1.Create a dataset in any one of the format (CSV,Excel) named bike_details with the following attributes: bike name ,model,color, prize,m_year,orderdate,shipped enter data for 15 bikes.

- To load and display the data using the following functions load(), csv.reader(), numpy.loadtxt() and numpy.genfromtxt(), pandas.read_csv().
- Use the python module to read the entire details of the bikes.
- Display bike name details which price is more than 100000.
- Display the bike model details price under 70000.
- Display alternative rows.

2. Implement a tree Map display for the top 5 bikes.

3. Visualize the Geospatial data of bikes sales across different countries.

Program :

```
import csv

import numpy as np

import pandas as pd


# Using csv.reader()

def load_csv_with_csv_reader(filename):

    with open(filename, 'r') as file:

        reader = csv.reader(file)

        for row in reader:

            print(row)


# Using numpy.loadtxt()

def load_csv_with_np_loadtxt(filename):

    data = np.loadtxt(filename, delimiter=',', dtype=str)

    print(data)
```

```
# Using numpy.genfromtxt()
def load_csv_with_np_genfromtxt(filename):
    data = np.genfromtxt(filename, delimiter=',', dtype=str, skip_header=1)
    print(data)

# Using pandas.read_csv()
def load_csv_with_pandas(filename):
    data = pd.read_csv(filename)
    print(data)

filename = 'bike_details.csv'

print("Using csv.reader():")
load_csv_with_csv_reader(filename)

print("\nUsing numpy.loadtxt():")
load_csv_with_np_loadtxt(filename)

print("\nUsing numpy.genfromtxt():")
load_csv_with_np_genfromtxt(filename)

print("\nUsing pandas.read_csv():")
load_csv_with_pandas(filename)
```

2. Program

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

# Load data using pandas
filename = 'bike_details.csv'
data = pd.read_csv(filename)

# Get the top 5 bikes based on price
top_bikes = data.nlargest(5, 'prize')

# Plotting the TreeMap
plt.figure(figsize=(10, 6))
squarify.plot(sizes=top_bikes['prize'], label=top_bikes['bike name'], alpha=0.7)
plt.axis('off')
plt.title('Top 5 Bikes by Price')
plt.show()
```

3. Program

```
import geopandas as gpd

import matplotlib.pyplot as plt

import pandas as pd


# Load the geospatial data of countries

world = gpd.read_file('path_to_shapefile_or_geojson')


# Load bike sales data using pandas

filename = 'bike_details.csv'

data = pd.read_csv(filename)


# Perform any necessary data preprocessing, aggregations, or joins

# For example, let's count bike sales per country for demonstration

country_sales = data['orderdate'].groupby(data['country_column']).count().reset_index()


# Merge bike sales data with the world geospatial data

merged_data = world.merge(country_sales, left_on='country_column', right_on='country_column')


# Create a choropleth map

fig, ax = plt.subplots(1, 1, figsize=(10, 6))

merged_data.plot(column='sales_column', ax=ax, legend=True, cmap='OrRd', legend_kwds={'label':
"Bike Sales"})

ax.set_title("Bike Sales by Country")
```

```
ax.set_axis_off()

plt.show()
```

2. Create a data set in any one of the format (CSV,Excel) named mobile details with the following attributes Mobile model,manufacturer,color,price,order date,shipped enter details for 20 mobiles.

- To load and display the data using the following function load(),csvreader(),numpy.loadtxt(),numpy.genfromtxt(),pandas.read_csv()
- By using python module to display the mobile details
- Display the mobile model details whose price is more than 25000
- Display the mobile model details whose price is in between the range of 10000 to 20000.

b. Design a Stacked Bar Chart, Grouped Bar Chart using Bivariate data of categorical vs continuous for the above data set.

c.Perform Trendline analysis for mobile dataset.

Program :

```
import csv

import numpy as np

import pandas as pd


# Using csv.reader()

def load_csv_with_csv_reader(filename):

    with open(filename, 'r') as file:

        reader = csv.reader(file)

        for row in reader:

            print(row)
```

```
# Using numpy.loadtxt()
def load_csv_with_np_loadtxt(filename):
    data = np.loadtxt(filename, delimiter=',', dtype=str)
    print(data)

# Using numpy.genfromtxt()
def load_csv_with_np_genfromtxt(filename):
    data = np.genfromtxt(filename, delimiter=',', dtype=str, skip_header=1)
    print(data)

# Using pandas.read_csv()
def load_csv_with_pandas(filename):
    data = pd.read_csv(filename)
    print(data)

filename = 'mobiledetails.csv'

print("Using csv.reader():")
load_csv_with_csv_reader(filename)

print("\nUsing numpy.loadtxt():")
load_csv_with_np_loadtxt(filename)

print("\nUsing numpy.genfromtxt():")
load_csv_with_np_genfromtxt(filename)
```

```
print("\nUsing pandas.read_csv():")
load_csv_with_pandas(filename)

# Load data using pandas
data = pd.read_csv(filename)

# Display mobile model details with price more than 25000
high_price_mobiles = data[data['price'] > 25000]
print("Mobiles with price more than 25000:")
print(high_price_mobiles)

# Display mobile model details with price in the range of 10000 to 20000
mid_range_mobiles = data[(data['price'] >= 10000) & (data['price'] <= 20000)]
print("\nMobiles with price in the range of 10000 to 20000:")
print(mid_range_mobiles)
```

2.

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

# Load data using pandas
filename = 'sales_data.csv'
data = pd.read_csv(filename)

# Create a stacked bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x='Category', y='Sales', data=data, estimator=sum, ci=None)
plt.title('Stacked Bar Chart of Sales by Category')
plt.xlabel('Category')
plt.ylabel('Total Sales')
plt.xticks(rotation=45)
plt.show()

# Create a grouped bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x='Category', y='Sales', data=data, ci=None)
plt.title('Grouped Bar Chart of Sales by Category')
plt.xlabel('Category')
plt.ylabel('Average Sales')
plt.xticks(rotation=45)
plt.show()
```

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import linregress

# Load the mobile dataset using pandas
filename = 'mobiledetails.csv'
data = pd.read_csv(filename)

# Assuming your dataset includes a 'price' column and an 'order date' column
# Convert the 'order date' column to datetime format
data['order date'] = pd.to_datetime(data['order date'])

# Sort the data by order date
data.sort_values(by='order date', inplace=True)

# Create a scatter plot of mobile prices over time
plt.figure(figsize=(10, 6))
plt.scatter(data['order date'], data['price'], label='Mobile Prices')

# Fit a linear regression line to the data
slope, intercept, r_value, p_value, std_err = linregress(data['order date'].view('int64'), data['price'])
trendline = intercept + slope * data['order date'].view('int64')
plt.plot(data['order date'], trendline, color='red', label='Trendline')

plt.title('Trendline Analysis of Mobile Prices')
plt.xlabel('Order Date')
```

```
plt.ylabel('Price')

plt.legend()

plt.xticks(rotation=45)

plt.tight_layout()


plt.show()
```

3.

a.create a employee dataset containing Employee ID,Name,Join year,Average ratings,Department

- Who are the employees that have joined the company after 2018 and have an average performance rating of at least 4.0?
- Identify employees who are in the "Sales" department and have been with the company for more than 2 years.
- Display the names of employees who are in the "HR" department and have an average rating less than 4.0.

b.visualize and perform Multivariate analysis using Multiple variables involving Multiple measures Scatterplot Matrix, Parallel Coordinates

C.analyze the geographical distribution of employees' home addresses to identify clusters of employees living in specific regions.

Program

```
import pandas as pd

from datetime import datetime


# Creating the Employee dataset

data = {

   'Employee ID': [101, 102, 103, 104, 105, 106, 107, 108, 109],

   'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve', 'Frank', 'Grace', 'Helen', 'Isaac'],

   'Join year': [2017, 2019, 2020, 2022, 2018, 2021, 2019, 2020, 2018],

   'Average ratings': [4.5, 4.0, 3.8, 4.2, 4.6, 3.9, 4.1, 3.7, 4.4],

   'Department': ['HR', 'Sales', 'IT', 'IT', 'Sales', 'HR', 'Sales', 'IT', 'HR']

}


df = pd.DataFrame(data)
```

```
# Employees joined after 2018 with average rating >= 4.0
condition1 = (df['Join year'] > 2018) & (df['Average ratings'] >= 4.0)
result1 = df[condition1]
print("Employees joined after 2018 with average rating >= 4.0:")
print(result1)

# Employees in "Sales" department with more than 2 years of experience
current_year = datetime.now().year
df['Years with company'] = current_year - df['Join year']
condition2 = (df['Department'] == 'Sales') & (df['Years with company'] > 2)
result2 = df[condition2]
print("\nEmployees in 'Sales' department with more than 2 years of experience:")
print(result2)

# Employees in "HR" department with average rating < 4.0
condition3 = (df['Department'] == 'HR') & (df['Average ratings'] < 4.0)
result3 = df[condition3]
print("\nEmployees in 'HR' department with average rating < 4.0:")
print(result3)
```

2.

```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Scatterplot Matrix
sns.pairplot(df, hue='Department')
plt.title('Scatterplot Matrix')
plt.show()

# Parallel Coordinates
sns.set(style="whitegrid")
plt.figure(figsize=(10, 6))
sns.parallel_coordinates(df, 'Department', color=['blue', 'green', 'red'])
plt.title('Parallel Coordinates')
plt.show()
```

3.

```
import pandas as pd
import numpy as np

# Generate sample data
np.random.seed(42)
n_employees = 50

data = {
    'Employee ID': range(1, n_employees + 1),
    'Name': [f'Employee_{i}' for i in range(1, n_employees + 1)],
    'Home Address': [f'City_{np.random.randint(1, 11)}, Country_{np.random.randint(1, 6)}' for _ in range(n_employees)]
}
```

```
df = pd.DataFrame(data)


from geopy.geocoders import Nominatim


geolocator = Nominatim(user_agent="employee_geocoder")


def geocode_address(address):
    location = geolocator.geocode(address)
    if location:
        return location.latitude, location.longitude
    else:
        return None, None


# Geocode addresses
df['Latitude'], df['Longitude'] = zip(*df['Home Address'].apply(geocode_address))
import folium


# Create a map centered at the mean of all latitude and longitude values
latitude_mean = df['Latitude'].mean()
longitude_mean = df['Longitude'].mean()
m = folium.Map(location=[latitude_mean, longitude_mean], zoom_start=4)


# Add markers for each employee
for idx, row in df.iterrows():
    folium.Marker(
        location=[row['Latitude'], row['Longitude']],
        popup=row['Name'],
```

```
        icon=folium.Icon(color='blue', icon='user')
    ).add_to(m)

# Save the map to an HTML file
m.save('employee_geographical_distribution.html')
```

4. a.Create a student data containing the following attributes Id,Name,Gender,Age,GPA Enter 15 details

Import the necessary modules,and data set.

Design a Bar chart, using Univariate analysis of Categorical Data(sex) and design a Scatterplot, analysis of continuous data given in above data set.

b. Construct a tree Map display for top 5 GPA students

c.visualize Trend Line analysis for student dataset.

Program:

a.

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

```
# Create the student dataset

data = {

    'Id': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15],

    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Eve', 'Frank', 'Grace', 'Helen', 'Isaac', 'Jane', 'Kathy', 'Leo', 'Mia', 'Nick', 'Olivia'],

    'Gender': ['Female', 'Male', 'Male', 'Male', 'Female', 'Male', 'Female', 'Female', 'Male', 'Female', 'Female', 'Male', 'Female', 'Male', 'Female'],

    'Age': [20, 21, 19, 22, 20, 23, 22, 21, 24, 20, 19, 22, 21, 23, 22],

    'GPA': [3.8, 3.9, 4.0, 3.6, 3.7, 4.2, 3.5, 3.9, 4.3, 3.6, 3.8, 4.1, 4.0, 3.7, 4.2]

}


df = pd.DataFrame(data)


# Univariate analysis - Bar chart of Gender

plt.figure(figsize=(6, 4))

sns.countplot(x='Gender', data=df)

plt.title('Bar Chart of Gender')

plt.show()


# Scatterplot analysis - Age vs GPA

plt.figure(figsize=(8, 6))

sns.scatterplot(x='Age', y='GPA', data=df)

plt.title('Scatterplot of Age vs GPA')

plt.xlabel('Age')

plt.ylabel('GPA')

plt.show()
```

b.

```
import squarify

# Get top 5 GPA students
top_gpa_students = df.nlargest(5, 'GPA')

# Create Tree Map display
plt.figure(figsize=(8, 6))
sizes = top_gpa_students['GPA']
labels = top_gpa_students['Name']
colors = sns.color_palette("viridis", len(sizes))
squarify.plot(sizes=sizes, label=labels, color=colors, alpha=0.7)
plt.axis('off')
plt.title('Top 5 GPA Students - Tree Map')
plt.show()
```

c.

```
# Create a scatter plot with trend line (regression line)
plt.figure(figsize=(8, 6))
sns.regplot(x='Age', y='GPA', data=df)
plt.title('Trend Line Analysis of Age vs GPA')
plt.xlabel('Age')
plt.ylabel('GPA')
plt.show()
```

5.

a.Create Flight Booking Dataset and analyze the flight connections and visualize the network of cities connected by flights.

B.Construct the Geospatial Data of the flight over different countries.

a.

```
import pandas as pd
import numpy as np

# Generate sample flight booking data
n_flights = 50

data = {
    'FlightID': range(1, n_flights + 1),
    'Source': [f'City_{np.random.randint(1, 11)}' for _ in
range(n_flights)],
    'Destination': [f'City_{np.random.randint(1, 11)}' for _ in
range(n_flights)]
}

df_flights = pd.DataFrame(data)
```

```
# Analyze flight connections
flight_connections = df_flights.groupby(['Source',
'Destination']).size().reset_index(name='Count')
print("Flight Connections:")
print(flight_connections)
```

```
import networkx as nx
import matplotlib.pyplot as plt

# Create a graph
G = nx.Graph()

# Add nodes (cities)
all_cities =
set(df_flights['Source']).union(set(df_flights['Destination']))
G.add_nodes_from(all_cities)
```

```
# Add edges (connections)
for _, row in df_flights.iterrows():
    G.add_edge(row['Source'], row['Destination'])

# Draw the network
plt.figure(figsize=(10, 8))
pos = nx.spring_layout(G, seed=42)  # Layout for arranging nodes
nx.draw(G, pos, with_labels=True, node_size=500, node_color='skyblue',
font_size=10, font_color='black', font_weight='bold')
plt.title('Network of Cities Connected by Flights')
plt.show()
```

b.

```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
from shapely.geometry import Point

# Generate sample geospatial data
data_geospatial = {
    'City': ['City_1', 'City_2', 'City_3', 'City_4', 'City_5'],
    'Latitude': [42.36, 37.77, 34.05, 40.71, 30.27],
    'Longitude': [-71.06, -122.42, -118.24, -74.01, -97.74]
}

df_geospatial = pd.DataFrame(data_geospatial)
geometry = [Point(lon, lat) for lon, lat in
zip(df_geospatial['Longitude'], df_geospatial['Latitude'])]
gdf = gpd.GeoDataFrame(df_geospatial, geometry=geometry)

# Plot the world map with cities
world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
ax = world.plot(figsize=(12, 8), color='white', edgecolor='black')
gdf.plot(ax=ax, marker='o', color='red', markersize=100, alpha=0.7)
plt.title('Geospatial Distribution of Cities')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```