import libraries to visualize, analyze, plot and detecting errors for giving dataset

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

load data (Customer Demography) and save in variable called "cdg"

```
cdg = pd.read_csv("/content/customer demograpy.csv")
cdg
```

	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	
0	Laraine	Medendorp	F	93	1953-10-12	
1	Eli	Bockman	Male	81	1980-12-16	A
2	Arlin	Dearle	Male	61	1954-01-20	
3	Talbot	NaN	Male	33	1961-10-03	
4	Sheila-kathryn	Calton	Female	56	1977-05-13	
...	...	...	...	...	...	...
3995	Rosalia	Halgarth	Female	8	1975-08-09	
3996	Blanch	Nisuis	Female	87	2001-07-13	
3997	Sarene	Woolley	U	60	NaN	
3998	Patrizius	NaN	Male	11	1973-10-24	
3999	Kippy	Oldland	Male	76	1991-11-05	

4000 rows × 12 columns

Basic Panda Commands

```
#print information of data
cdg.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2630 entries, 0 to 3996
Data columns (total 12 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
0   first_name                           2630 non-null   object
1   last_name                            2630 non-null   object
2   gender                               2630 non-null   object
3   past_3_years_bike_related_purchases  2630 non-null   int64
4   Age                                  2630 non-null   int64
5   job_title                            2630 non-null   object
6   job_industry_category                2630 non-null   object
7   wealth_segment                       2630 non-null   object
8   deceased_indicator                   2630 non-null   object
9   default                              2630 non-null   object
```

```
 10  owns_car                             2630 non-null   object  11  tenure
     2630 non-null   float64 dtypes: float64(1), int64(2), object(9) memory usage:
     267.1+ KB
```

```
#print statistical values of given dataset

cdg.describe()
```

	past_3_years_bike_related_purchases	Age	tenure
count	2630.000000	2630.000000	2630.000000
mean	49.363498	45.452091	10.671483
std	28.841657	12.497325	5.676862
min	0.000000	21.000000	1.000000
25%	25.000000	36.000000	6.000000
50%	48.500000	46.000000	10.000000
75%	74.000000	55.000000	16.000000
max	99.000000	91.000000	22.000000

```
#print first 5 rows of given dataset

cdg.head()
```

	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	jo
0	Laraine	Medendorp	F	93	1953-10-12	E S
1	Eli	Bockman	Male	81	1980-12-16	Adm
2	Arlin	Dearle	Male	61	1954-01-20	R
3	Talbot	NaN	Male	33	1961-10-03	
4	Sheila-Calton		Female	56	1977-Sen kathryn 05-13	

```
#print last 5 rows of given dataset

cdg.tail()
```

	first_name	last_name	gender	past_3_years_bike_related_purchases	DOB	
3995	Rosalia	Halgarth	Female	8	1975-08-09	M
3996	Blanch	Nisuis	Female	87	2001-07-13	S
3997	Sarene	Woolley	U	60	NaN	
3998	Patrizius	NaN	Male	11	1973-10-24	
3999	Kippy	Oldland	Male	76	199111-05	

```
#print no.of Rows and Columns

cdg.shape
```

```
(4000, 12)
```

```
#print column names

cdg.columns
```

```
Index(['first_name', 'last_name', 'gender',
       'past_3_years_bike_related_purchases', 'DOB', 'job_title',
 'job_industry_category', 'wealth_segment', 'deceased_indicator',
       'default', 'owns_car', 'tenure'],
 dtype='object')
```

```
#This command shows how many missing values in all columns

cdg.isnull().sum()
```

```
first_name                             0
last_name                            125
gender                                 0
past_3_years_bike_related_purchases    0
DOB                                   87
job_title                            506
job_industry_category                656
wealth_segment                         0
deceased_indicator                     0
default                              302
owns_car                               0
tenure                                87
dtype: int64
```

```
# This command drops missing values and show summary if there is any missing

values cdg.dropna(inplace = True) cdg.isnull().sum()
```

```
first_name                           0
last_name                            0
gender                               0
past_3_years_bike_related_purchases  0
DOB                                  0
job_title                            0
job_industry_category                0
wealth_segment                       0
deceased_indicator                   0
default                              0
owns_car                             0
tenure                               0
dtype: int64
```

```
#print Rows and Columns after removing missing value

cdg.shape
```

```
(2630, 12)
```

```
#This command shows how many duplicate values in given dataset

cdg.duplicated().sum()
```

```
0
```

Finding and correcting gender errors

```
#This command Shows Error values in Gender
#column

gender = cdg.groupby(['gender']) gender.size()
```

```
gender
F            1
Femal        1
Female    1366
Male      1262
dtype: int64
```

```
#Replace all error values to correct values


cdg['gender'] =
cdg['gender'].replace(['F','Femal'],['Femal','Female'])

#This command shows values of Gender column after correcting the
#errors

gender = cdg.groupby(['gender'])

gender.size()
```

```
gender
Female    1368
Male      1262
dtype: int64
```

Detecting DOB columns for correcting date of birth value and convert it into corresponding age.

```
#This command “find errors in DOB columns"
from datetime import datetime,
date born ='1953-10-12' print("Born :", born)
born = datetime.strptime(born, "%Y-%m-%d").date()
today = date.today() print("Age :", today.year - born.year - ((today.month,today.day) <
(born.month,born.day)))
```

```
Born : 1953-10-12
Age : 69
```

```
# This commands "convert date of birth into Current age"
from datetime import datetime, date
for i in cdg['DOB']:
    print(i)
    born = i
    born =datetime.strptime(born,"%Y-%m-%d").date()
    #Get today's date
    today = date.today()
    j = today.year - born.year - ((today.month, today.day) < (born.month, born.day))
    cdg['DOB'] = cdg['DOB'].replace([i],[j])
```

```
# This command "change column name" (DOB into Age)

cdg.rename(columns = {'DOB': 'Age'}, inplace=True)
cdg
```

	first_name	last_name	gender	past_3_years_bike_related_purchases
0	Laraine	Medendorp	F	93
1	Eli	Bockman	Male	81
2	Arlin	Dearle	Male	61
3	Talbot	NaN	Male	33
4	Sheila-kathryn	Calton	Female	56
...	...	...	...	...
3995	Rosalia	Halgarth	Female	8
3996	Blanch	Nisuis	Female	87
3997	Sarene	Woolley	U	60
3998	Patrizius	NaN	Male	11
3999	Kippy	Oldland	Male	76

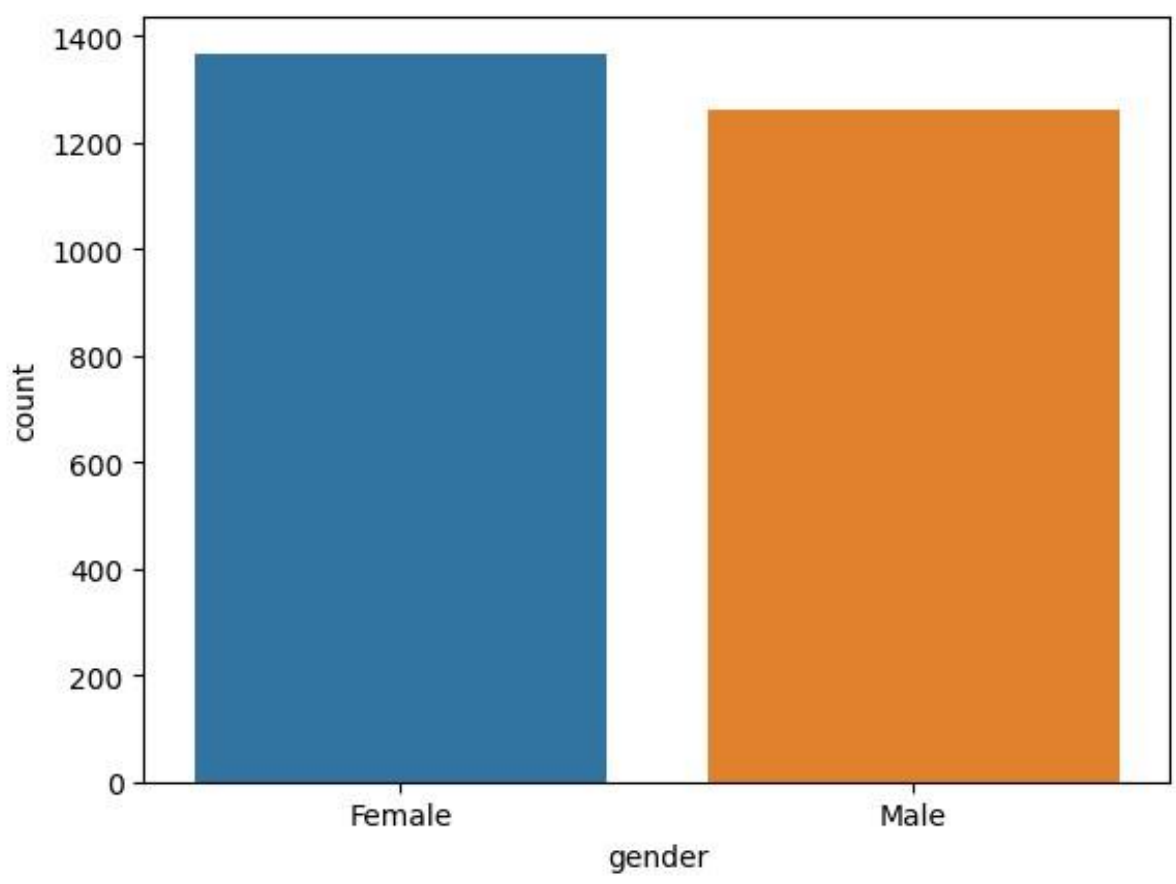
4000 rows × 12 columns

Visualize data in the form of Graphical view

```
# Count plot of "gender" column

sns.countplot(x= cdg['gender'], data=cdg)
```

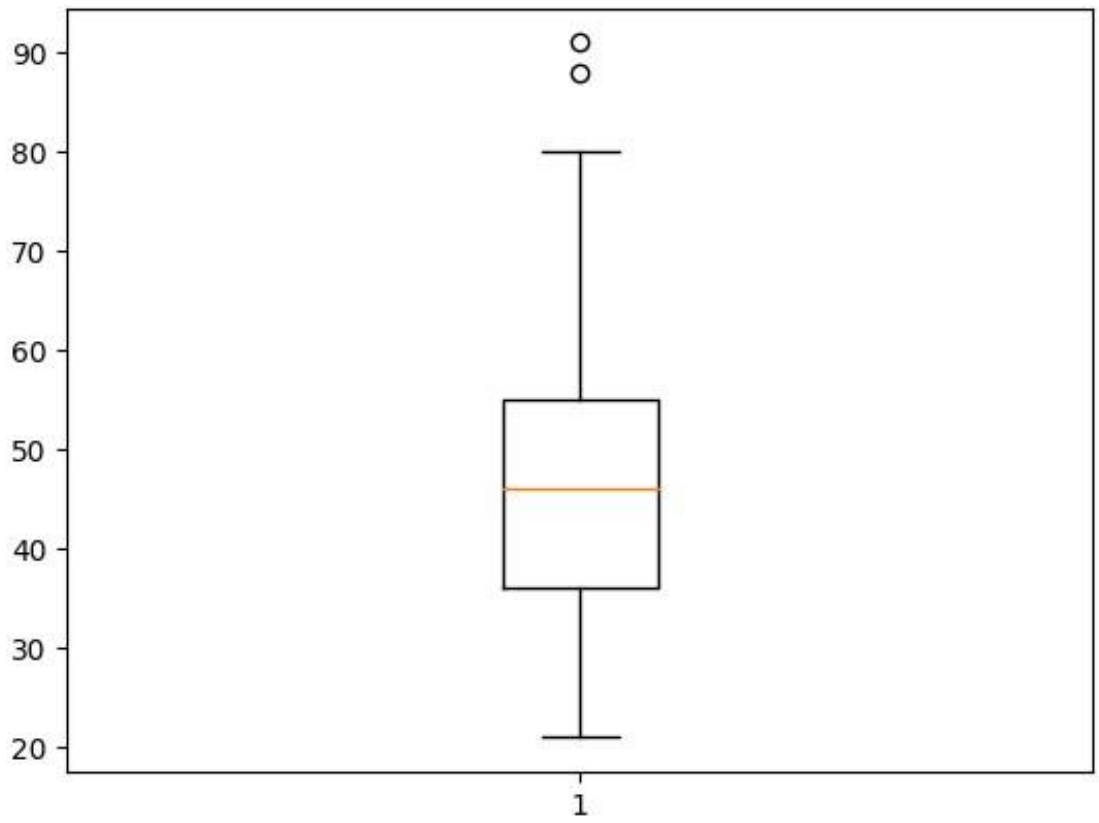
```
<Axes: xlabel='gender', ylabel='count'>
```



```
# box plot of "Age" Column

plt.boxplot(cdg['Age'])
```

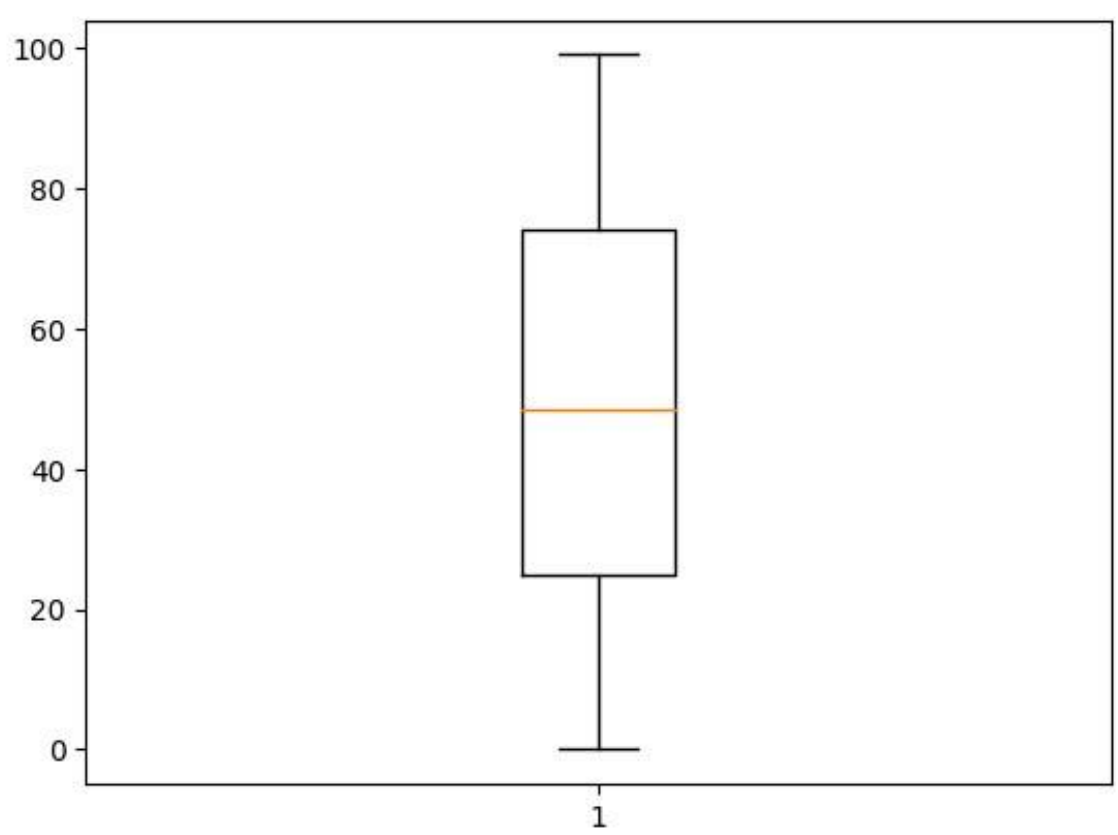
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7950c2002290>,
  <matplotlib.lines.Line2D at 0x7950c2002530>],
 'caps': [<matplotlib.lines.Line2D at 0x7950c20027d0>,
  <matplotlib.lines.Line2D at 0x7950c2002a70>],
 'boxes': [<matplotlib.lines.Line2D at 0x7950c2001ff0>],
 'medians': [<matplotlib.lines.Line2D at 0x7950c2002d10>],
 'fliers': [<matplotlib.lines.Line2D at 0x7950c2002fb0>],
 'means': []}
```




```
# box plot of "past_3_years_bike_related_purchases" Column

plt.boxplot(cdg['past_3_years_bike_related_purchases'])
```

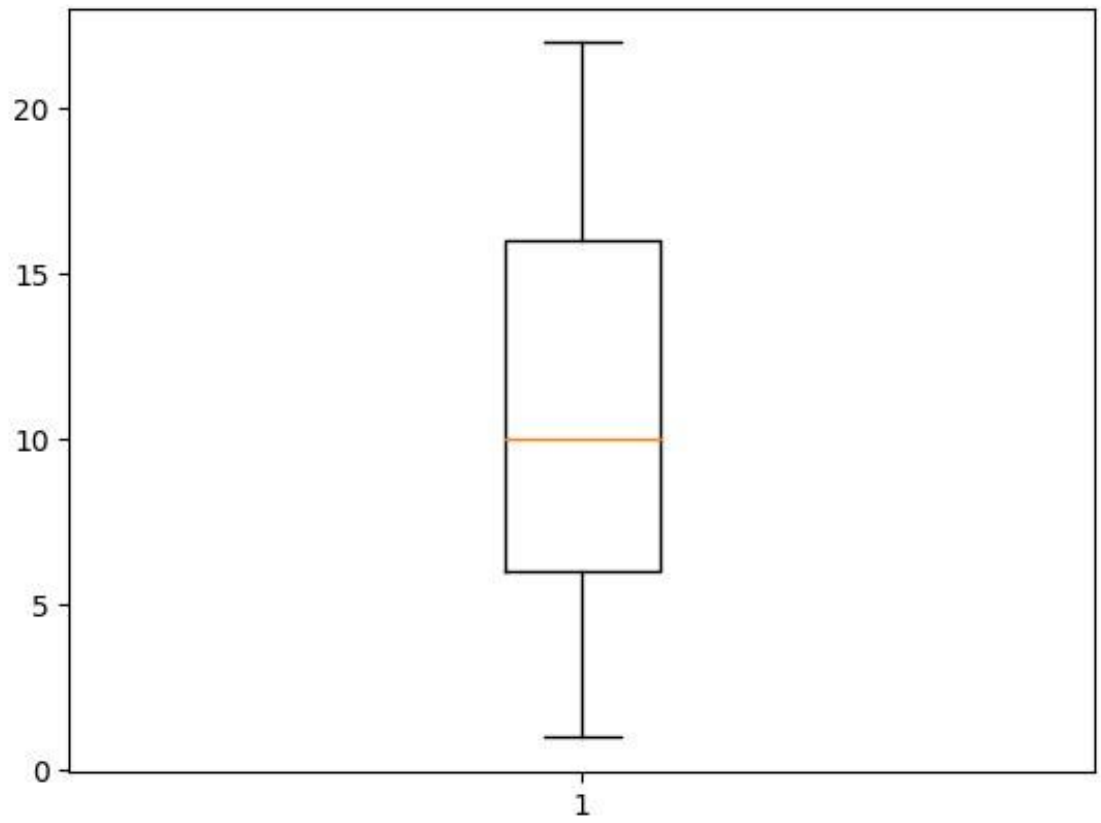
```
{'whiskers': [<matplotlib.lines.Line2D at 0x7950c1e821a0>,
  <matplotlib.lines.Line2D at 0x7950c1e82440>],
 'caps': [<matplotlib.lines.Line2D at 0x7950c1e826e0>,
  <matplotlib.lines.Line2D at 0x7950c1e82980>],
 'boxes': [<matplotlib.lines.Line2D at 0x7950c1e81f00>],
 'medians': [<matplotlib.lines.Line2D at 0x7950c1e82c20>],
 'fliers': [<matplotlib.lines.Line2D at 0x7950c1e82ec0>],
 'means': []}
```



```
# box plot of "tenure" Column

plt.boxplot(cdg['tenure']) {'whiskers':
[<matplotlib.lines.Line2D at 0x7950c1efce20>,

       <matplotlib.lines.Line2D at 0x7950c1efd0c0>],
      'caps': [<matplotlib.lines.Line2D at 0x7950c1efd360>,
       <matplotlib.lines.Line2D at 0x7950c1efd600>],
      'boxes': [<matplotlib.lines.Line2D at 0x7950c1efcb80>],
      'medians': [<matplotlib.lines.Line2D at 0x7950c1efd8a0>],
'fliers': [<matplotlib.lines.Line2D at 0x7950c1efdb40>],
'means': []}
```

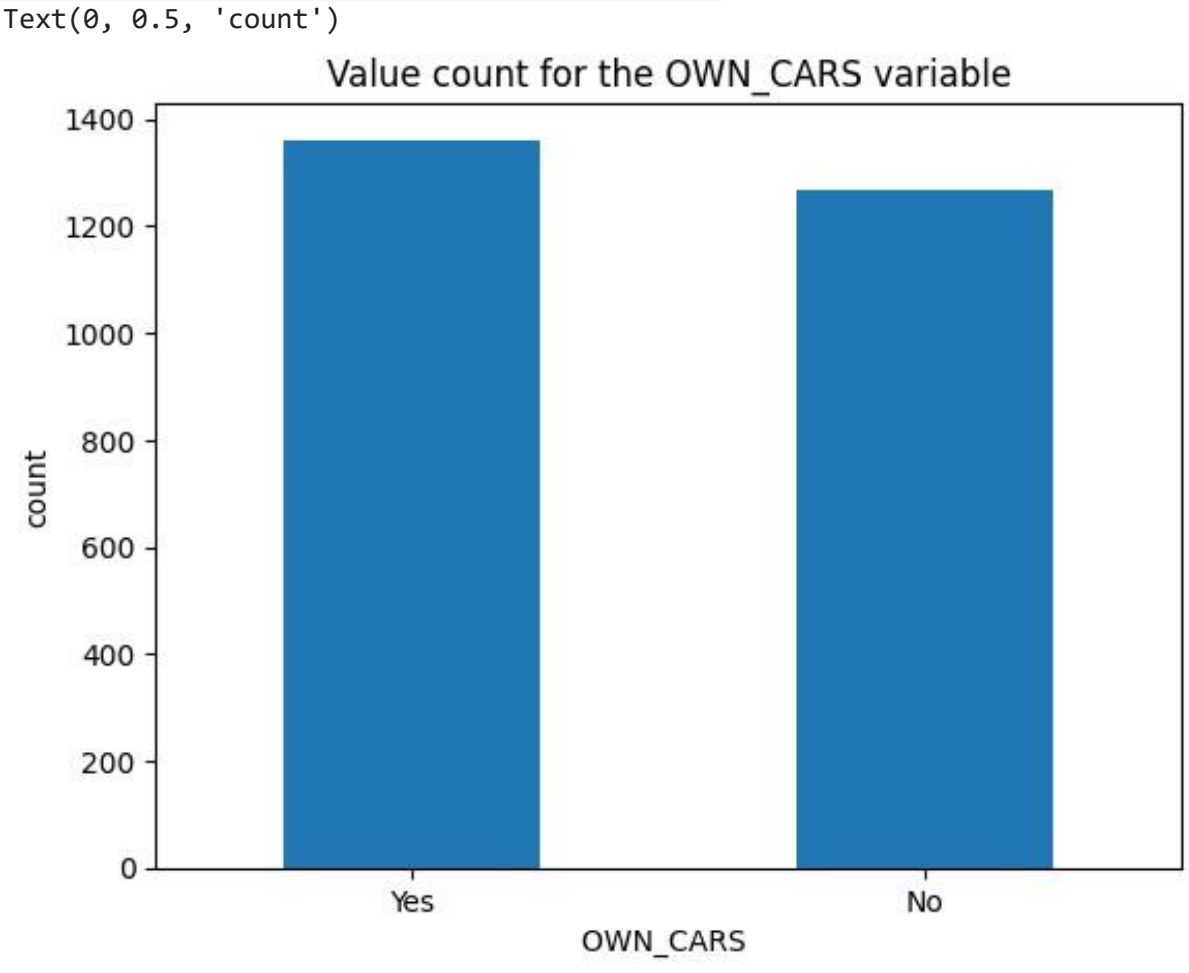



```
#This command counts values which is in
#“owns_car” column

Cdg.owns_car.value_counts()
    Yes    1361
    No     1269
    Name: owns_car, dtype: int64

#Bar chart of “owns_car” Column

Cdg.owns_car.value_count().plot(kind = “bar”)
plt.title("Value count for the OWN_CARS
variable") plt.xlabel("OWN_CARS ")
plt.xticks(rotation = 0) plt.ylabel("count")
     Text(0, 0.5, 'count')
```



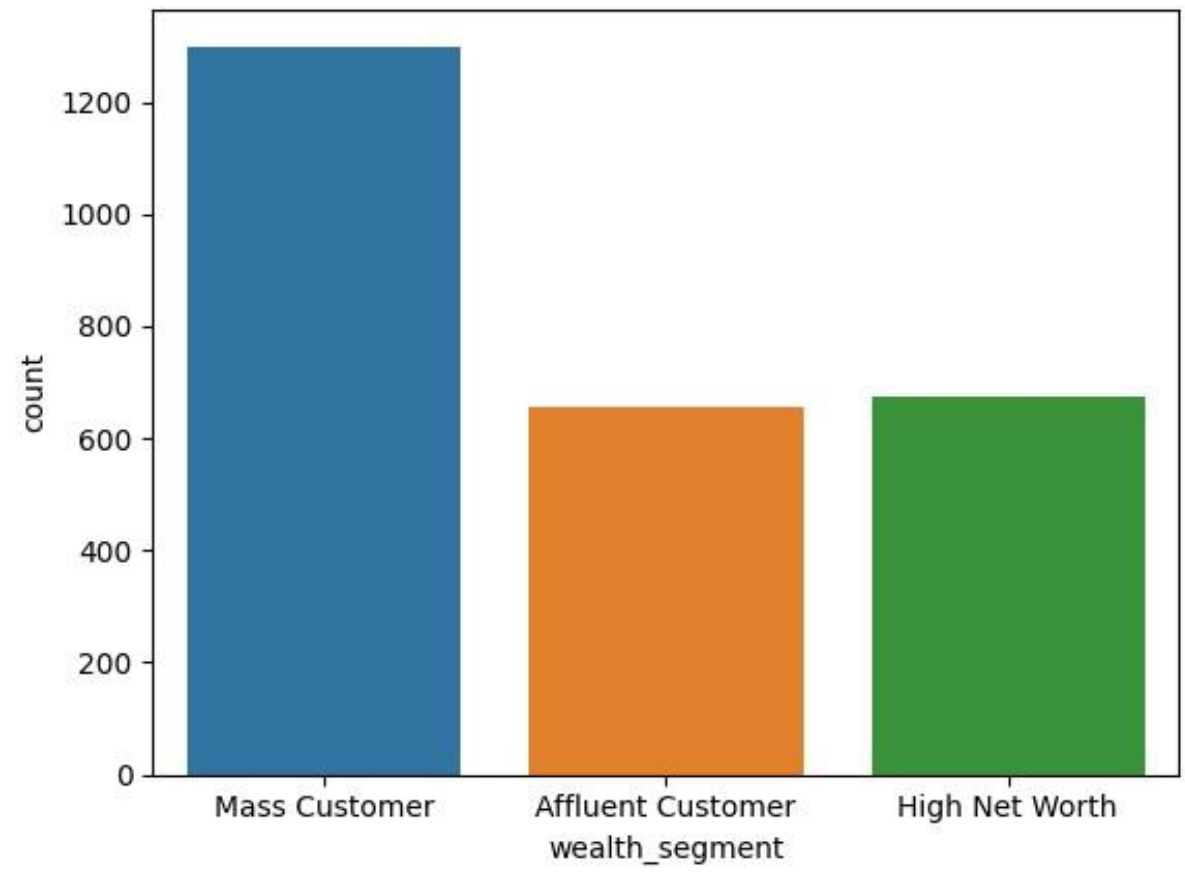
```
# This Command count how many values is in "wealth_segment" column

cdg.wealth_segment.value_counts()
```

```
Mass Customer        1300
High Net Worth        674
Affluent Customer     656
Name: wealth_segment, dtype: int64
```

```
# count plot of "wealth_segment" column

sns.countplot(x='wealth_segment', data =cdg)
plt.xlabel('wealth_segment')
plt.ylabel('count') plt.show()
```



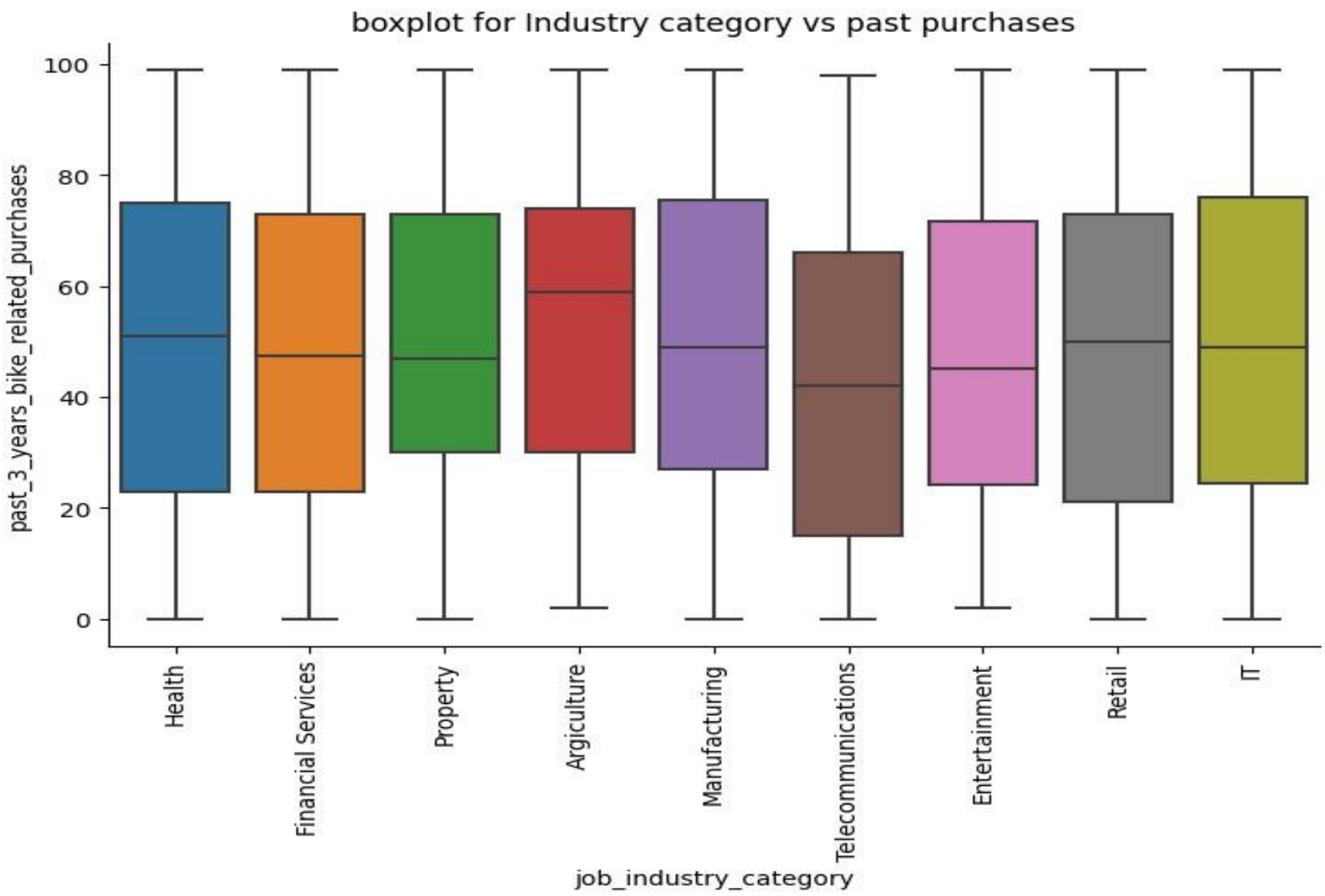

```
#This command gives How many values in "job_industry_category" column

cdg.job_industry_category.value_counts()
```

```
Manufacturing         635
Financial Services    626
Health                496
Retail                278
Property              222
IT                    119
Entertainment         110
Argiculture            91
Telecommunications     53
Name: job_industry_category, dtype: int64
```

```
# This command gives box plot between "job_industry_category and past_purchasers" Column

sns.catplot(x= "job_industry_category", y ="past_3_years_bike_related_purchases", data = cdg, kind = "box",
aspect = 1.5)
plt.xticks(rotation = 'vertical')
plt.title("boxplot for Industry category vs past purchases")
plt.show()
```



0s completed at 10:11 PM