837838               61.702703
```

```
# Analyze the distribution of customers within each cluster
cluster_counts = df.groupby('Cluster').size()
print(cluster_counts)
```

```
Cluster
0    59
1    28
2    23
3    23
4    37
dtype: int64
```

```
# Visualize the clusters with different colors
colors = ['red', 'green', 'blue', 'orange', 'purple']
for i in range(kmeans.n_clusters):
    plt.scatter(df['Annual Income (k$)'][df['Cluster'] == i], df['Spending Score (1-100)'][df['Cluster'] == i], s=100, c=colors[i], label='Cluster '+str(i))
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s=300, c='black', label='Centroids')
plt.title('Customer Segmentation')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```



The clusters are defined on the basis of Spending score and Age of customers.

The various clusters formed are:

1. High-spending customers
2. Budget-conscious customers
3. Young customers
4. Senior citizen customers
5. Middle-aged customers

Based on the evaluation of the customer segmentation, we can define marketing strategies to target each segment effectively. Here are some potential strategies for each segment:

- High-Spending Customers: These customers are the most valuable to the mall, and so it is important to prioritize their needs. One strategy could be to offer personalized shopping experiences, such as exclusive access to sales or personalized recommendations. Another strategy could be to offer loyalty programs that reward these customers for their continued patronage.

- Budget-Conscious Customers: These customers are looking for value and may be more price-sensitive than other segments. One strategy could be to offer discounts or promotions that are targeted specifically to this segment. Another strategy could be to focus on providing high-quality, affordable products that meet their needs.
- Young Customers: These customers may be more interested in trendy or fashionable products, and may be more likely to use social media and other digital channels to engage with brands. One strategy could be to leverage social media to showcase new products and trends, and to encourage customers to share their experiences with the brand online.
- Seniors: These customers may be more interested in products that support their health and wellness, such as fitness equipment or healthy foods. One strategy could be to offer discounts or promotions on these types of products, and to provide educational resources that help seniors make informed decisions about their health and wellness.
- Middle-Aged Customers: These customers may be interested in products that support their professional or personal lives, such as business attire or home appliances. One strategy could be to offer a range of products that meet their diverse needs, and to provide customer service that is tailored to their specific needs and preferences.

Conclusion: We were able to identify a real-world problem like customer segmentation and perform the various KDD steps on it. We then evaluated the dataset and performed clustering of customers to generate various strategies that can be implemented for better sales at the mall.

Lab Outcome: Implement various data mining algorithms from scratch using languages like Python / Java/ R, etc.Apply BI to solve practical problems: Analyze the problem domain, use the data collected in enterprise, apply the appropriate data mining technique, interpret and visualize the results and provide decision support.

Submitted Details -

Name of Student: Roshni Bhanushali

Roll No.: 06

Date of Performance: 06/04/2023

Date of Submission: 13/04/2023