GOVERNAMENT POLYTECHNIC NAGAMANGALA

Department of computer science and eng.

"5th semester diploma"

Artificial Intalligents And Machine Learning(20CS51)

Assignment-2

Name : ***Aishwarya B .S***

Reg.no : ***158CS22001***

Course co-ordinator :

Jayaramu sir

AIML(20CS51) ASSIGNMENT-WEEK 02

1.Download any two datasets from the internet and perform the following functions.

Dataset:1

a)Aggrigate functions:

--->Head():

```
import pandas as pd
path="/content/sample_data/california_housing_test.csv" df =
pd.read_csv(path) df.head()
```

output:

longitudelatitudehousing_median_agetotal_roomstotal_bedroomspopulationhouseholdsmedian_incomemedian_house_value0-122.0537.3727.03885.0661.01537.0606.06.6085344700.01118.3034.2643.01510.0310.0809.0277.03.5990176500.02117.8133.7827.03589.0507.01484.0495.05.7934270500.03-

118.3633.8228.067.015.049.011.06.1359330000.04-

119.6736.3319.01241.0244.0850.0237.02.937581700.0 *-->tail():*

```
df.tail()
```

output:

	lon	latitud	housing_	total_ro	total_	pop	hous	median	me-	git e	m edian o ms	b
	ulatio	ehol _	in-	dian_house								
u		_age		edroom n			ds	come			_ value	
de				s								
299	119		23.0	1450.	642.	1258				1.1790	2250	
5	.	34.42		0	0		607.0				0	
	86					.0					0.0	

2996	118.14	34.06	27.0	5257.0	1082.0	3496.0	1036.0	3.3906	237200.0
2997	119.70	36.30	10.0	956.0	201.0	693.0	220.0	2.2895	62000.0
2998	117.12	34.10	40.0	96.0	14.0	46.0	14.0	3.2708	162500.0
2999	119.63	34.42	42.0	1765.0	263.0	753.0	260.0	8.5608	500001.0

-->*Sum():*

```
df.sum()
```

output:

```
longitude          -3.587676e+05
latitude            1.069062e+05
housing_median_age  8.653600e+04
total_rooms         7.798736e+06
total_bedrooms      1.589852e+06
population          4.208396e+06
households          1.469736e+06
median_income       1.142182e+04
median_house_value  6.175388e+08
dtype: float64
```

-->Minimum():

```
df.min()
```

Output:

```
longitude          -124.1800
latitude
32.5600
housing_median_age    1.0000
total_rooms           6.0000
```

```
total_bedrooms          2.0000
population              5.0000
households              2.0000
median_income           0.4999
median_house_value    22500.0000
dtype: float64
```

-->Maximum():

```
df.max()
```

output:

```
longitude          -114.4900
latitude             41.9200
housing_median_age      52.0000
total_rooms          30450.0000
total_bedrooms        5419.0000
population           11935.0000 households
4930.0000  median_income          15.0001
median_house_value    500001.0000
dtype: float64
```

-->Count():

```
df.count()
```

output:

```
longitude           3000 latitude
3000  housing_median_age
3000 total_rooms
3000
total_bedrooms       3000 population
3000 households          3000
median_income        3000
median_house_value    3000 dtype:
int64
```

-->Median():

```
df.median()
```

output:

```
longitude           -118.48500
latitude             34.27000
housing_median_age      29.00000
total_rooms          2106.00000
total_bedrooms        437.00000 population
1155.00000 households           409.50000
median_income          3.48715
median_house_value   177650.00000
dtype:
float64
```

-->Mean():

```
df.mean()
```

output:

```
longitude           -119.589200
latitude             35.635390
housing_median_age      28.845333
total_rooms                    2599.578667
total_bedrooms        529.950667
population          1402.798667 households
489.912000  median_income
3.807272
median_house_value   205846.275000
dtype: float64
```

b) Use Map,Filter,Reduce,and Lambda function with pandas dataframes

```
-->import pandas as pd
df = pd.read_csv('/content/sample_data/california_housing_test.csv')
df['longitude'] = df['latitude'].map(lambda x: x * 1.10) print(df)
```

output:

```
   longitude  latitude  housing_median_age  total_rooms  total_bedrooms
\
0     41.107     37.37                27.0       3885.0           661.0
1     37.686     34.26                43.0       1510.0           310.0
```

```
2     37.158     33.78                27.0       3589.0           507.0
3     37.202     33.82                28.0         67.0            15.0
4        39.963     36.33                19.0       1241.0
244.0
...          ...       ...                 ...          ...
...
2995  37.862     34.42                23.0       1450.0           642.0
2996  37.466     34.06                27.0       5257.0          1082.0
2997  39.930     36.30                10.0        956.0           201.0
      2998 37.510     34.10                40.0         96.0            14.0
      2999 37.862     34.42                42.0       1765.0           263.0

      population  households  median_income  median_house_value     0
1537.0        606.0          6.6085            344700.0
1     809.0        277.0          3.5990            176500.0
2     1484.0        495.0          5.7934            270500.0
3     49.0         11.0          6.1359            330000.0
4     850.0        237.0          2.9375             81700.0...     ...
      ...           ...                 ...    2995 1258.0        607.0
      1.1790            225000.0
2996 3496.0      1036.0          3.3906            237200.0    2997
693.0        220.0          2.2895             62000.0    2998 46.0
14.0          3.2708            162500.0    2999 753.0        260.0
8.5608            500001.0
```

```
-->filtered_df = df[df['households'].map(lambda x: x >=30)]
```

`print(filtered_df)` output:

```
longitude  latitude  housing_median_age  total_rooms  total_bedrooms
\
0     41.107     37.37                27.0       3885.0           661.0
1     37.686     34.26                43.0       1510.0           310.0
2     37.158     33.78                27.0       3589.0           507.0
4        39.963     36.33                19.0       1241.0           244.0
5        40.161     36.51                37.0       1018.0
213.0
...          ...       ...                 ...          ...
```

```
2997      39.930      36.30                  10.0         956.0
201.0
2999      37.862      34.42                  42.0        1765.0
...
2994      37.246     33.86                35.0        931.0          181.0
2995      37.862     34.42                23.0       1450.0          642.0
2996      37.466     34.06                27.0       5257.0         1082.0
          263.0           population  households  median_income
median_house_value    0           1537.0         606.0           6.6085
344700.0
1           809.0        277.0          3.5990             176500.0
2           1484.0        495.0          5.7934             270500.0
4           850.0        237.0          2.9375              81700.0
5           663.0        204.0          1.6635              67000.0   ...
            ...          ...              ...                  ...
2994        516.0        174.0          5.5867             182500.0
2995        1258.0        607.0          1.1790             225000.0
2996        3496.0       1036.0          3.3906             237200.0
2997        693.0        220.0         2.2895              62000.0   2999
            753.0        260.0          8.5608             500001.0
```

```
from functools import reduce if not
filtered_df['population'].empty:  # Check if 'population' column
is empty      total = reduce(lambda x, y: x + y,
filtered_df['population'])    print(total) else:
    print("The filtered DataFrame is empty, cannot calculate total
calculate.")
```

output:

```
    4199858.0
```

```
-->import pandas as pd
# read CSV file into Dataframe df =
pd.read_csv('/content/sample_data/california_housing_test.csv')
df['population'] = df['population'].map(lambda x: x * 1.10) print(df)
```

output:

```
longitude  latitude  housing_median_age  total_rooms  total_bedrooms
\
0       -122.05     37.37                27.0       3885.0          661.0
1       -118.30      34.26                  43.0       1510.0
```

```
310.0
2        -117.81     33.78                27.0       3589.0           507.0
         3 -118.36     33.82                28.0         67.0            15.0
4        -119.67     36.33                  19.0       1241.0
244.0
...          ...        ...                   ...          ...
...
2995  -119.86     34.42                  23.0       1450.0           642.0
2996  -118.14     34.06                  27.0       5257.0          1082.0
2997  -119.70     36.30                  10.0        956.0           201.0
2998  -117.12     34.10                  40.0         96.0            14.0
2999  -119.63     34.42                  42.0       1765.0           263.0
      population  households  median_income  median_house_value    0
1690.7        606.0          6.6085             344700.0
1             889.9        277.0          3.5990             176500.0
2             1632.4        495.0          5.7934             270500.0
3             53.9         11.0          6.1359             330000.0
4             935.0        237.0          2.9375              81700.0   ...
              ...          ...              ...                 ...
2995         1383.8        607.0          1.1790             225000.0
2996         3845.6       1036.0          3.3906             237200.0
2997         762.3        220.0          2.2895              62000.0
2998         50.6         14.0          3.2708             162500.0   2999
             828.3        260.0          8.5608              500001.0
```

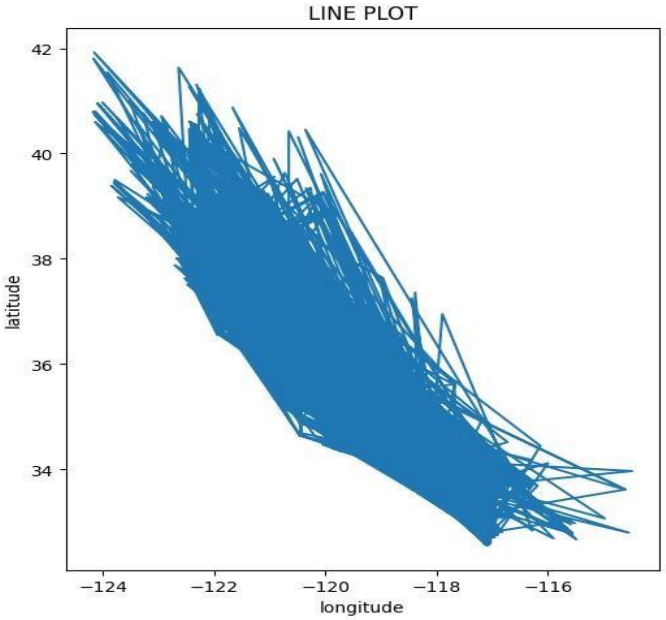
c) Visualize the dataset(At least 6 different plots).

1)line plot

```
import pandas as pd
# load the CSV file into DataFrame
df=pd.read_csv('/content/sample_data/california_housing_test.csv')
```

```
import matplotlib.pyplot as plt import
seaborn as sns # Set plot size
plt.figure(figsize=(20, 15))  #1. Line
plot plt.subplot(2, 3, 1)
plt.plot(df['longitude'],
df['latitude']) plt.title("LINE PLOT")
plt.xlabel('longitude')
plt.ylabel('latitude')
```

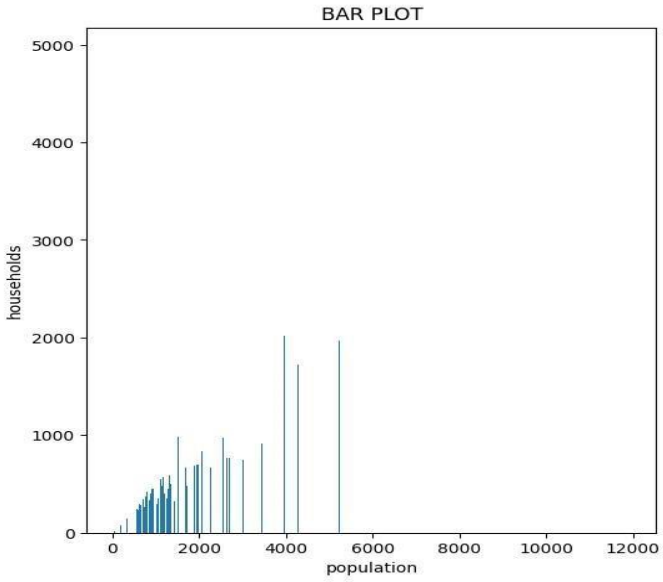
output:



2)Bar plot

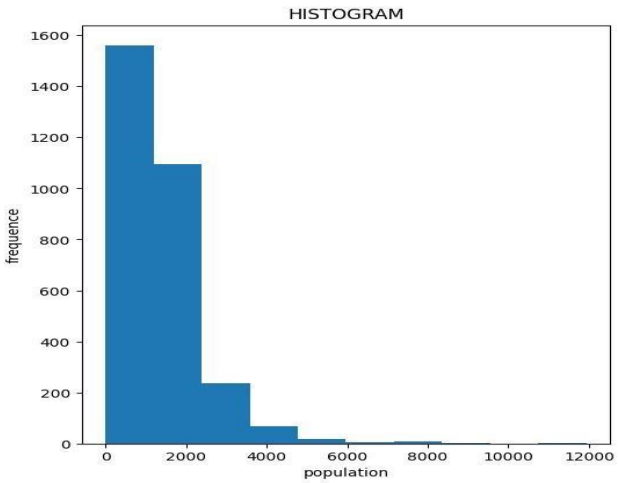
```
import pandas as pd
# load the CSV file into DataFrame
df=pd.read_csv('/content/sample_data/california_housing_test.csv')
import matplotlib.pyplot as plt
# Set plot size
plt.figure(figsize=(20, 15))
#2. Bar plot plt.subplot(2, 3, 2)
plt.bar(df['population'], df['households'])
plt.title("BAR PLOT") plt.xlabel('population')
plt.ylabel('households')
```

output:

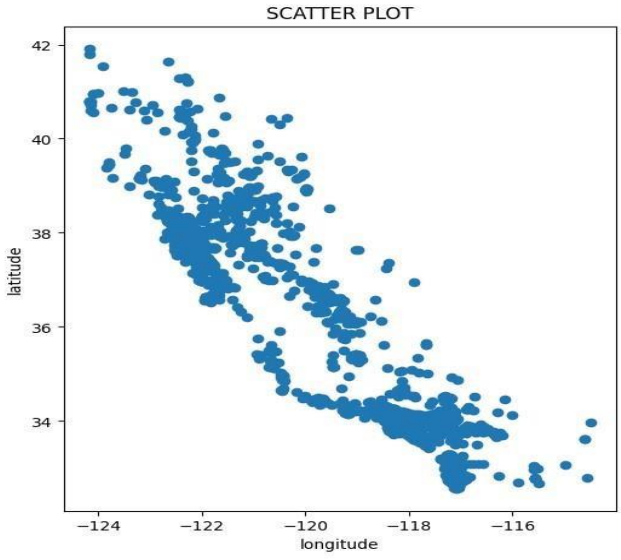


3)histogram

```
import pandas as pd df =
pd.read_csv('/content/sample_data/california_housing_test.csv')
import matplotlib.pyplot as plt import seaborn as sns
plt.figure(figsize=(20, 15)) plt.subplot(2, 3, 3)
plt.hist(df['population']) plt.title("HISTOGRAM")
plt.xlabel('population') plt.ylabel('frequence') plt.show()
```

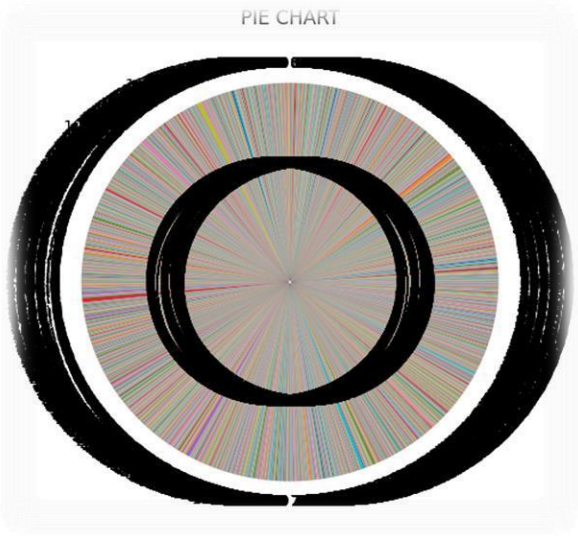



```
4)scatter plot:
 import pandas as pd
df=pd.read_csv('/content/sample_data/california_housing_test.csv')
import matplotlib.pyplot as plt import seaborn as sns
plt.figure(figsize=(20, 15)) plt.subplot(2, 3, 4)
plt.scatter(df['longitude'], df['latitude']) plt.title("SCATTER PLOT")
plt.xlabel('longitude') plt.ylabel('latitude')
```



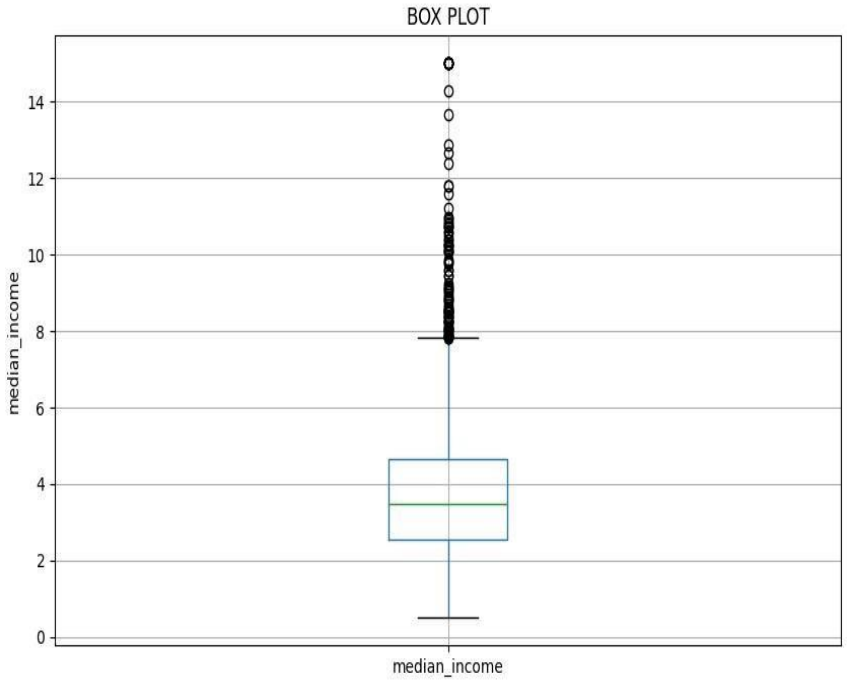
5)pie chart

```
Import pandas as pd
# add the CSV file into DataFrame
df=pd.read_csv('/content/sample_data/california_housing_test.csv')
import matplotlib.pyplot as plt import seaborn as sns
plt.figure(figsize=(20, 15)) plt.subplot(2, 3, 6)
plt.pie(df['households'], labels=df['population'], autopct='%1.1f%%')
plt.title('PIE CHART')
```



6)box plot

```
import pandas as pd
df=pd.read_csv('/content/sample_data/california_housing_test.csv')
import matplotlib.pyplot as plt import seaborn as sns
plt.figure(figsize=(20, 15)) plt.figure(figsize=(10, 6))
df.boxplot(column=['median_income']) plt.title("BOX PLOT")
plt.ylabel('median_income')
```



dataset:2

a)Aggrigate functions:

-->head():

```
import pandas as pd path='/content/P1-UKBank-
Customers[1].csv' df=pd.read_csv(path) df.head()
```

output:

Customer IDNameSurnameGenderAgeRegionJob ClassificationDate JoinedBalance0100000001SimonWalshMale21EnglandWhite Collar05.Jan.15113810.151400000002JasmineMillerFemale34Northern IrelandBlue Collar06.Jan.1536919.732100000003LiamBrownMale46EnglandWhite Collar07.Jan.15101536.833300000004TrevorParrMale32WalesWhite

Collar08.Jan.151421.524100000005 DeirdrePullmanFemale38EnglandBlue Collar09.Jan.1535639.79

-->tail:

```
df.tail()
```

output:

Customer IDNameSurnameGenderAgeRegionJob ClassificationDate JoinedBalance4009200004010SamLewisMale64ScotlandOther30.Dec.1519711.664 010200004011KeithHughesMale52ScotlandBlue Collar30.Dec.1556069.724011200004012HannahSpringerFemale50ScotlandOther3 0.Dec.1559477.824012200004013ChristianReidMale51ScotlandBlue Collar30.Dec.15239.454013300004014StephenMayMale33WalesBlue Collar30.Dec.1530293.19

-->sum:

```
df.sum()
```

output:

```
Customer ID                                  681108058105
Name                 SimonJasmineLiamTrevorDeirdreAvaDorothyLisaRut...
Surname                WalshMillerBrownParrPullmanColemanThomsonKnoxC...
Gender               MaleFemaleMaleMaleFemaleFemaleFemaleFemaleFema...
Age                                          154985
Region               EnglandNorthern IrelandEnglandWalesEnglandWale...
Job Classification    White CollarBlue CollarWhite CollarWhite Colla...
Date Joined           05.Jan.1506.Jan.1507.Jan.1508.Jan.1509.Jan.150...
Balance                                  159622523.37 dtype: object
```

-->Minimum:

```
df.min()
```

output:

Customer ID 100000001
Name Abigail
Surname Abraham
Gender Female
Age 15
Region England
Job Classification Blue Collar Date Joined 01.Apr.15

Balance 11.52 dtype: object

-->Maximum:

```
df.max()
```

output:

```
Customer ID          400003848
Name                     Zoe
Surname                Young
Gender                  Male
Age                       64
Region                 Wales
Job Classification  White Collar
Date Joined           31.Oct.15
Balance           183467.7 dtype:
object
```

-->Count:

```
df.count()
```

output:

```
Customer ID        4014
Name               4014
Surname            4014
Gender             4014
Age                4014
Region             4014
Job Classification 4014

4014
```

b)use map,reduce,filter and lambda functionwith pandas data frames

```
import pandas as pd

❼df=pd.read_csv('/content/P1-UK-Bank-
Customers[1].csv') df['Age'] = df['Age'].map(lambda x:
x * 1.10)  print(df)
```

output:

```
Customer ID       Name    Surname  Gender    Age            Region  \
```

```
0     100000001      Simon     Walsh    Male  23.1           England
1     400000002    Jasmine    Miller  Female  37.4  Northern Ireland
2     100000003       Liam     Brown    Male  50.6           England
3     300000004     Trevor      Parr    Male  35.2             Wales
4     100000005    Deirdre   Pullman  Female  41.8           England
...           ...        ...       ...     ...   ...               ...
4009    200004010        Sam     Lewis    Male  70.4          Scotland
4010    200004011      Keith    Hughes    Male  57.2          Scotland
4011    200004012     Hannah  Springer  Female  55.0          Scotland
4012    200004013  Christian      Reid    Male  56.1          Scotland
4013    300004014    Stephen       May    Male  36.3             Wales
```

```
     Job Classification Date Joined    Balance
0           White Collar   05.Jan.15  113810.15
1           Blue Collar   06.Jan.15   36919.73
2           White Collar   07.Jan.15  101536.83
3           White Collar   08.Jan.15    1421.52
4           Blue Collar   09.Jan.15   35639.79   ...
            ...         ...        ...
4009              Other   30.Dec.15   19711.66
4010              Blue Collar   30.Dec.15   56069.72
4011              Other   30.Dec.15   59477.82
4012              Blue Collar   30.Dec.15     239.45
4013              Blue Collar   30.Dec.15   30293.19
```

```
-->filtered_df = df[df['Age'].map(lambda x: x >=30)]
print(filtered_df)
```

output:

```
Customer ID       Name   Surname  Gender  Age           Region  \
1     400000002    Jasmine    Miller  Female   34  Northern Ireland
2     100000003       Liam     Brown    Male   46           England
3     300000004     Trevor      Parr    Male   32             Wales
4     100000005    Deirdre   Pullman  Female   38           England
5     300000006        Ava   Coleman  Female   30             Wales
...           ...        ...       ...     ...  ...               ...
4009 200004010        Sam     Lewis    Male   64          Scotland
4010    200004011      Keith    Hughes    Male   52          Scotland
4011    200004012     Hannah  Springer  Female   50          Scotland
4012    200004013  Christian      Reid    Male   51          Scotland
4013    300004014    Stephen       May    Male   33             Wales
```

```
     Job Classification Date Joined    Balance
```

```
1           Blue Collar   06.Jan.15   36919.73
2           White Collar   07.Jan.15  101536.83
3           White Collar   08.Jan.15    1421.52
4           Blue Collar   09.Jan.15   35639.79
5           Blue Collar   09.Jan.15  122443.77   ...
            ...         ...        ...
4009              Other   30.Dec.15   19711.66
4010              Blue Collar   30.Dec.15   56069.72
4011              Other   30.Dec.15   59477.82
4012              Blue Collar   30.Dec.15     239.45
4013              Blue Collar   30.Dec.15   30293.19
```

```
                    from functools import
              reduce    not
              filtered_df['Age'].empty:
-->
```

```
if total = reduce(lambda x, y: x + y,
       filtered_df['Age'])  print(total)
```

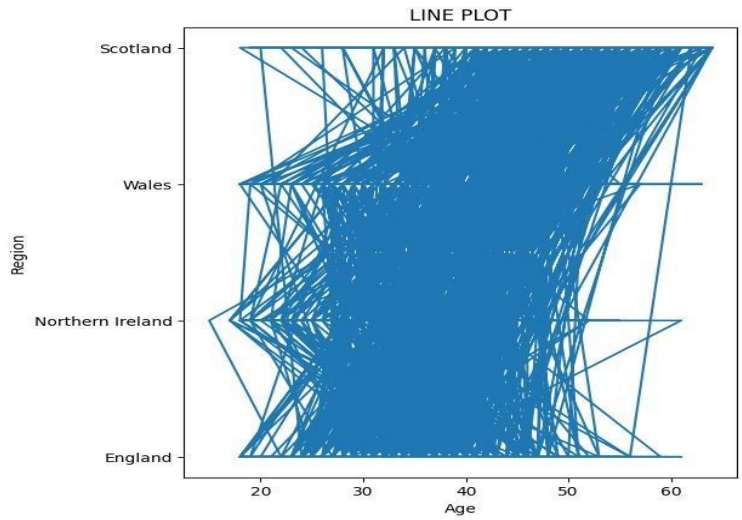
output:

```
136716
```

c)Visualize the data sets(atleast 6 different plots

line plot:

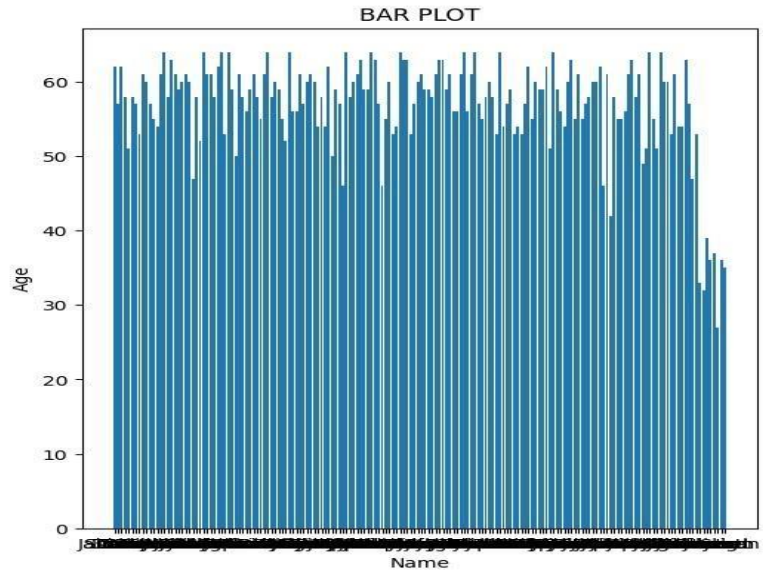
```
import pandas as pd df=pd.read_csv('/content/P1-
UKBank-Customers.csv') import matplotlib.pyplot as
plt import seaborn as sns plt.figure(figsize=(20,
15)) plt.subplot(2, 3, 1) plt.plot(df['Age'],
df['Region']) plt.title("LINE PLOT")
plt.xlabel('Age') plt.ylabel('Region')
```



Bar plot:

```
import pandas as pd df=pd.read_csv('/content/P1-
UKBank-Customers.csv') import matplotlib.pyplot as
plt import seaborn as sns plt.figure(figsize=(20,
15)) plt.subplot(2,3,2) plt.bar(df['Name'],
df['Age']) plt.title("BAR PLOT") plt.xlabel('Name')
plt.ylabel('Age')
```

output:

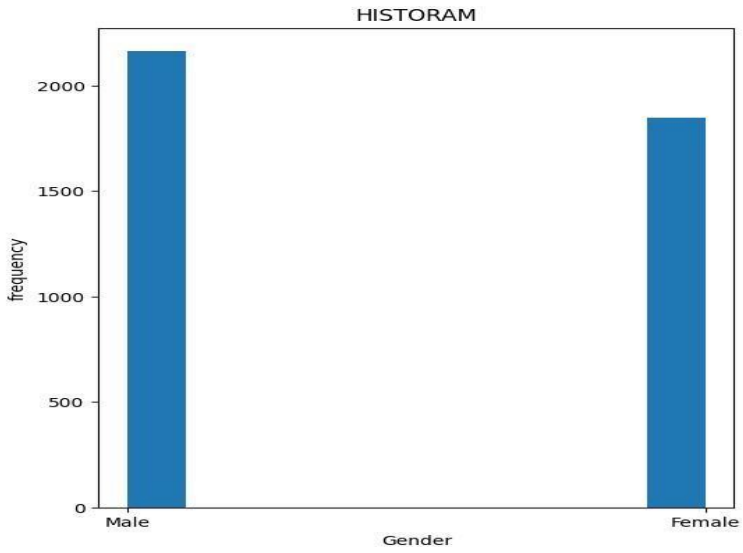


Histogram:

```
import pandas as pd df=pd.read_csv('/content/P1-
UKBank-Customers.csv') import matplotlib.pyplot as
plt import seaborn as sns plt.figure(figsize=(20,
15)) plt.subplot(2, 3, 3) plt.hist(df['Gender'])
```

```
plt.title("HISTORAM") plt.xlabel('Gender')
plt.ylabel('frequency')
```

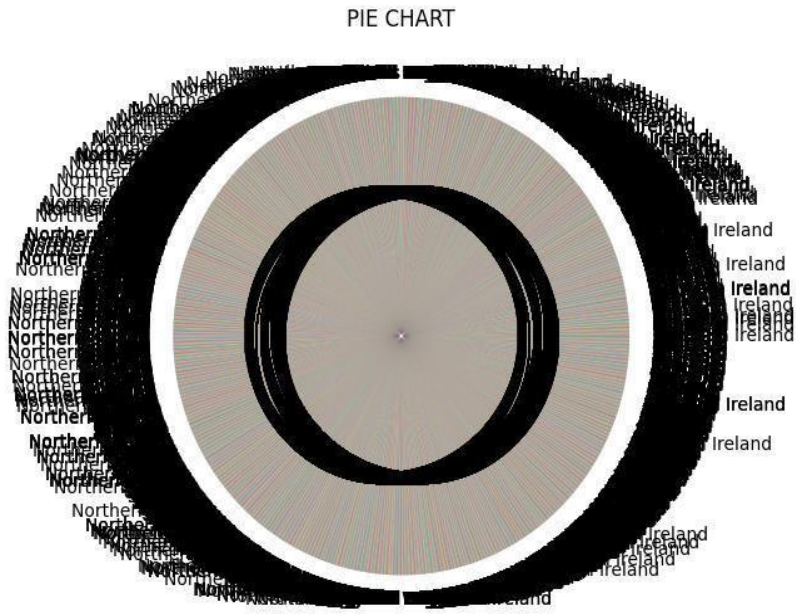
output:



Pie chart:

```
import pandas as pd df=pd.read_csv('/content/P1-UK-
BankCustomers.csv') import matplotlib.pyplot as plt import
seaborn as sns plt.figure(figsize=(20, 15)) plt.subplot(2,
3, 6) plt.pie(df['Age'], labels=df['Region'],
autopct='%1.1f%%') plt.title('PIE CHART')
```

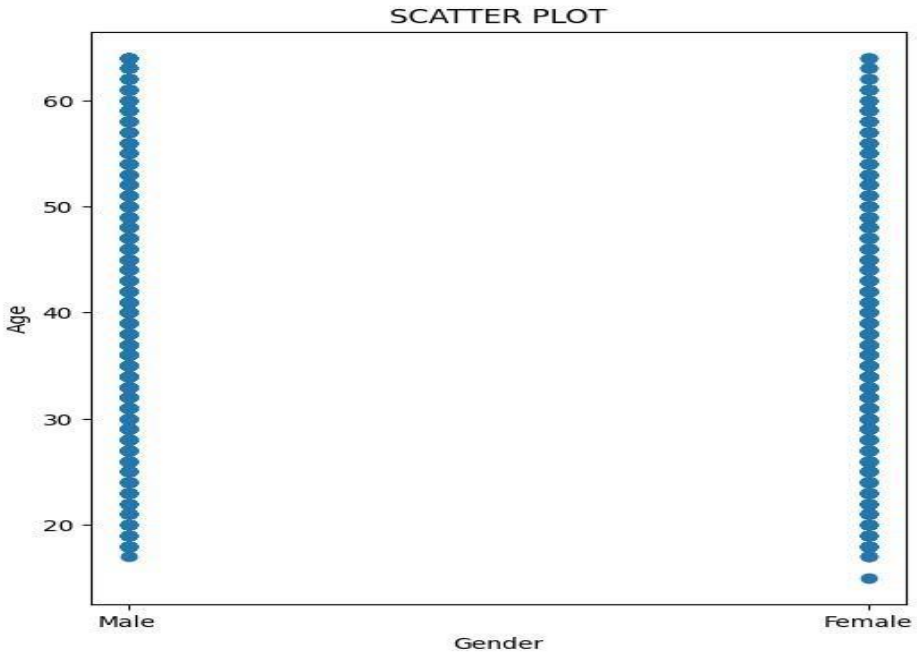
output:



Scatter plot:

```
import pandas as pd df=pd.read_csv('/content/P1-
UKBank-Customers.csv') import matplotlib.pyplot as
plt import seaborn as sns plt.figure(figsize=(20,
15)) plt.subplot(2, 3, 4) plt.scatter(df['Gender'],
df['Age']) plt.title("SCATTER PLOT")
plt.xlabel('Gender') plt.ylabel('Age')
```

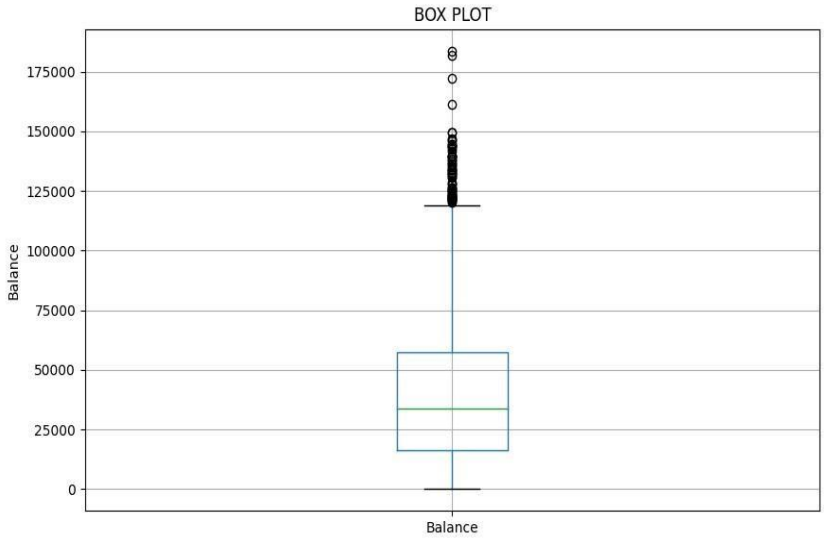
output:



Box plot:

```
import pandas as pd df=pd.read_csv('/content/P1-UKBank-
Customers.csv') import matplotlib.pyplot as plt import
seaborn as sns plt.figure(figsize=(10, 6))
df.boxplot(column=['Balance']) plt.title("BOX
PLOT") plt.ylabel('Balance')
```

output:



d) How do you create a project plan and product backlog for an AI project? b. Classification Project – ML / deep learning.

Creating a project plan and product backlog for an AI project, specifically a Classification Project using Machine Learning (ML) or deep learning, involves several key steps to ensure clarity, organization, and success throughout the project lifecycle. Here's a structured approach:

1. Define Project Objectives and Scope

- **Objective Definition:** Clearly articulate what the project aims to achieve (e.g., classify images into specific categories).
- **Scope Definition:** Determine the boundaries of the project in terms of features, data sources, and technology stack (e.g., using TensorFlow for deep learning).

2. Identify Stakeholders and Gather Requirements

- **Stakeholder Identification:** Identify who will be impacted by the project and who needs to be involved (e.g., data scientists, domain experts, project managers).
- **Requirements Gathering:** Collect detailed functional and non-functional requirements (e.g., accuracy threshold, performance metrics).

3. Create a Product Backlog

- **List User Stories:** Create user stories that describe the features from an end-user perspective (e.g., "As a user, I want the model to correctly classify images with at least 90% accuracy.").
- **Prioritize:** Prioritize user stories based on business value, dependencies, and criticality.
- **Estimate:** Estimate the effort (e.g., story points) required for each user story.

4. Define Epics and Tasks

- **Epics:** Group related user stories into epics (e.g., "Image preprocessing," "Model training," "Model evaluation").
- **Tasks:** Break down each epic into smaller, manageable tasks (e.g., "Collect and preprocess training data," "Implement deep learning model architecture").

5. Create the Project Plan

- **Task Sequencing:** Sequence tasks in a logical order, considering dependencies (e.g., data preprocessing before model training).
- **Timeline:** Estimate the duration for each task or phase based on team capacity and dependencies.

- **Milestones:** Define key milestones (e.g., completion of data preprocessing, model training, validation) to track progress.

6. Determine Tools and Infrastructure

- **Tools Selection:** Decide on tools and frameworks for data preprocessing, model development, training, and evaluation (e.g., Python, TensorFlow, scikit-learn).
- **Infrastructure Requirements:** Identify and provision necessary hardware (e.g., GPU servers) and software environments.

7. Risk Assessment and Mitigation

- **Identify Risks:** List potential risks such as data quality issues, model overfitting, or technology limitations.
- **Mitigation Strategies:** Develop strategies to mitigate risks (e.g., regular data validation, early validation with a smaller dataset).

8. Establish Monitoring and Evaluation Mechanisms

- **Performance Metrics:** Define metrics (e.g., accuracy, precision, recall) to evaluate model performance.
- **Monitoring:** Set up mechanisms to monitor model performance in real-time or periodically (e.g., automated testing, logging).

9. Plan for Iterative Development and Feedback

- **Iterative Development:** Plan for iterative model development based on feedback and evaluation results.
- **Feedback Loop:** Establish a feedback loop with stakeholders and end-users to incorporate their input and refine requirements.

10. Communication and Collaboration

- **Communication Plan:** Define how progress, issues, and changes will be communicated (e.g., regular status meetings, shared documentation).
- **Collaboration Tools:** Use collaboration tools (e.g., Jira, Trello, GitHub) to manage tasks, track progress, and facilitate team communication.

By following these steps, you can create a comprehensive project plan and product backlog for your AI Classification Project, ensuring alignment with business objectives, clarity in execution, and effective management of resources and risks throughout the project lifecycle.

*****