EX NO : 1 DATE :	**DATA PREPROCESSING AND EXPLORATION**

AIM :

To clean dataset and explore raw data for insights using the data preprocessing and exploration.

ALGORITHM:

1. Load the Tips dataset.
2. Handle missing values and duplicates.
3. Convert data types and encode categoricals.
4. Normalize or scale numerical data.
5. Visualize and summarize statistics.

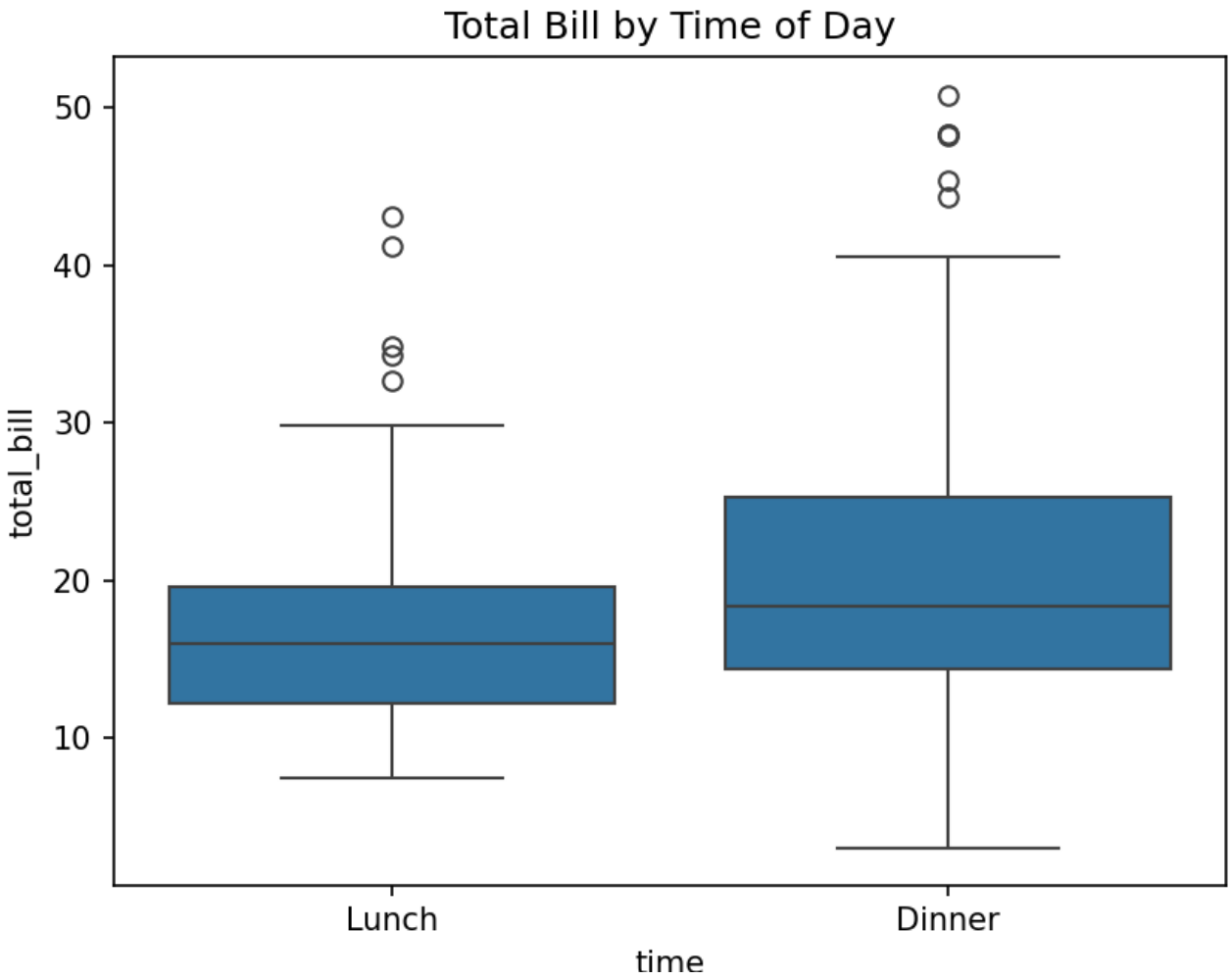
PROGRAM:

```
import pandas as pd

import seaborn as data

import matplotlib.pyplot as plt

df = data.load_dataset("tips")

print("First 5 rows:")

print(df.head())

print(df.describe())


print("\nMissing values:")

print(df.isnull().sum())


df.drop_duplicates(inplace=True)

print("\nCount of records per day:")

print(df['day'].value_counts())
```

```
data.boxplot(x='time', y='total_bill', data=df)

plt.title("Total Bill by Time of Day")

plt.show()
```

OUTPUT

```
   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
2       21.01  3.50    Male     No  Sun  Dinner     3
3       23.68  3.31    Male     No  Sun  Dinner     2
4       24.59  3.61  Female     No  Sun  Dinner     4
       total_bill         tip        size
count  244.000000  244.000000  244.000000
mean    19.785943    2.998279    2.569672
std      8.902412    1.383638    0.951100
min      3.070000    1.000000    1.000000
25%     13.347500    2.000000    2.000000
50%     17.795000    2.900000    2.000000
75%     24.127500    3.562500    3.000000
max     50.810000   10.000000    6.000000

Missing values:
total_bill    0
tip           0
sex           0
smoker        0
day           0
time          0
size          0
dtype: int64

Count of records per day:
day
Sat     87
Sun     76
Thur    61
Fri     19
Name: count, dtype: int64
```



RESULT

Hence the dataset is pre processed and filled the missing values and Exploration done by visualization.

EX NO : 2 DATE :	**CONDUCTING CLUSTER ANALYSIS USING K-MEANS CLUSTERING**

AIM :

To make a Cluster Analysis we group similar data points into clusters using K-Means.

ALGORITHM:

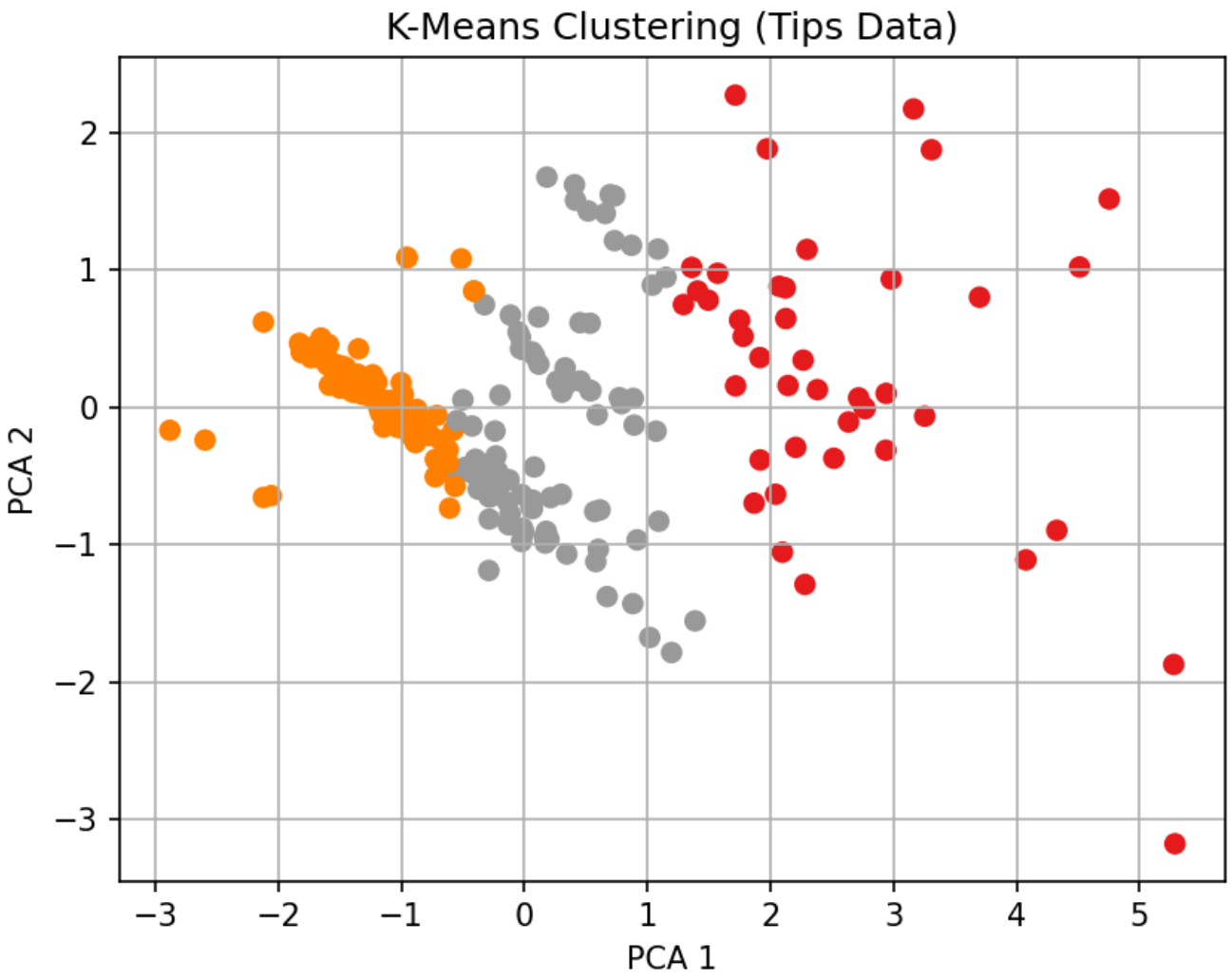
1. Load the required features (total_bill, tip, size) from the dataset.
2. Normalize the data using Standard Scaler
3. Fit the K-Means clustering algorithm
4. Use PCA to reduce data to 2 dimensions for visualization.
5. Plot the clusters using a scatter plot

PROGRAM:

```
import seaborn as data

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA


df = data.load_dataset("tips")[['total_bill', 'tip', 'size']]


model = StandardScaler()

result = model.fit_transform(df)


kmeans = KMeans(n_clusters=3)

df['Cluster'] = kmeans.fit_predict(result)


pca = PCA(n_components=2)

pca_data= pca.fit_transform(result)
```

```
plt.scatter(pca_data[:, 0], pca_data[:, 1], c=df['Cluster'], cmap='Set1')
plt.title("K-Means Clustering (Tips Data)")
plt.xlabel("PCA 1")
plt.ylabel("PCA 2")
plt.grid(True)
plt.show()
```

OUTPUT



RESULT

Hence the data points are grouped into clusters and cluster center and group characteristics are visualized

EX NO : 3 DATE :	**CONFIGURING DECISION TREE**

AIM :

To predict whether a student will *Pass* or *Fail* using a Decision Tree based on their marks and attendance.

ALGORITHM:

1. Create a dataset with Marks, Attendance, and Result.
2. Split the dataset into input features (Marks, Attendance) and target label (Result).
3. Train a Decision Tree Classifier using the input and output data.
4. Get marks and attendance from the user as input.
5. Predict and display whether the student will *Pass* or *Fail*.

PROGRAM :

```
from sklearn.tree import DecisionTreeClassifier

import pandas as pd

data = {

   'Marks': [90, 70, 50, 30, 85, 40, 60, 25],

   'Attendance': [90, 80, 60, 30, 95, 50, 75, 20],

   'Result': ['Pass', 'Pass', 'Pass', 'Fail', 'Pass', 'Fail', 'Pass', 'Fail']

}

df = pd.DataFrame(data)

X = df[['Marks', 'Attendance']]

y = df['Result']

model = DecisionTreeClassifier()

model.fit(X, y)

mark = int(input("Enter the student's mark: "))

attendance = int(input("Enter the student's attendance: "))

user_input = [[mark, attendance]]
```

```
prediction = model.predict(user_input)

print("Prediction:", prediction[0])
```

OUTPUT

```
Enter the student's mark: 70
Enter the student's attendance: 95

Prediction: Pass

Enter the student's mark: 20
Enter the student's attendance: 40

Prediction: Fail
```

RESULT:

Hence we build the decision tree based on student mark and attendence mode predict the pass or fail.

EX NO : 4 DATE :	**PERFORMING REGRESSION ANALYSIS USING LINEAR REGRESSION**

AIM :

To predict a student's marks based on the number of hours studied using Linear Regression

ALGORITHM:

1. Make a dataset with study hours and marks.
2. Separate hours as input and marks as output.
3. Train a Linear Regression model using this data.
4. Ask the user to enter study hours.
5. Predict and show the marks, and draw the regression line.

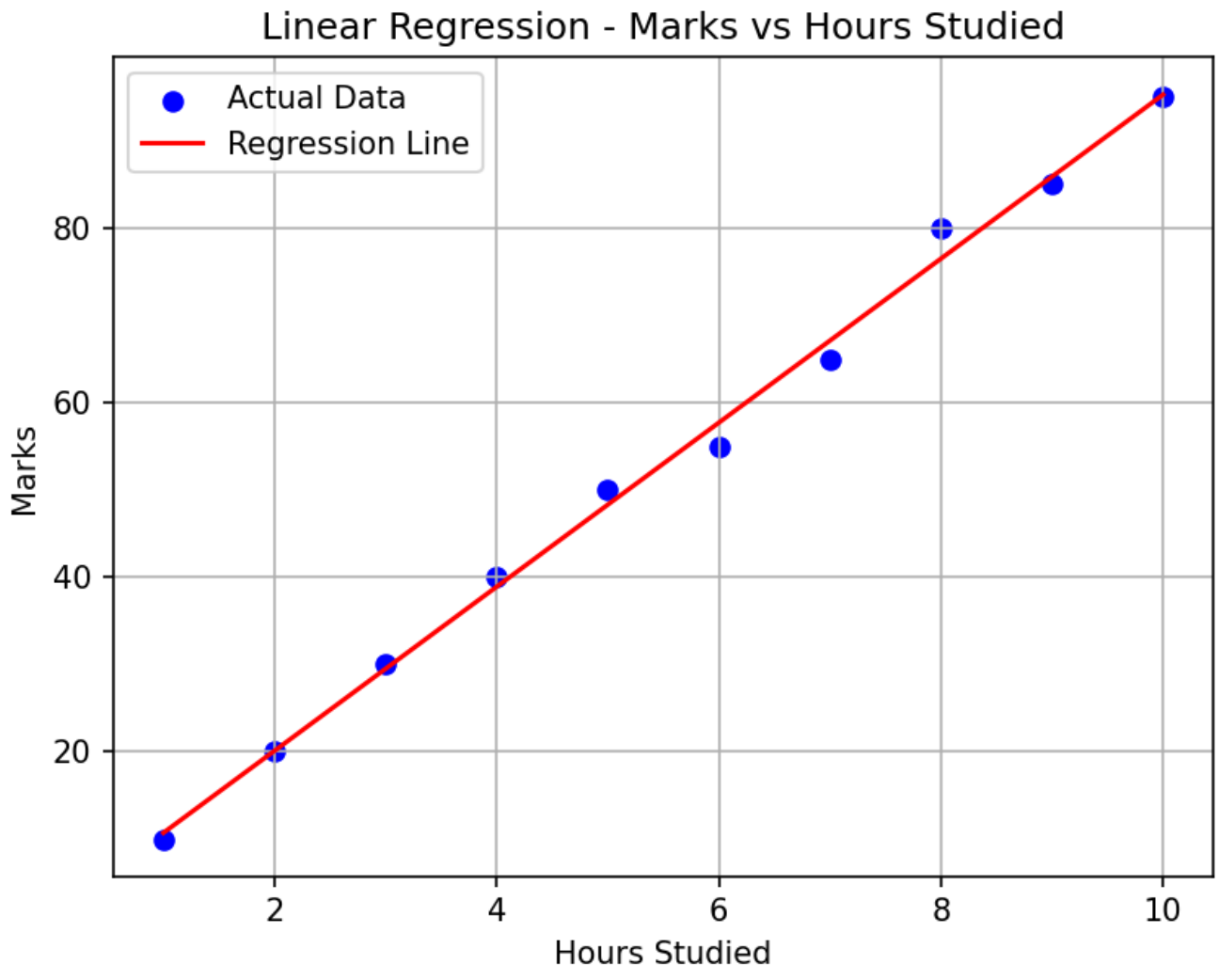
PROGRAM

```
import pandas as pd
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
data = {
   'Hours': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
   'Marks': [10, 20, 30, 40, 50, 55, 65, 80, 85, 95]
}
df = pd.DataFrame(data)
X = df[['Hours']]
y = df['Marks']

model = LinearRegression()
model.fit(X, y)

hour = float(input("Enter hours studied: "))
predicted_marks = model.predict([[hour]])
print(f"Predicted Marks: {predicted_marks[0]:.2f}")
plt.scatter(X, y, color='blue', label='Actual Data')
```

```
plt.plot(X, model.predict(X), color='red', label='Regression Line')
plt.xlabel('Hours Studied')
plt.ylabel('Marks')
plt.title('Linear Regression - Marks vs Hours Studied')
plt.legend()
plt.grid(True)
plt.show()
```

OUTPUT

```
Enter hours studied: 5
Predicted Marks: 48.30
```



RESULT

Hence, we built a Linear Regression model to predict student marks based on the number of study hours

EX NO : 5 DATE :	**CREATING AN INTERACTIVE DASHBOARD USING TABLEAU**

AIM :

To design a dashboard to visualize and interact with data using Tableau.

ALGORITHM:

1. Connect to a data source.
2. Clean and format data in Tableau.
3. Drag fields to create visualizations.
4. Add filters and interactivity.
5. Arrange visuals into dashboard layout.

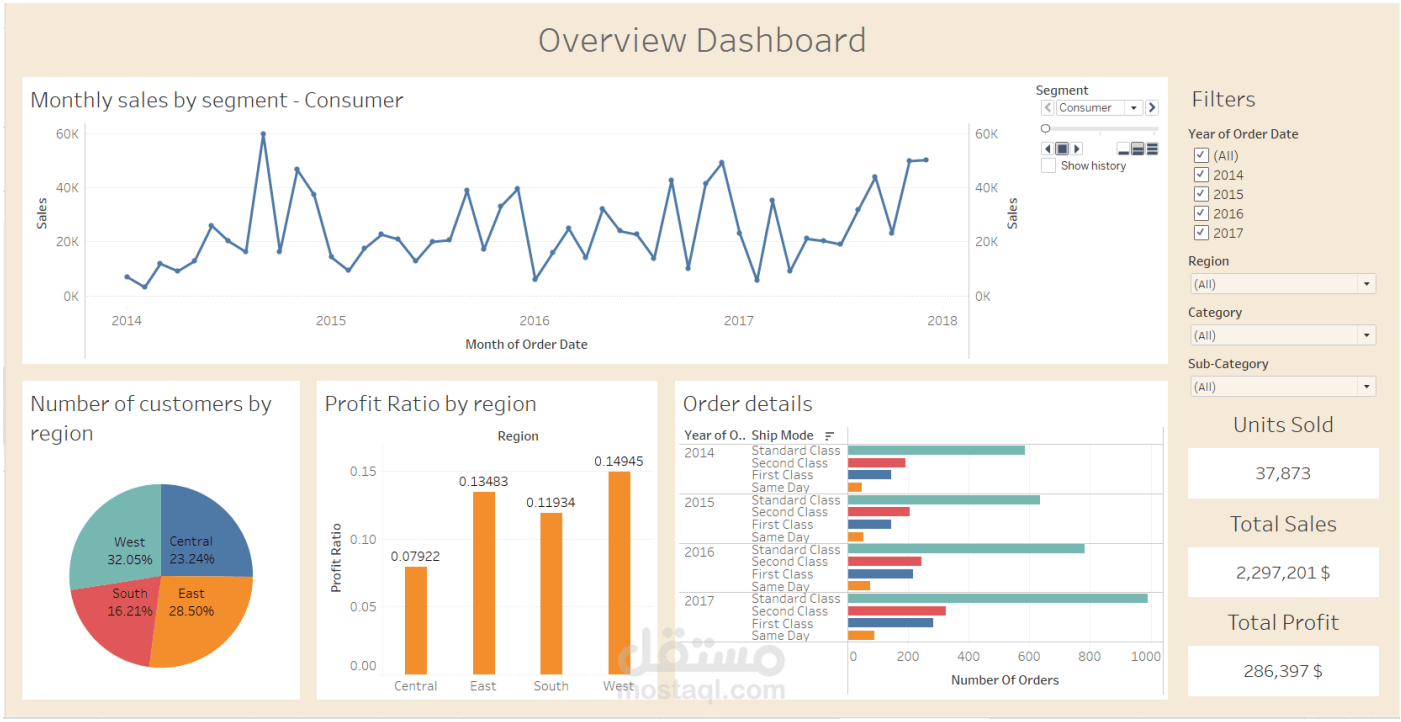
PROCEDURE

1. **Connect to Data Source**
    - Open Tableau and choose the built-in dataset **"Sample - Superstore"** from the saved samples.
2. **Clean and Format Data**
    - Review the fields (e.g., Order Date, Sales, Profit, Category).
    - Rename fields or change data types if needed.
3. **Create Visualizations**
    - Drag Category to Columns and Sales to Rows to create a bar chart.
    - Drag Region and Profit to create a map or pie chart.
4. **Add Filters and Interactivity**
    - Add filters like Region, Category, or Year to allow users to explore the data.
    - Use parameters or highlight actions for interactivity.

5. **Build Dashboard Layout**

   - Go to the **Dashboard** tab.
   - Drag created charts onto the dashboard.
   - Arrange them neatly and add filters, legends, and titles.

OUTPUT:



RESULT

An interactive Tableau dashboard is created using the “Sample - Superstore” dataset showing sales, profit, and category-wise insights with filters and user interactivity

EX NO : 6 DATE :	**DATA AGGREGATION AND STATISTICAL FUNCTIONS IN TABLEAU**

AIM :

To Summarize data using aggregation and statistical tools in Tableau.

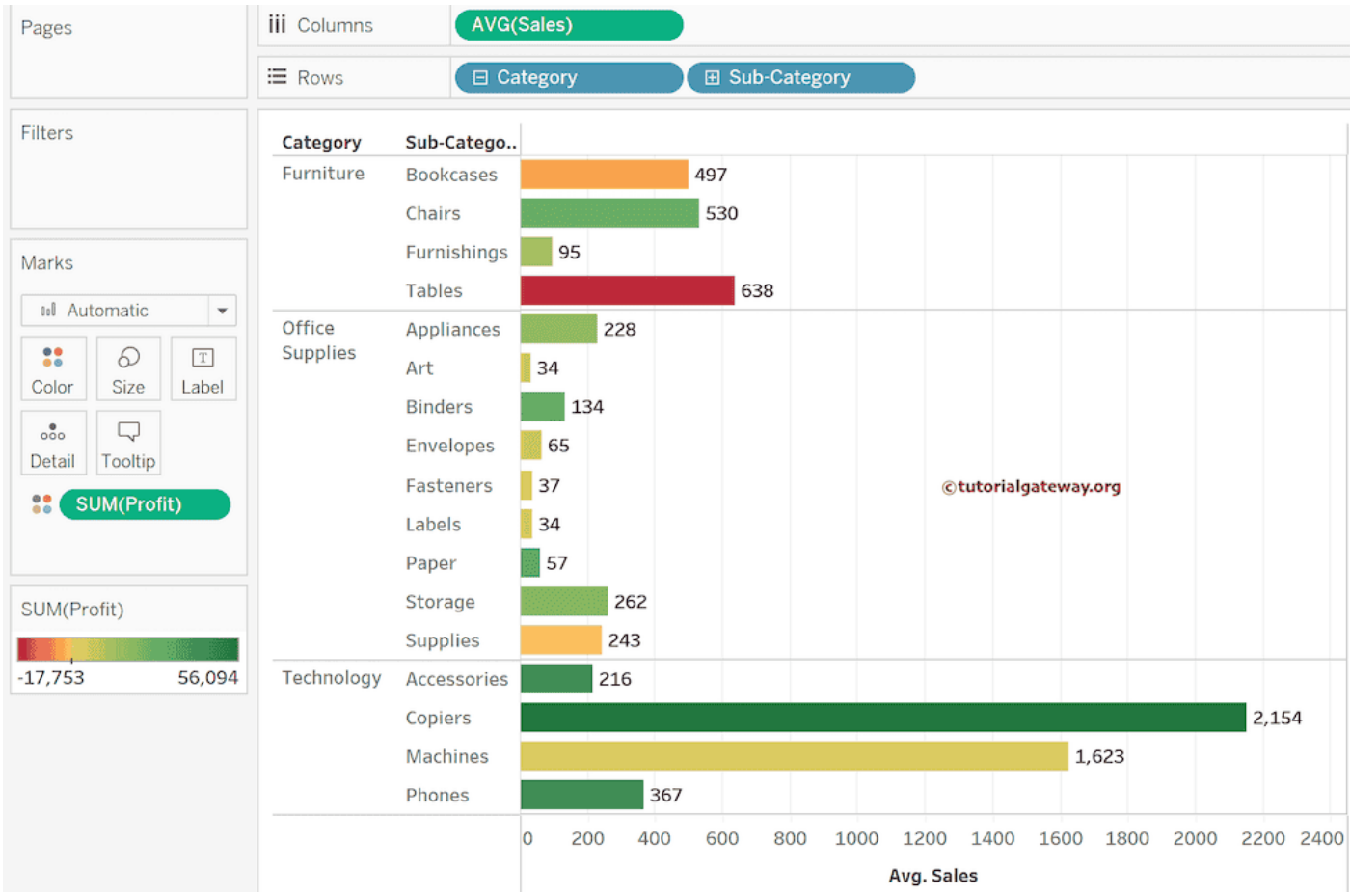
ALGORITHM:

1. Import data to Tableau.
2. Use drag-and-drop to set dimensions/measures.
3. Apply aggregations like SUM, AVG, COUNT.
4. Use statistical functions like median, stdev.
5. Display results in visual charts.

PROCEDURE

1. Connect to Data Source
    - Open Tableau and load a dataset (e.g., "Sample - Superstore").
2. Drag Measures and Dimensions
    - Drag fields like Sales, Profit, or Quantity to Rows or Columns shelf.
3. Apply Aggregation
    - Tableau automatically applies aggregations like SUM, AVG, MIN, MAX, etc.
    - You can change aggregation by right-clicking the measure and selecting an aggregation type.
4. Use Statistical Tools
    - Use built-in tools like Trend Lines, Reference Lines, or Forecasting from the Analytics pane.
5. Visualize Summary
    - Create charts (e.g., bar, line, scatter) that reflect the aggregated data.
    - Add tooltips and filters to make the analysis interactive.

OUTPUT



RESULT

Hence the data is summarized and visualized using aggregations and statistical tools, helping identify trends, totals, and patterns within the dataset.

EX NO : 7 DATE :	**MODEL EVALUATION USING ACCURACY, PRECISION, RECALL AND F1 SCORE**

AIM :

To evaluate model performance using key metrics.

ALGORITHM:

1. Train a classifier on labeled data.
2. Make predictions on test set.
3. Compute confusion matrix.
4. Calculate accuracy, precision, recall.
5. Derive F1 score for balanced measure.

PROGRAM:

```
from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

from sklearn.model_selection import train_test_split



X = [[1], [2], [3], [4], [5], [6], [7], [8]]

y = [0, 0, 0, 0, 1, 1, 1, 1]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)



model = LogisticRegression()

model.fit(X_train, y_train)



y_pred = model.predict(X_test)



accuracy = accuracy_score(y_test, y_pred)

precision = precision_score(y_test, y_pred)
```

```
recall = recall_score(y_test, y_pred)

f1 = f1_score(y_test, y_pred)


print("Accuracy:", accuracy)

print("Precision:", precision)

print("Recall:", recall)

print("F1 Score:", f1)
```

OUTPUT:

```
Accuracy: 1.0
Precision: 1.0
Recall: 1.0
F1 Score: 1.0
```

RESULT

Hence, we evaluated the classification model using accuracy, precision, recall, and F1-score to measure its performance effectively

EX NO : 8 DATE :	**CONFIGURING DATA VISUALIZATION USING POWER BI**

AIM :

To evaluate model performance using key metrics.

ALGORITHM:

1. Import data into Power BI.
2. Clean and transform data in Power Query.
3. Create visual charts.
4. Use slicers and filters for interaction.
5. Publish report to Power BI service.

PROCEDURE:

1. **Open Power BI Desktop**

   Launch Power BI on your computer.

2. **Import Data**

   Click on **"Get Data"**, choose a source Excel, CSV and load your dataset.

3. **Transform Data**

   Use **Power Query Editor** to clean, filter, rename columns, and format data as needed.

4. **Create Visuals**

   Drag fields onto the canvas to create charts, tables, maps, and KPIs.

5. **Add Filters and Slicers**

   Insert slicers or use the Filters pane to make visuals interactive.

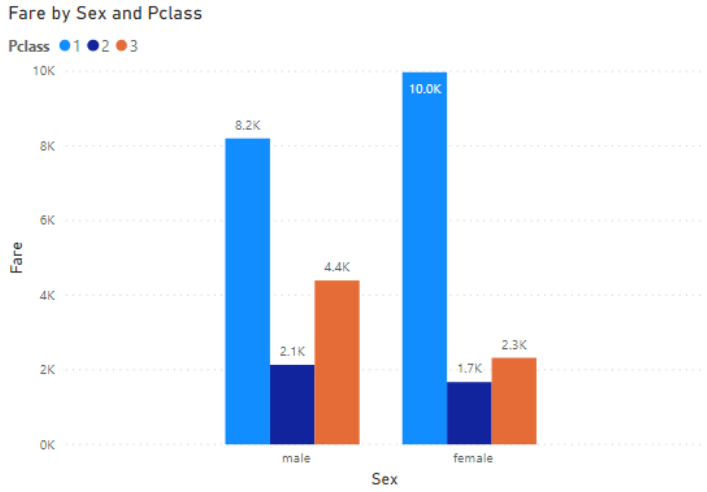
6. **Design the Report Layout**

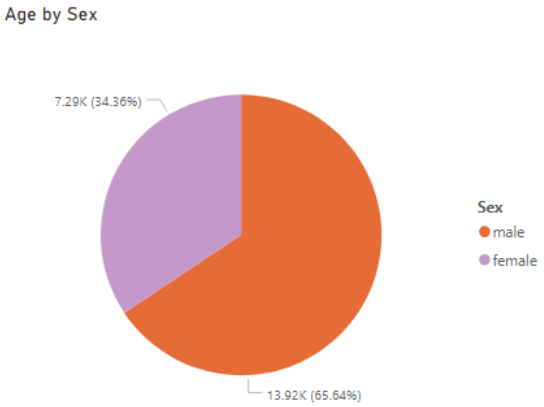
   Arrange visuals neatly, add titles, and apply themes or formatting.

7. **Publish to Power BI Service**

   Sign in and click **"Publish"** to share your dashboard with others online.

OUTPUT





RESULT

Hence, we configured an interactive Power BI dashboard by importing, transforming, and visualizing data, allowing users to explore trends and gain meaningful insights easily

EX NO : 9 DATE :	**CONDUCTING COMPARATIVE ANALYSIS OF VISUALIZATION TECHNIQUES**

AIM :

To Compare different chart types for the same dataset

ALGORITHM:

1. Choose a dataset and multiple chart types.
2. Plot charts using same metrics.
3. Evaluate readability and clarity.
4. Measure effectiveness of data communication.
5. Recommend best visualization method.

PROCEDURE:

1. **Select Dataset**

   Choose a dataset for visualization.

2. **Choose Multiple Chart Types**

   Decide on various visual types like bar chart, pie chart, line graph, heatmap, etc.

3. **Create Visualizations**

   Use a tool like Tableau, Power BI to plot the charts using the same metrics.

4. **Analyze Each Chart**

   Assess readability, clarity, visual appeal, and ease of interpretation.

5. **Compare and Conclude**

   Evaluate which chart communicates the data most effectively and recommend it for the intended audience.

OUTPUT:

RESULT:

Hence, we compared various visualization techniques for the same dataset and identified the most effective chart type based on clarity, readability, and communication strength.

EX NO : 10 DATE :	**CONNECT TO DATA, BUILD CHARTS AND ANALYZE DATA, CREATE DASHBOARD, CREATE STORIES USING TABLEAU**

AIM :

Use Tableau to explore data, build visuals, and narrate insights.

ALGORITHM:

1. Connect to and prepare the dataset.
2. Build charts for KPIs and trends.
3. Combine visuals into dashboards.
4. Add interactivity via filters.
5. Create a story to guide insights.

PROCEDURE:

a. Connect to Dataset

Open Tableau and load a dataset Superstore

b. **Prepare the Data**

Clean, rename, and format fields as needed in the Data Source or Data Pane.

c. **Build Charts**

Drag fields to rows/columns and create visuals like bar charts, line graphs, and maps for KPIs and trends.

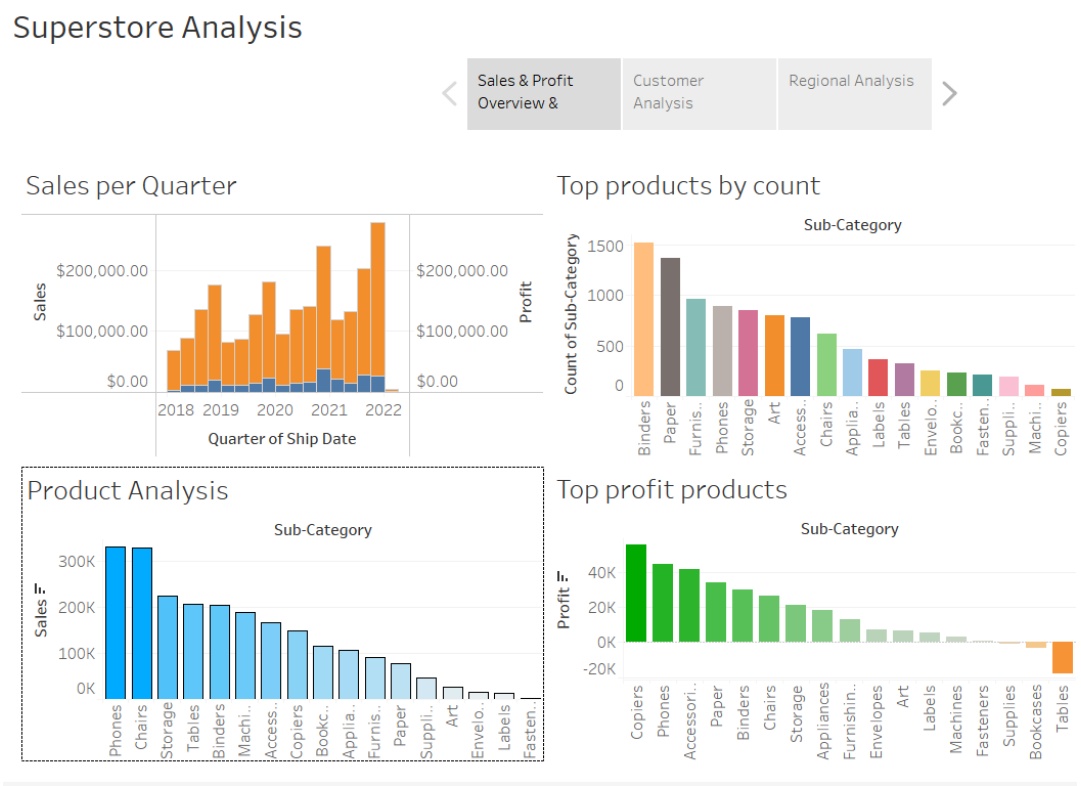
d. Create Dashboard

Go to the Dashboard tab and arrange your visuals with filters, legends, and layout settings**.**

e. **Create Story**

Click on the Story tab, add sheets/dashboards, and annotate each step to narrate the insight progression.

OUTPUT



RESULT:

Hence, we used Tableau to connect, visualize, and analyse data by building dashboards and stories, helping stakeholders explore and understand insights interactively.