matplot-1

August 12, 2024

Data visualization = to represent data graphically

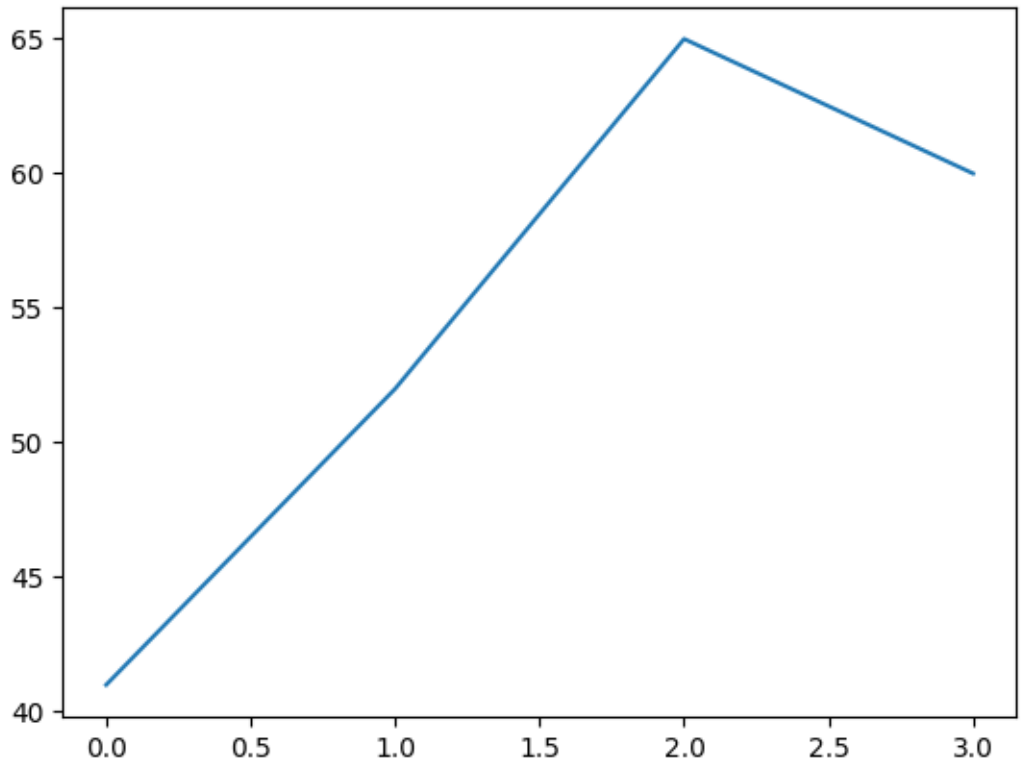
[2]:
```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

1 Linechart

When data of both axis is numerical there use Linechart

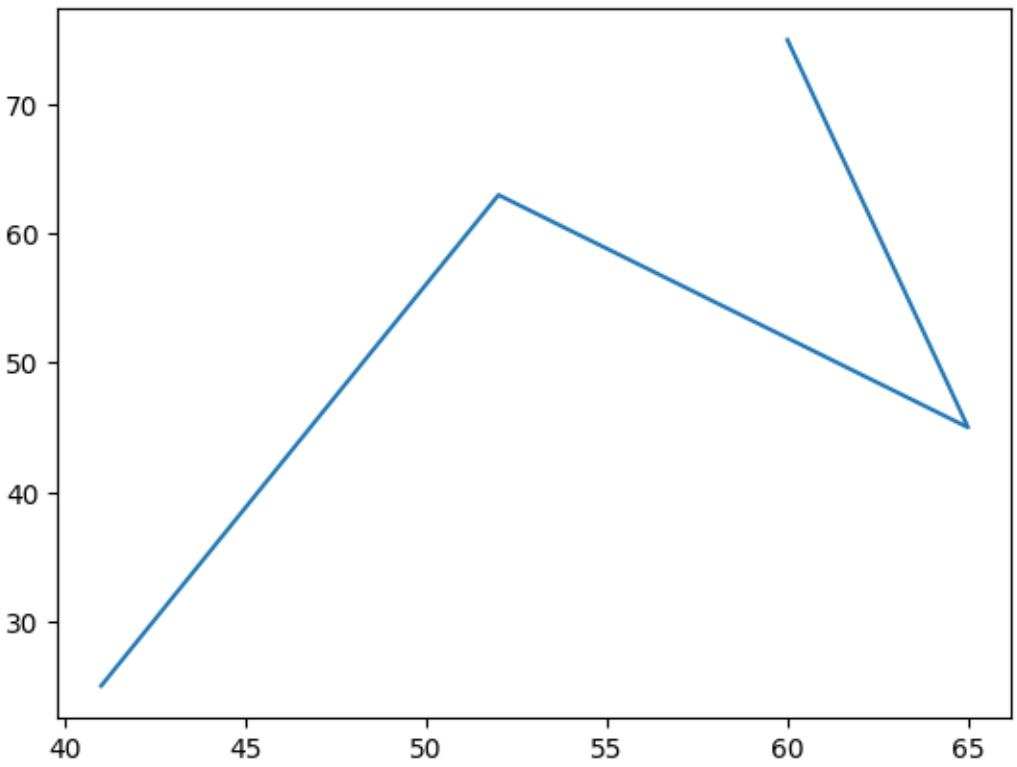
[3]:
```
ls=[41,52,65,60]
plt.plot(ls)
```

[3]:
```
[<matplotlib.lines.Line2D at 0x23af50ad9d0>]
```



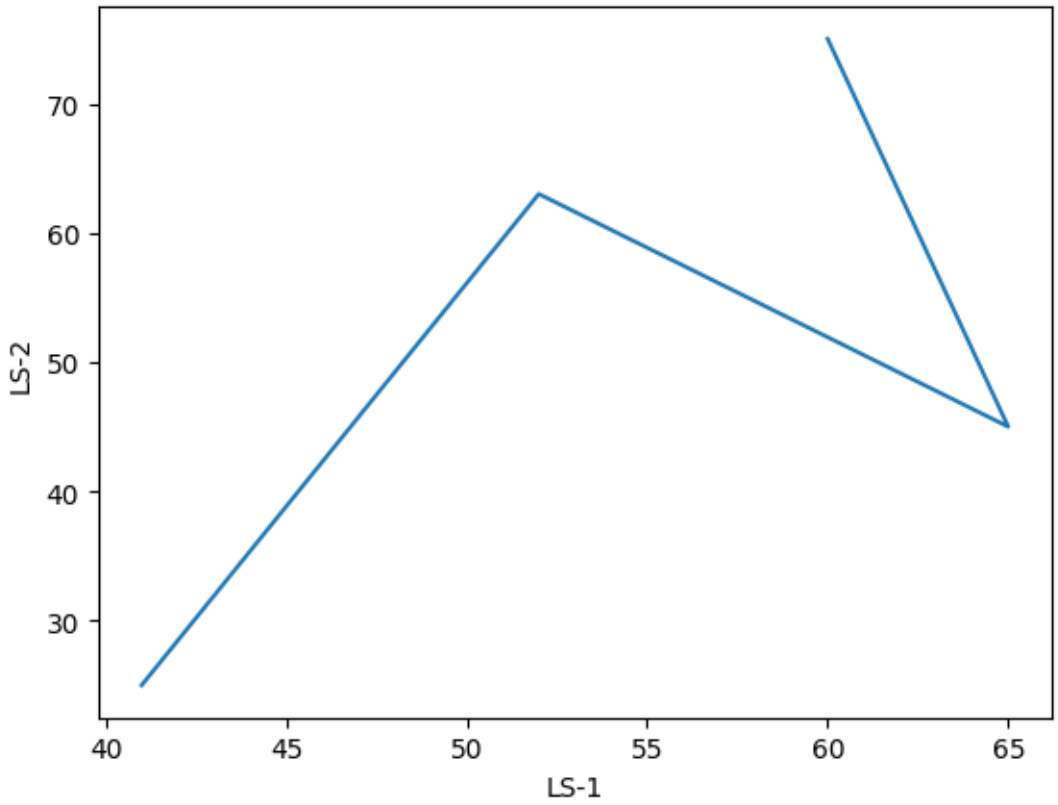
[4]:
```
ls1=[41,52,65,60]
ls2=[25,63,45,75]
plt.plot(ls,ls2)
```

[4]: [<matplotlib.lines.Line2D at 0x23af52d8050>]



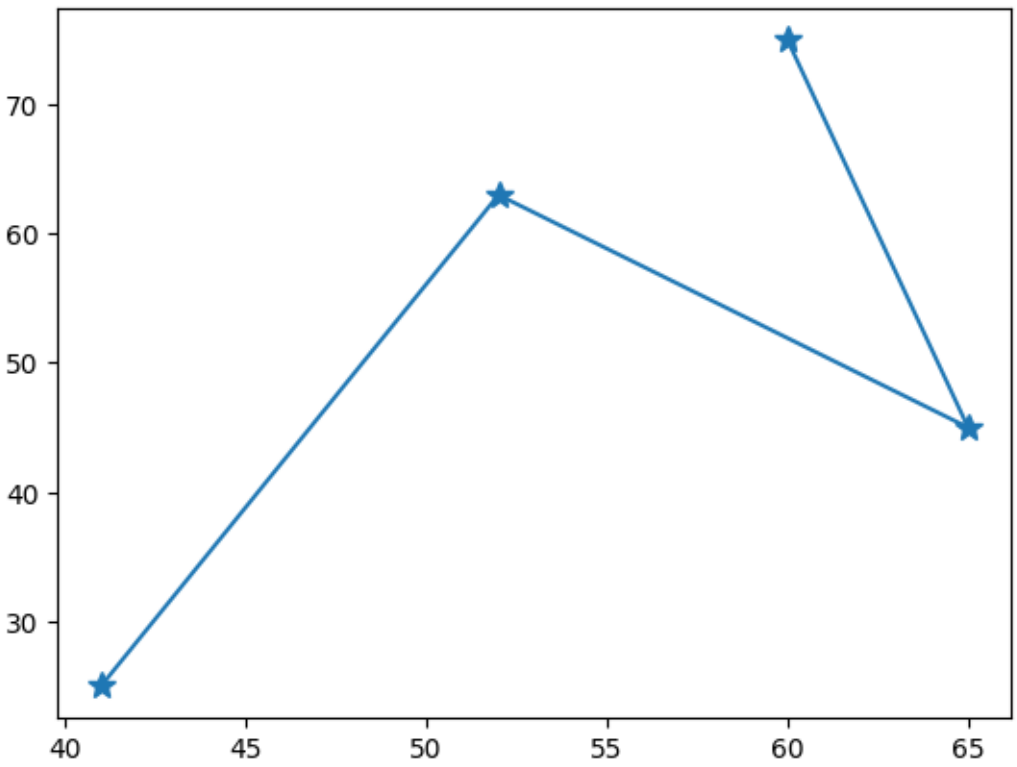
[5]:
```
ls1=[41,52,65,60]
ls2=[25,63,45,75]
plt.plot(ls,ls2)
plt.xlabel("LS-1")
plt.ylabel("LS-2")
```

[5]: Text(0, 0.5, 'LS-2')



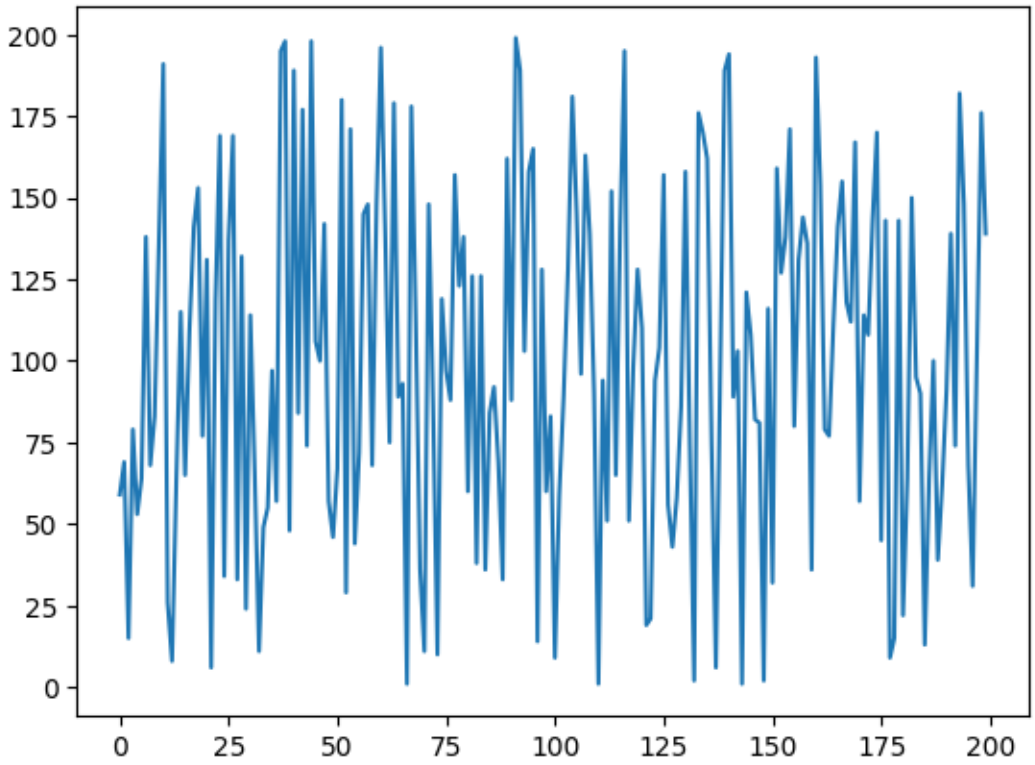
[6]:
```
ls1=[41,52,65,60]
ls2=[25,63,45,75]
plt.plot(ls1,ls2,marker ="*",ms=10)
```

[6]: [<matplotlib.lines.Line2D at 0x23af74e01d0>]



[7]:
```
arr=np.random.randint(1,200,200)
plt.plot(arr)
```

[7]: [<matplotlib.lines.Line2D at 0x23af7497110>]



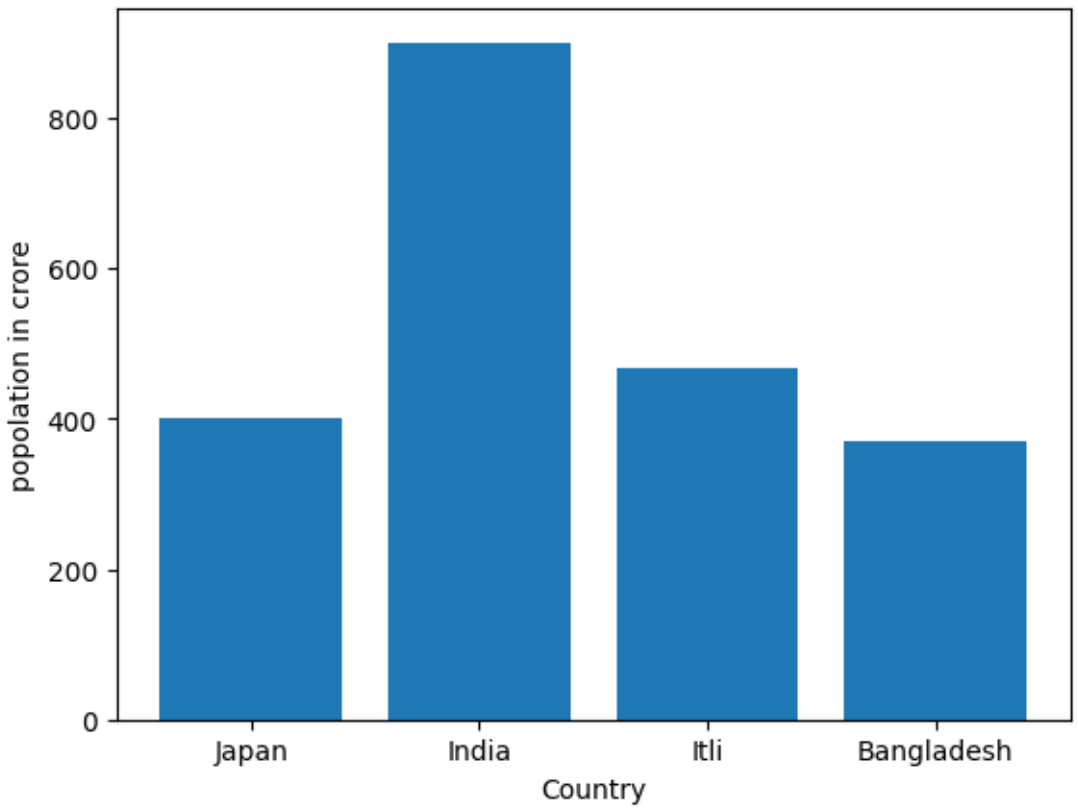
2 Bar plot

If one data is categorical(alphabet) and second is numerical then use Bar plot

[8]:
```
country_name=["Japan","India","Itli","Bangladesh"]
country_population=[400,900,466,370]

plt.xlabel("Country")
plt.ylabel("popolation in crore")
plt.bar(country_name,country_population)
```

[8]:
```
<BarContainer object of 4 artists>
```



Working on real data

[9]:
```
df=pd.read_csv("Used_Bikes.csv")
df.head()
```

[9]:

	bike_name	price	city	kms_driven	owner	age	power	brand
0	TVS Star City Plus Dual Tone 110cc	35000.0	Ahmedabad	17654.0	First Owner	3.0	110.0	TVS
1	Royal Enfield Classic 350cc	119900.0	Delhi	11000.0	First Owner	4.0	350.0	Royal Enfield
2	Triumph Daytona 675R	600000.0	Delhi	110.0	First Owner	8.0	675.0	Triumph
3	TVS Apache RTR 180cc	65000.0	Bangalore	16329.0	First Owner	4.0	180.0	TVS
4	Yamaha FZ S V 2.0 150cc-Ltd. Edition	80000.0	Bangalore	10000.0	First Owner	3.0	150.0	Yamaha

[10]:
```
df.drop_duplicates(inplace=True)
```

[11]:
```
df["owner"].value_counts()
```

[11]:
```
owner
First Owner             6642
Second Owner             588
Third Owner               84
Fourth Owner Or More      10
Name: count, dtype: int64
```

[12]:
```
df["owner"].value_counts().keys()
```

[12]:
```
Index(['First Owner', 'Second Owner', 'Third Owner', 'Fourth Owner Or More'],
dtype='object', name='owner')
```

[13]:
```
ownership=(df["owner"].value_counts().keys())
ownership
```

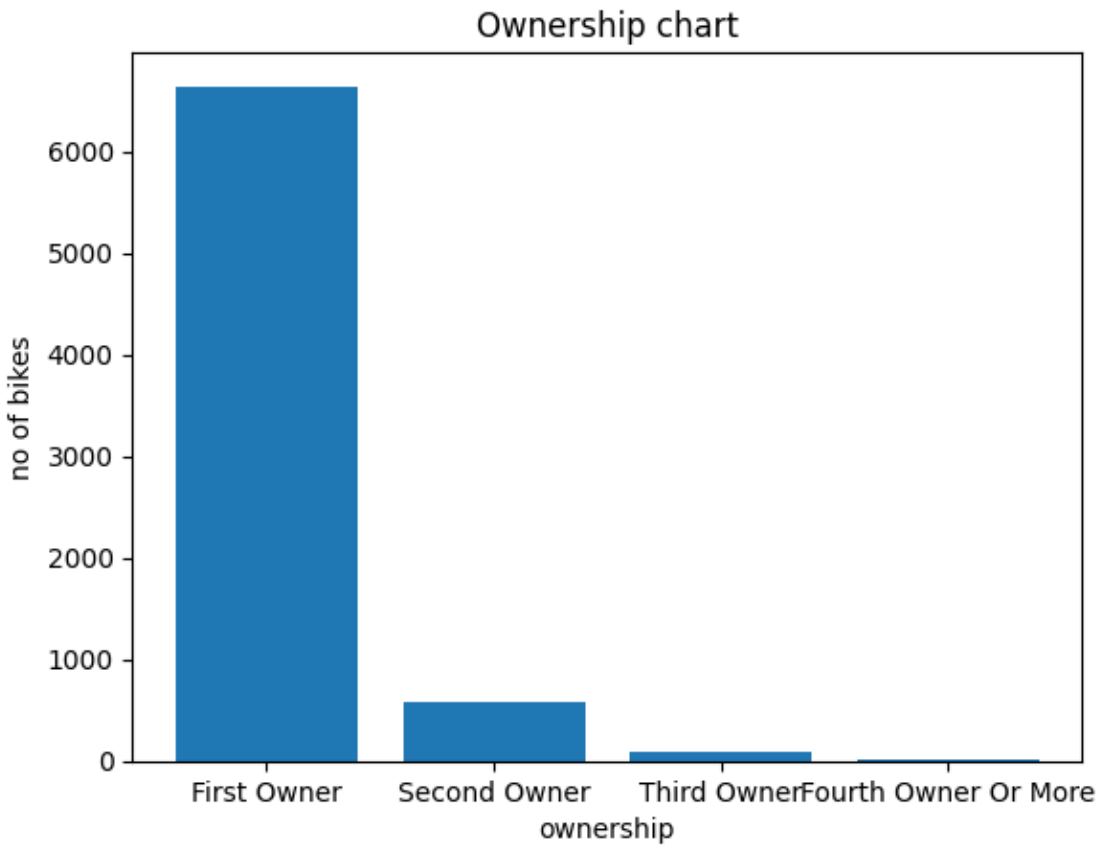
[13]:
```
Index(['First Owner', 'Second Owner', 'Third Owner', 'Fourth Owner Or More'],
dtype='object', name='owner')
```

[14]:
```
no_of_bikes=list(df["owner"].value_counts().values)
no_of_bikes
```

[14]:
```
[np.int64(6642), np.int64(588), np.int64(84), np.int64(10)]
```

[15]:
```
plt.bar(ownership,no_of_bikes)
plt.xlabel("ownership")
plt.ylabel("no of bikes")
plt.title("Ownership chart")
```

[15]:
```
Text(0.5, 1.0, 'Ownership chart')
```



[16]:
```
df["brand"].value_counts()
```

[16]:
```
brand
Bajaj                2081
Royal Enfield        1346
Hero                 1142
Honda                 676
Yamaha                651
TVS                   481
KTM                   375
Suzuki                203
Harley-Davidson        91
Kawasaki               61
Hyosung                53
Mahindra               50
Benelli                46
Triumph                21
Ducati                 20
BMW                    10
Jawa                    7
```

```
Indian                3
MV                    3
Rajdoot               1
LML                   1
Yezdi                 1
Ideal                 1
Name: count, dtype: int64
```

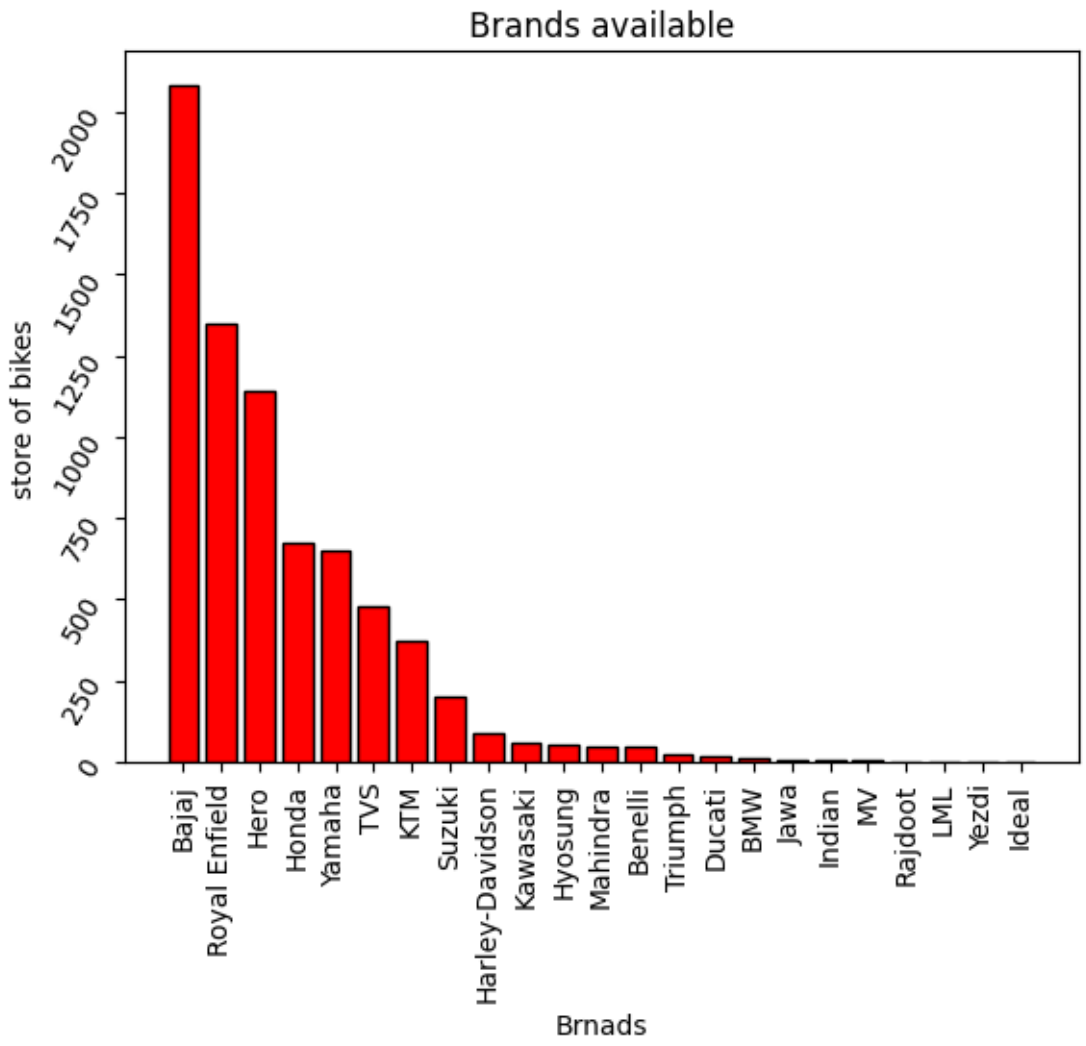
[17]:
```
brand_names=list(df["brand"].value_counts().keys())
brand_names
```

[17]:
```
['Bajaj',
 'Royal Enfield',
 'Hero',
 'Honda',
 'Yamaha',
 'TVS',
 'KTM',
 'Suzuki',
 'Harley-Davidson',
 'Kawasaki',
 'Hyosung',
 'Mahindra',
 'Benelli',
 'Triumph',
 'Ducati',
 'BMW',
 'Jawa',
 'Indian',
 'MV',
 'Rajdoot',
 'LML',
 'Yezdi',
 'Ideal']
```

[21]:
```
brand_count=list(df["brand"].value_counts().values)
```

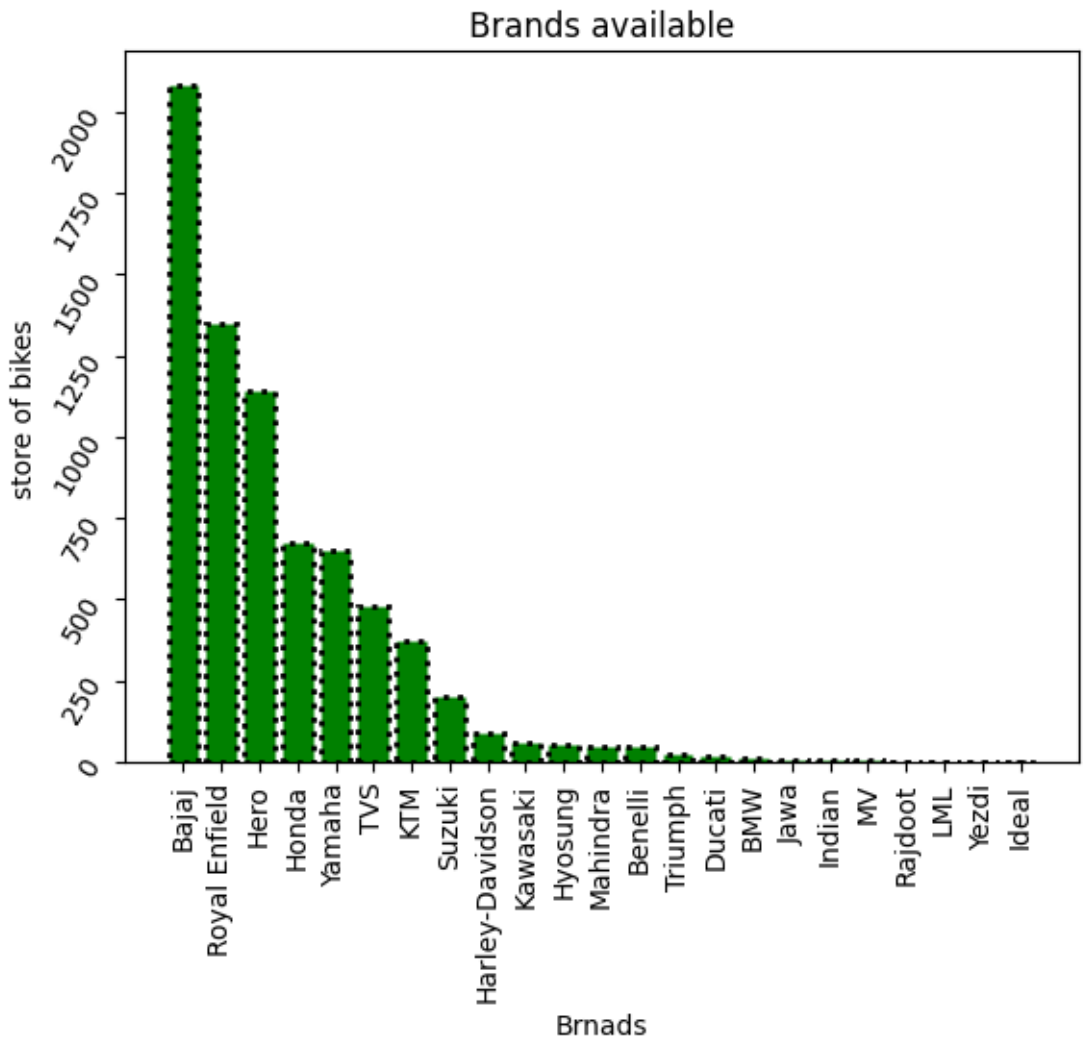
[22]:
```
plt.bar(brand_names,brand_count,color="red",ec="k")    # ec = edge color  and␣
 ↪b=blue,k = black  , lw = line width
plt.xlabel("Brnads")
plt.ylabel("store of bikes")
plt.xticks(rotation=90)
plt.yticks(rotation=60)
plt.title("Brands available")
```

[22]:
```
Text(0.5, 1.0, 'Brands available')
```



[ ]:
```
plt.bar(brand_names,brand_count,color="green",ec="k",lw=2,ls="dotted")    # ec =␣
 ↪edge color  and b=blue,k = black  , lw = line width, ls= line style
plt.xlabel("Brnads")
plt.ylabel("store of bikes")
plt.xticks(rotation=90)
plt.yticks(rotation=60)
plt.title("Brands available")
```

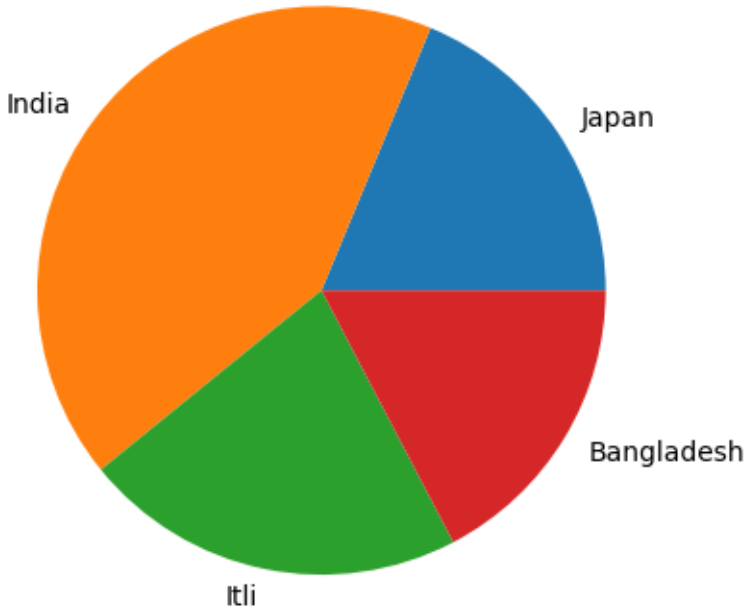
[ ]:
```
Text(0.5, 1.0, 'Brands available')
```



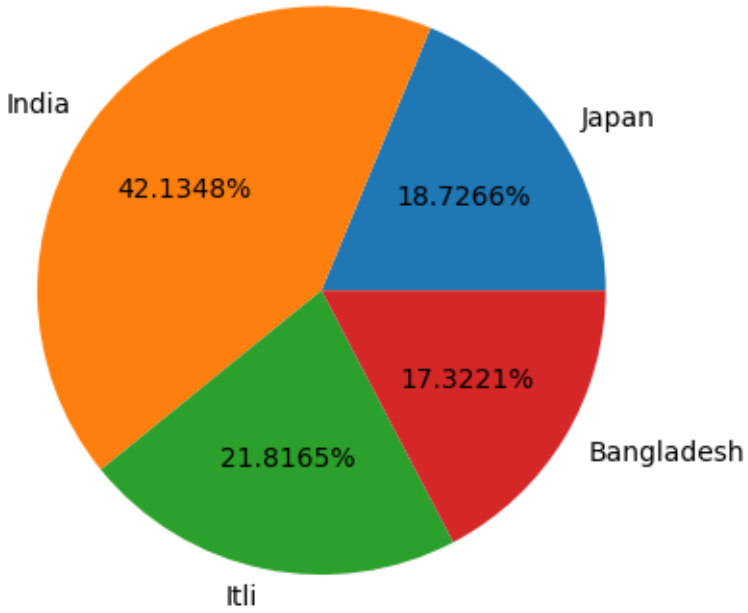
3 pie plot

[ ]:
```
print(country_name)
print(country_population)
plt.pie(country_population,labels=country_name)
plt.show()
```

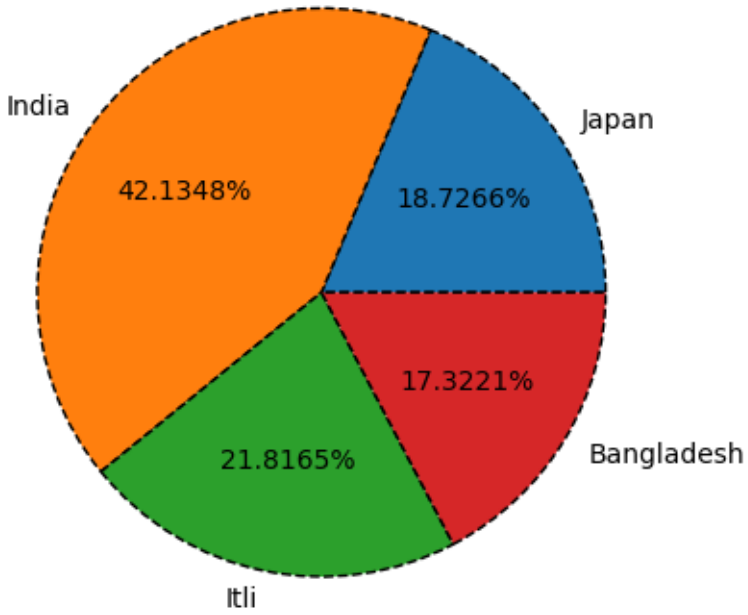
```
['Japan', 'India', 'Itli', 'Bangladesh']
[400, 900, 466, 370]
```



[ ]:
```
plt.pie(country_population,labels=country_name,autopct="%2.4f%%")  #autopct=
↪autopercentage
plt.show()
```

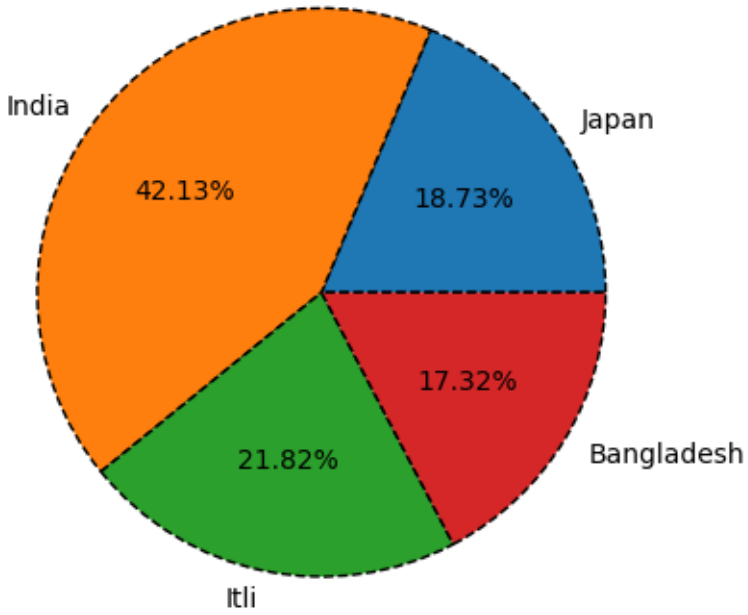


[ ]:
```
plt.pie(country_population,labels=country_name,autopct="%2.
  ↪4f%%",wedgeprops={'ec':"black","ls":"dashed","lw":1})  #autopct=␣
  ↪autopercentage
plt.figure(figsize=(8,8))
plt.show()
```

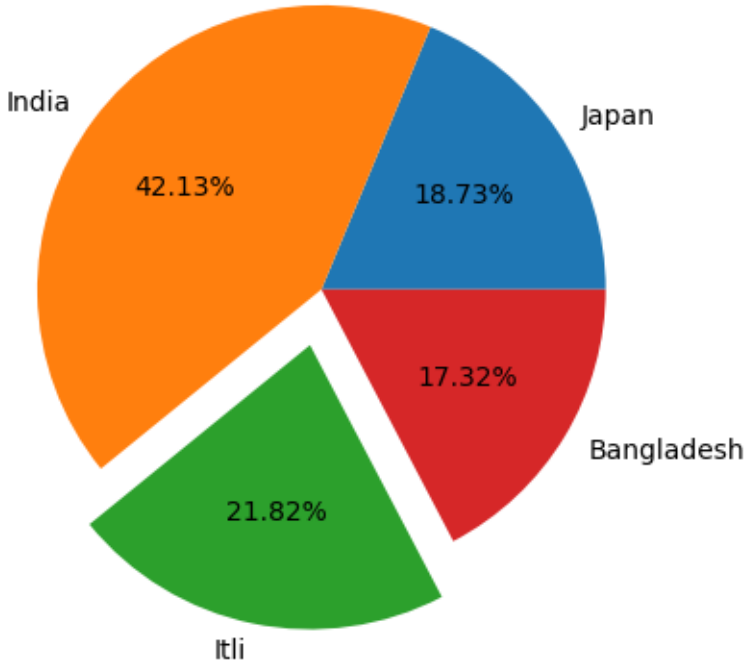



```
<Figure size 800x800 with 0 Axes>
```

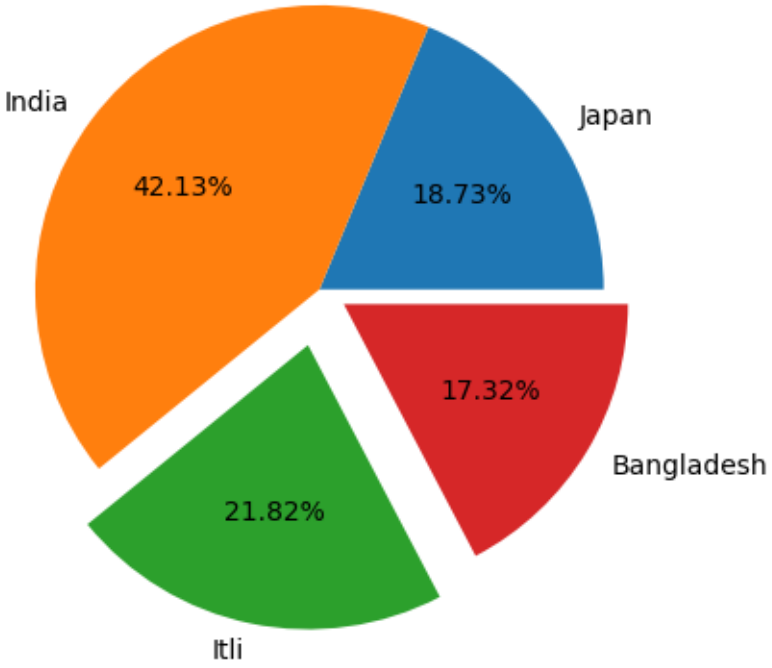
[ ]:
```
explosion=[0,0,0,0]
plt.pie(country_population,labels=country_name,autopct="%2.
  ↪2f%%",wedgeprops={'ec':"black",'ls':'dashed','linewidth':
  ↪1},explode=explosion)
plt.show()
```



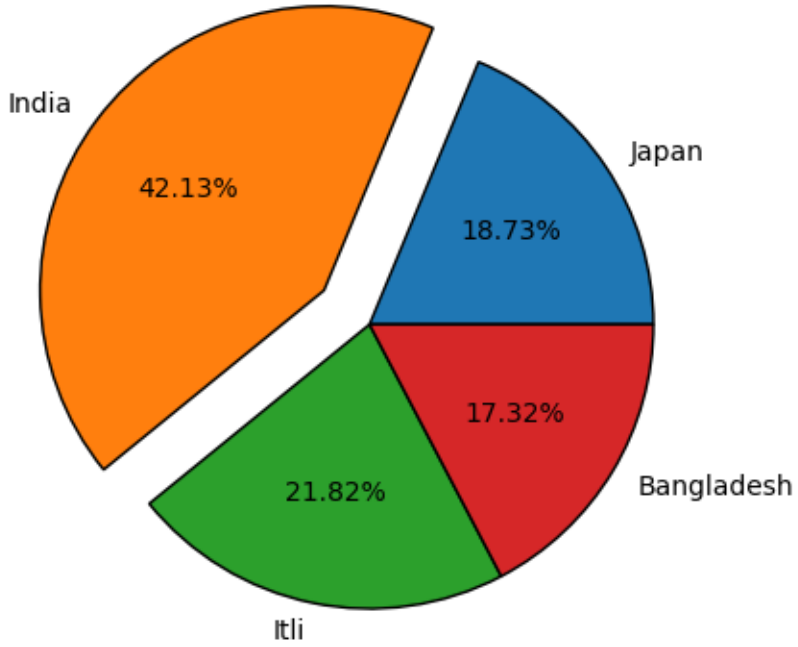
[ ]:
```
explosion=[0,0,0.2,0]
plt.pie(country_population,labels=country_name,autopct="%2.
  ↪2f%%",explode=explosion)
plt.show()
```



[ ]:
```
explosion=[0,0,0.2,0.1]
plt.pie(country_population,labels=country_name,autopct="%2.
  ↪2f%%",explode=explosion)
plt.show()
```

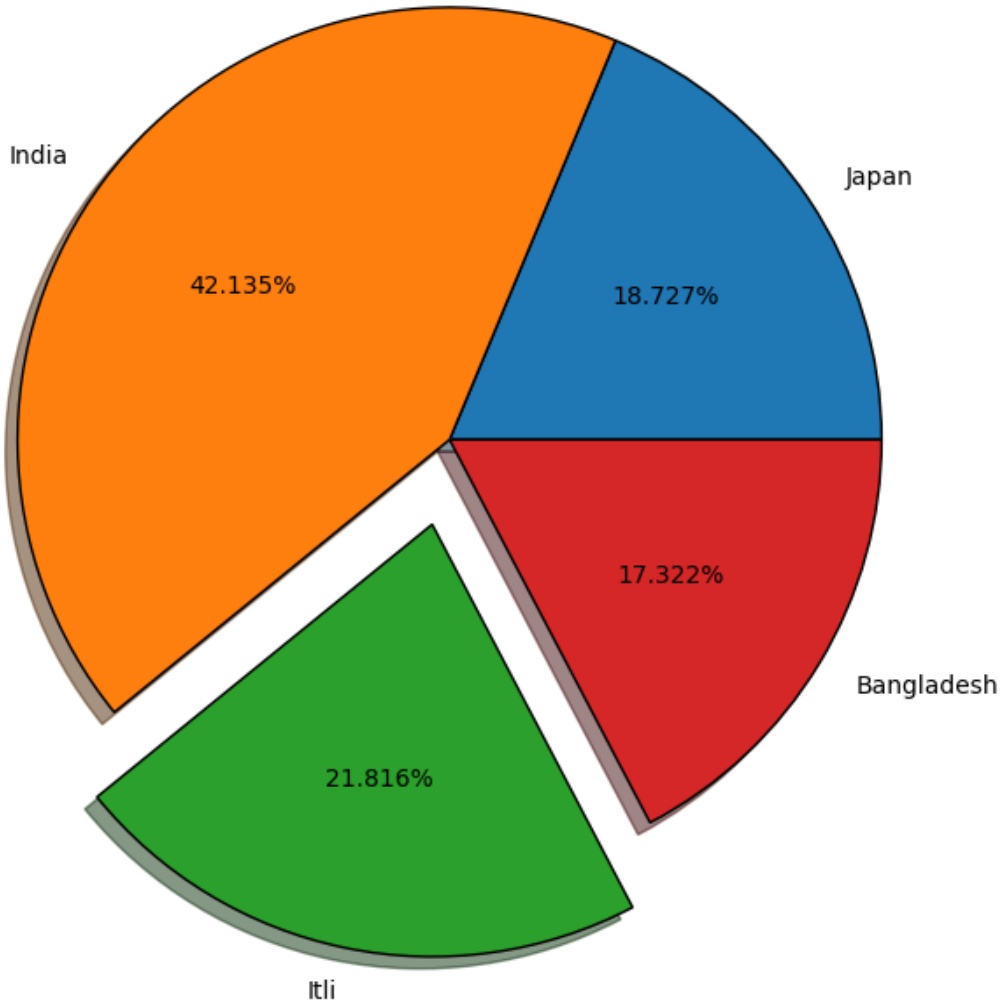


[ ]:
```
explosion=[0,.2,0,0]
plt.pie(country_population,labels=country_name,autopct="%2.
  ↪2f%%",wedgeprops={"ec":"black"},explode=explosion)
plt.show()
```



[ ]:
```
print(country_name)
print(country_population)
explosion=[0,0,0.2,0]
print(explosion)
plt.figure(figsize=(8,8))
plt.pie(country_population,labels=country_name,autopct="%2.
  ↪3f%%",wedgeprops={"ec":"black"},explode=explosion,shadow="green")
plt.savefig("country_data")
plt.show()
```

```
['Japan', 'India', 'Itli', 'Bangladesh']
[400, 900, 466, 370]
[0, 0, 0.2, 0]
```


India
42.135%
Japan
18.727%
17.322%
Bangladesh
21.816%
Itli

matplot-2

August 12, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```
df=pd.read_csv("Used_Bikes.csv")
df.head()
```

[2]:

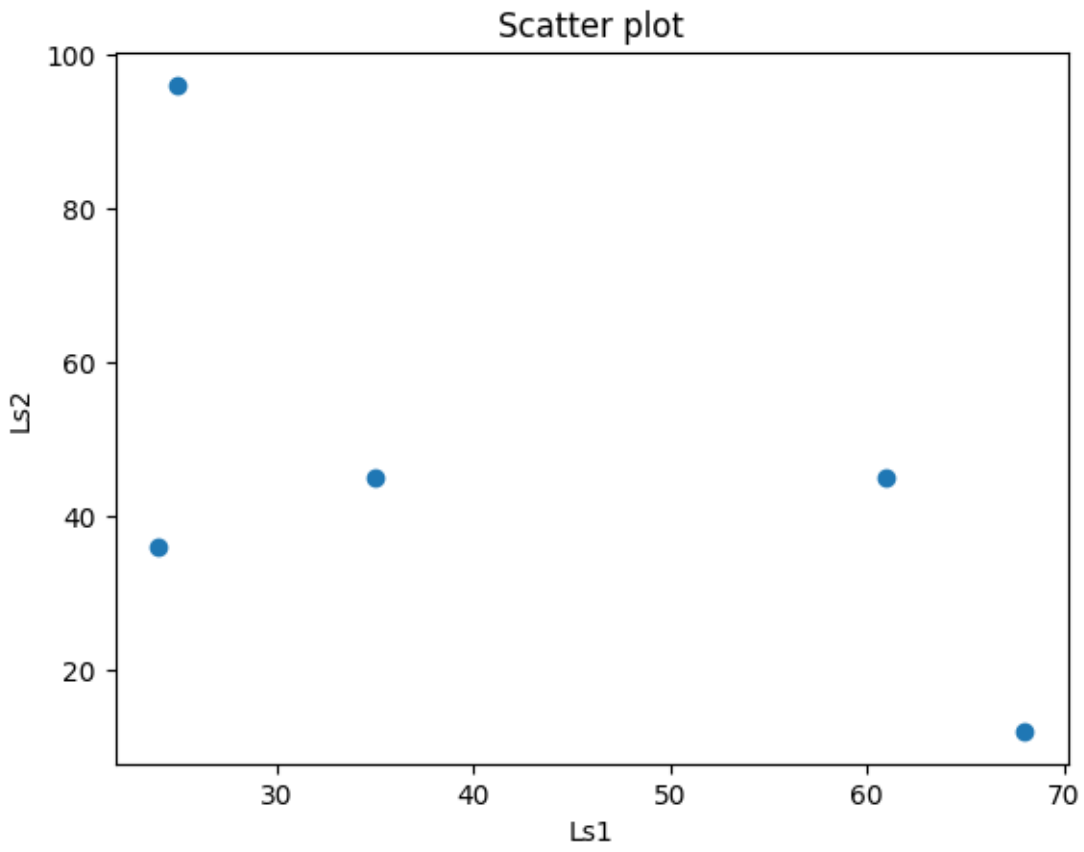
	bike_name	price	city	kms_driven	owner	age	power	brand
0	TVS Star City Plus Dual Tone 110cc	35000.0	Ahmedabad	17654.0	First Owner	3.0	110.0	TVS
1	Royal Enfield Classic 350cc	119900.0	Delhi	11000.0	First Owner	4.0	350.0	Royal Enfield
2	Triumph Daytona 675R	600000.0	Delhi	110.0	First Owner	8.0	675.0	Triumph
3	TVS Apache RTR 180cc	65000.0	Bangalore	16329.0	First Owner	4.0	180.0	TVS
4	Yamaha FZ S V 2.0 150cc-Ltd. Edition	80000.0	Bangalore	10000.0	First Owner	3.0	150.0	Yamaha

0.1 scatter plot

it shows thebullet points . a single point represent as (x,y)

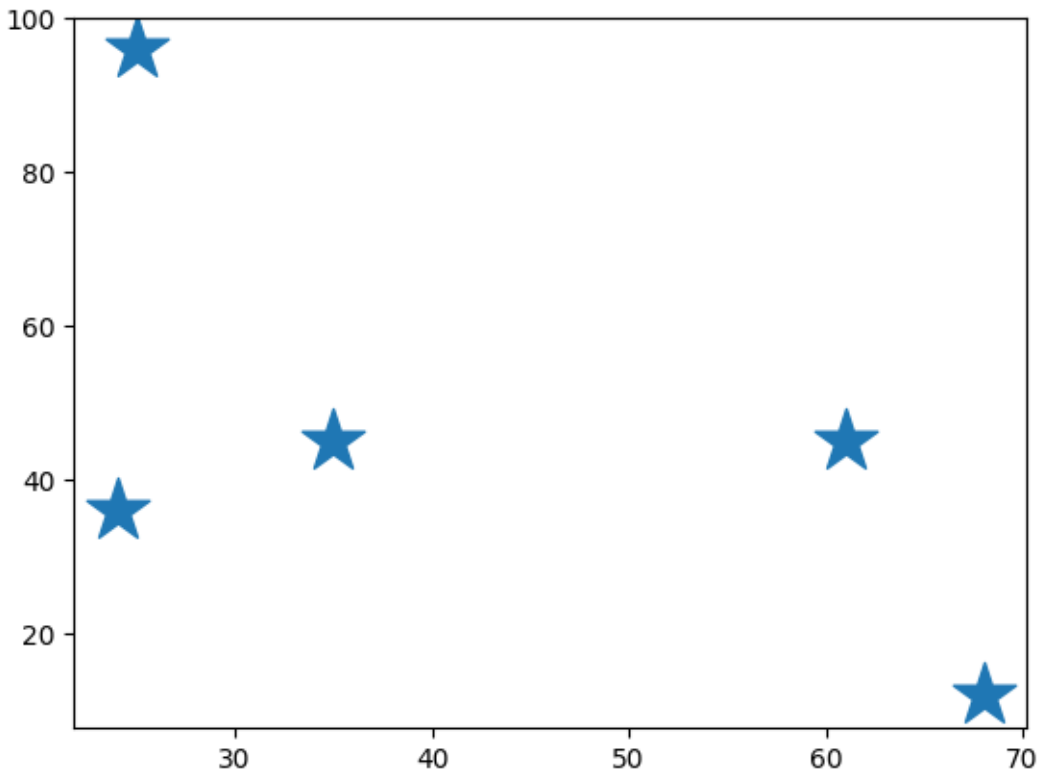
[3]:
```
ls1=[25,35,68,24,61]
ls2=[96,45,12,36,45]
plt.scatter(ls1,ls2)
plt.xlabel("Ls1")
plt.ylabel("Ls2")
plt.title("Scatter plot")
plt.show
```

[3]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



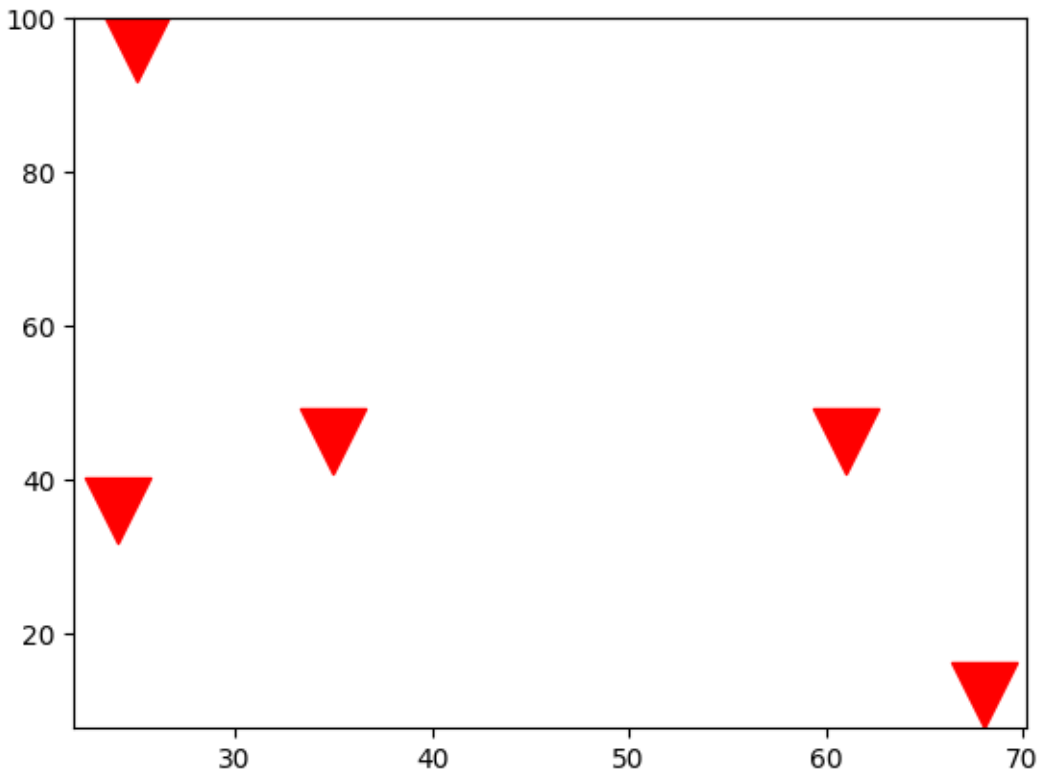
[4]:
```
plt.scatter(ls1,ls2,marker="*",s=600)  # s for size of marker
```

[4]: <matplotlib.collections.PathCollection at 0x162038027e0>



[5]:
```
plt.scatter(ls1,ls2,marker="v",color="red",s=600)
plt.show
```

[5]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



[6]:
```
df
```

[6]:

	bike_name	price	city	kms_driven	owner	age	power	brand
0	TVS Star City Plus Dual Tone 110cc	35000.0	Ahmedabad	17654.0	First Owner	3.0	110.0	TVS
1	Royal Enfield Classic 350cc	119900.0	Delhi	11000.0	First Owner	4.0	350.0	Royal Enfield
2	Triumph Daytona 675R	600000.0	Delhi	110.0	First Owner	8.0	675.0	Triumph
3	TVS Apache RTR 180cc	65000.0	Bangalore	16329.0	First Owner	4.0	180.0	TVS
4	Yamaha FZ S V 2.0 150cc-Ltd. Edition	80000.0	Bangalore	10000.0	First Owner	3.0	150.0	Yamaha
…	…	…	…	…	…	…	…	…
32643	Hero Passion Pro 100cc	39000.0	Delhi	22000.0				
32644	TVS Apache RTR 180cc	30000.0	Karnal	6639.0				
32645	Bajaj Avenger Street 220	60000.0	Delhi	20373.0				
32646	Hero Super Splendor 125cc	15600.0	Jaipur	84186.0				
32647	Bajaj Pulsar 150cc	22000.0	Pune	60857.0				

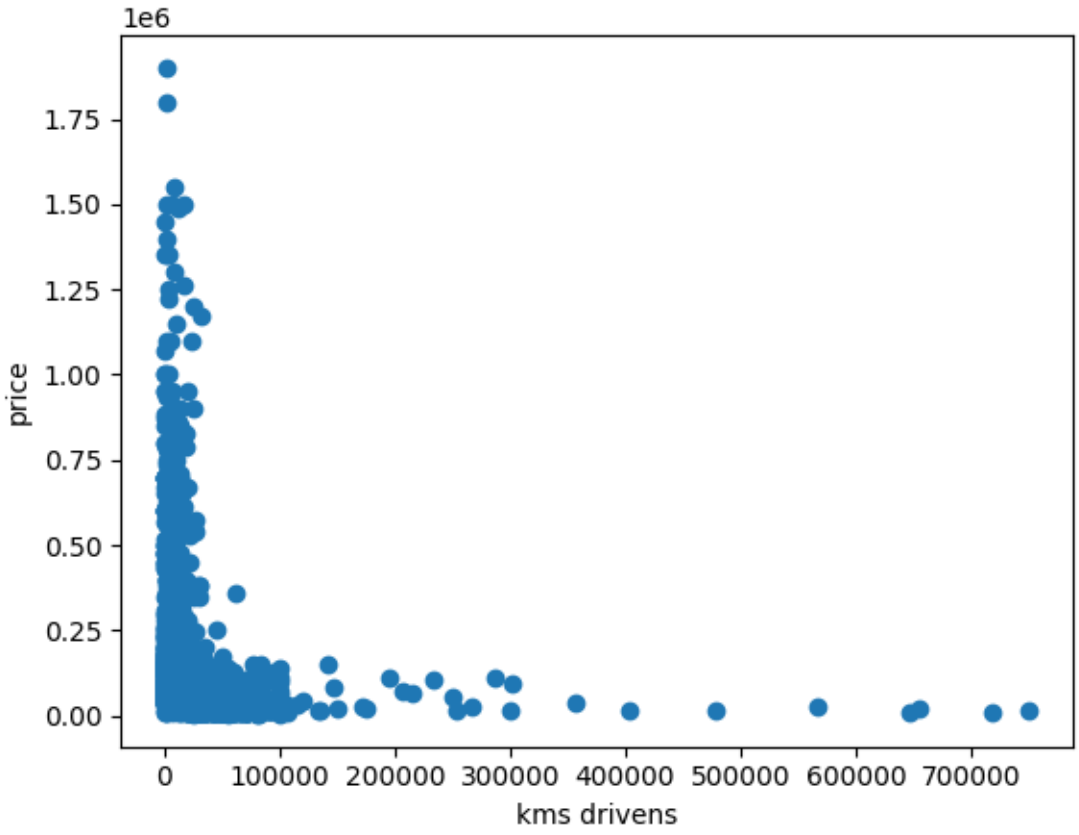
```
32643  First Owner   4.0  100.0          Hero
32644  First Owner   9.0  180.0           TVS
32645  First Owner   6.0  220.0         Bajaj
32646  First Owner  16.0  125.0          Hero
32647  First Owner  13.0  150.0         Bajaj

[32648 rows x 8 columns]
```

[7]:
```
df.drop_duplicates(inplace=True)
df["price"]
```

[7]:
```
0        35000.0
1       119900.0
2       600000.0
3        65000.0
4        80000.0
          …
9362     25000.0
9369     35000.0
9370    450000.0
9371    139000.0
9372     80000.0
Name: price, Length: 7324, dtype: float64
```

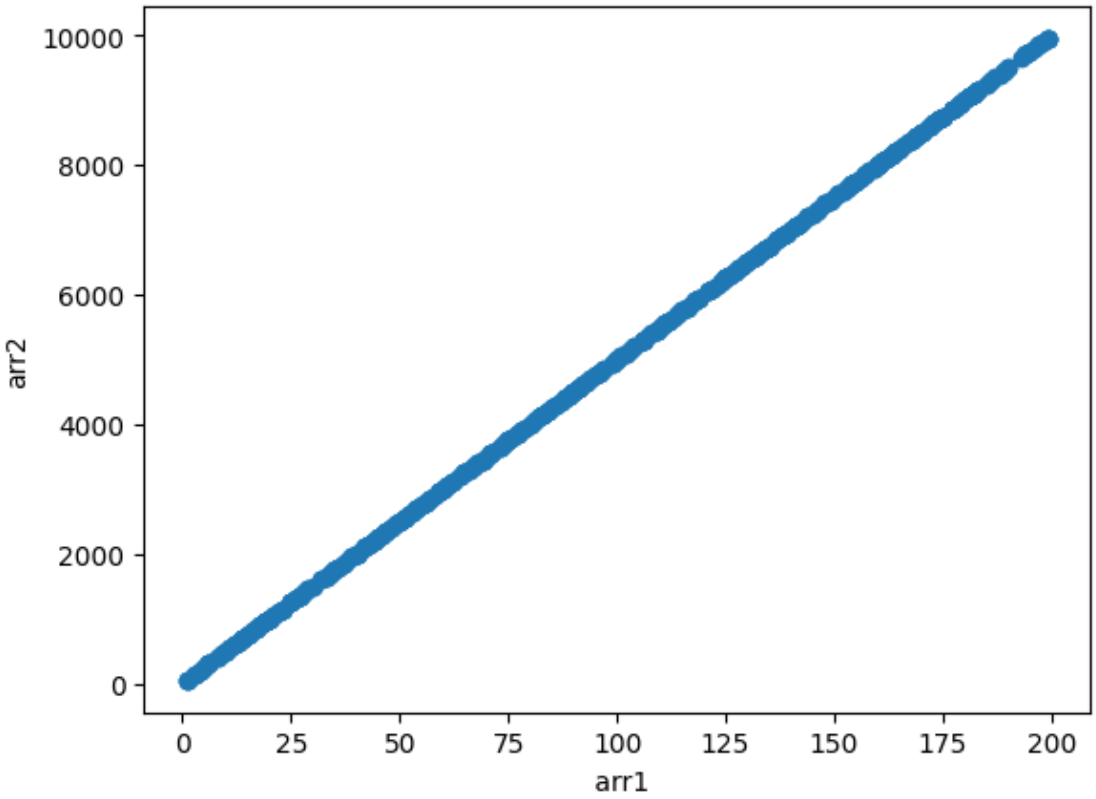
[8]:
```
plt.scatter(x=df["kms_driven"],y=df["price"])
plt.xlabel("kms drivens")
plt.ylabel("price")
plt.show()
```



also used to check density is +ve or -ve

[9]:
```
arr=np.random.randint(1,200,500)
arr2=arr*50
```

[10]:
```
plt.scatter(arr,arr2)
plt.xlabel(" arr1")
plt.ylabel("arr2")
plt.show()
```

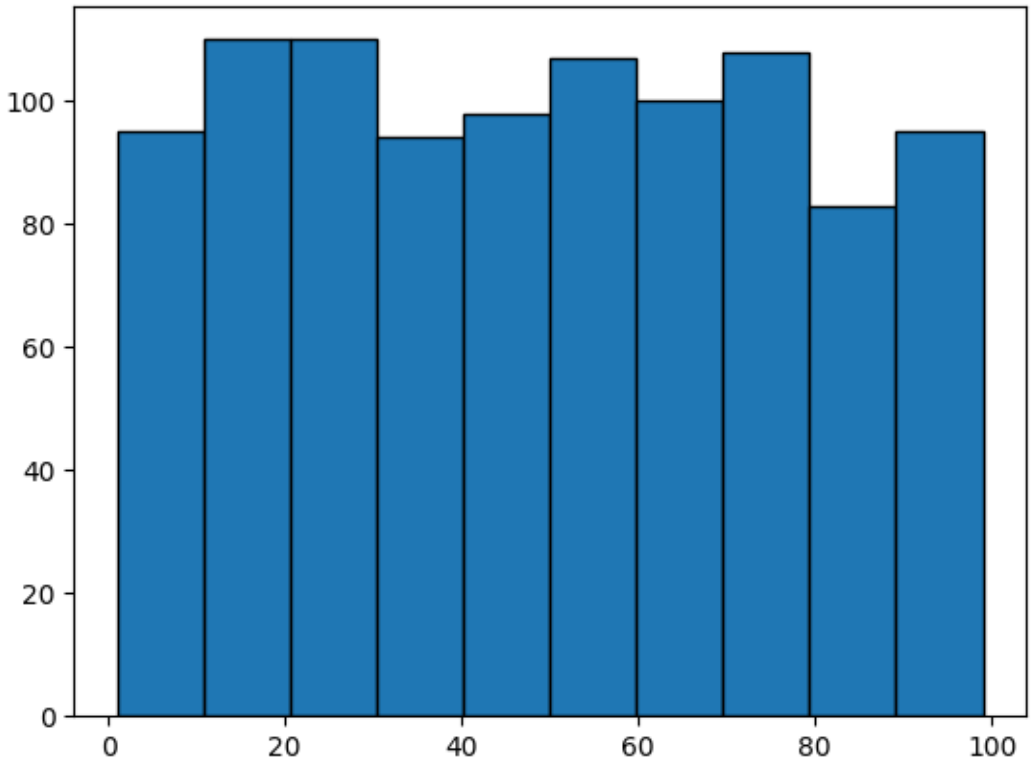


0.2 HISOGRAM

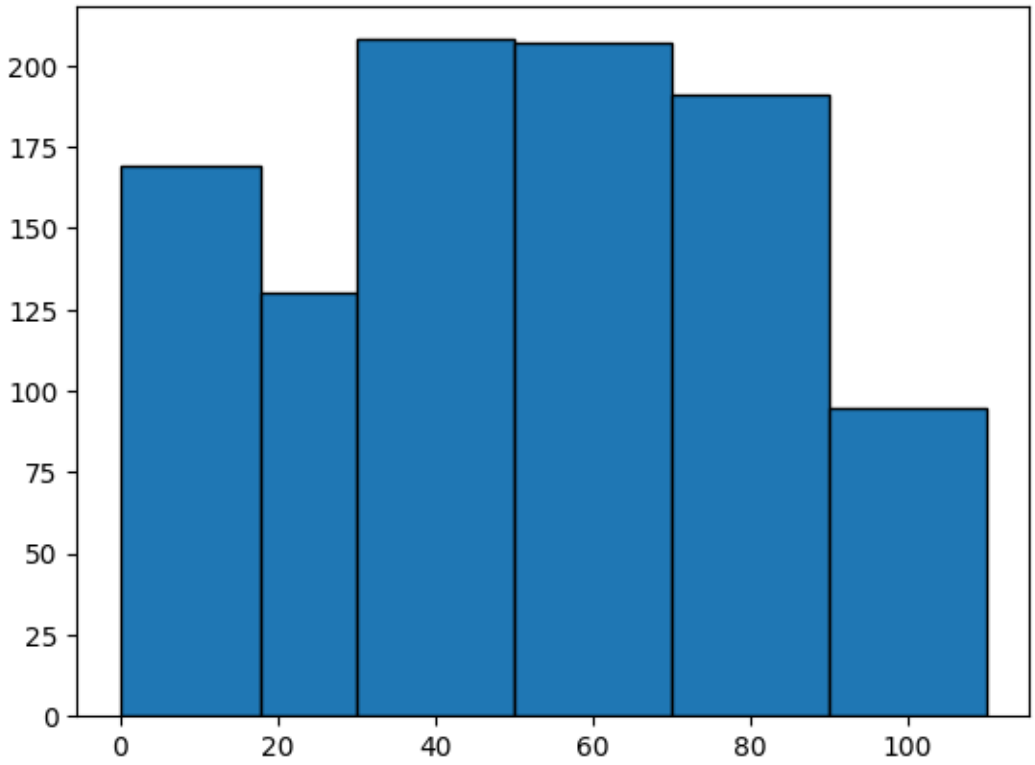
It gives information about the distribution of data

[23]:
```
age=np.random.randint(1,100,1000)
```

[24]:
```
plt.hist(age,ec="k")
plt.show()
```



[25]:
```
boxex_dist=[0,18,30,50,70,90,110]        # By using this bins we can  create a␣
 ↪range like here 0-18,18-30,30-50,50-70,70-90,90-110
plt.hist(age,bins=boxex_dist,ec="k")
plt.show()
```



[28]:
```
df["price"].value_counts().sum()
```

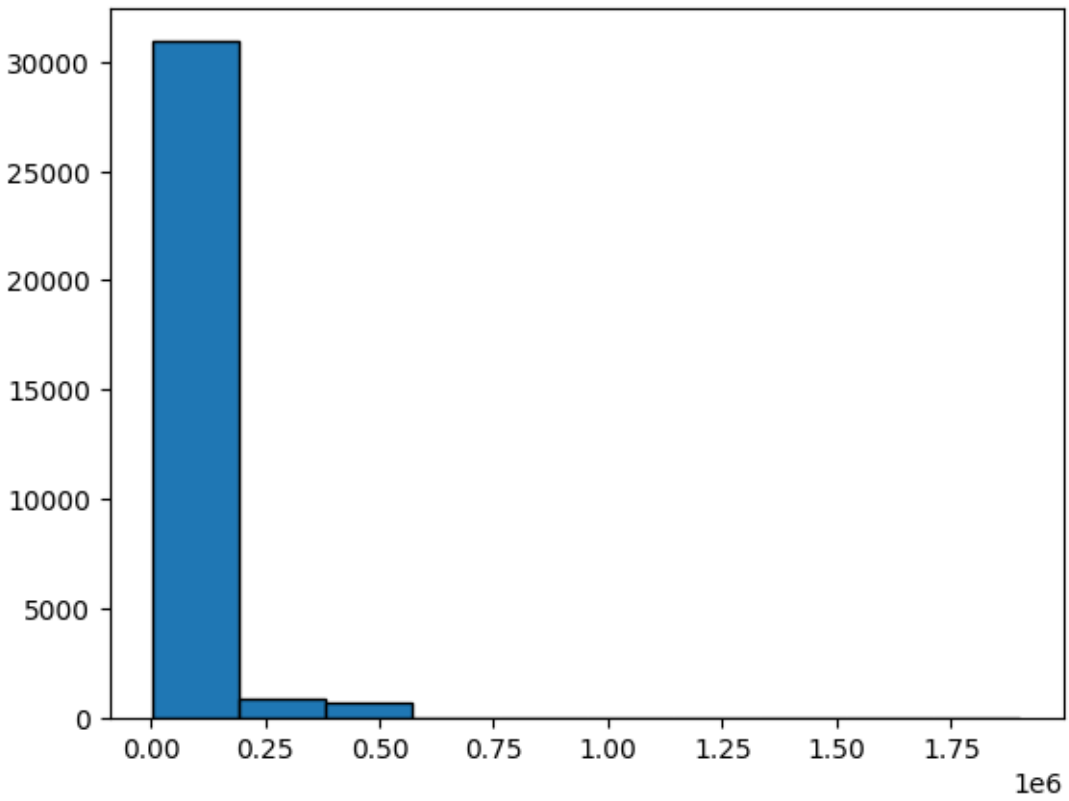
[28]:
```
np.int64(32648)
```

[32]:
```
print(df["price"].max())
print(df["price"].min())
```

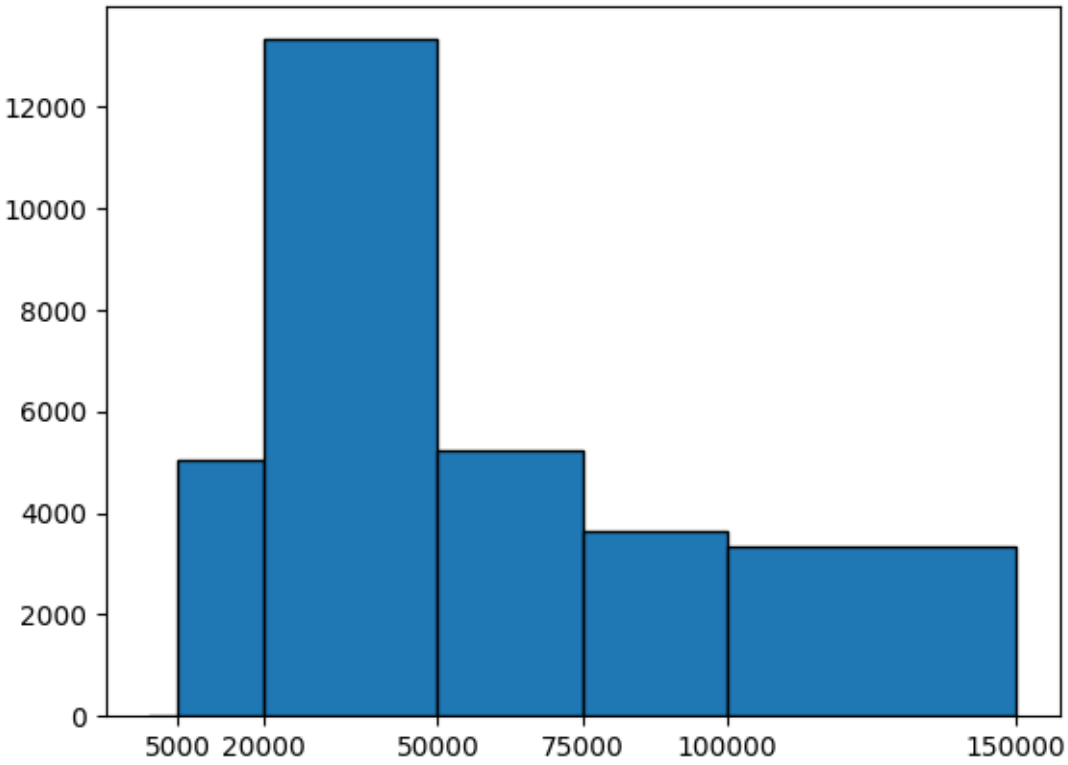
```
1900000.0
4400.0
```

[27]:
```
plt.hist(df["price"],ec="k")
plt.show
```

[27]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

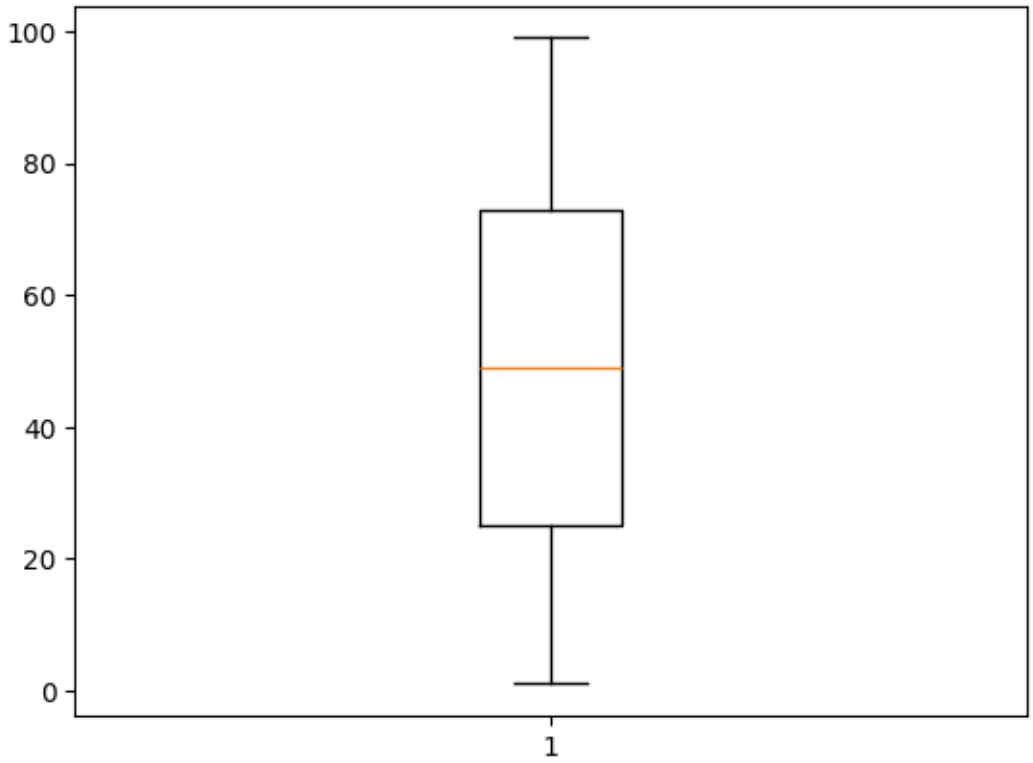


[62]:
```
intervels=[0,5000,20000,50000,75000,100000,150000]
plt.hist(df["price"],bins=intervels,ec="k")
plt.xticks([5000,20000,50000,75000,100000,150000])
plt.show()
```



0.3 BOX chart

[65]:
```
plt.boxplot(age)
plt.show()
```

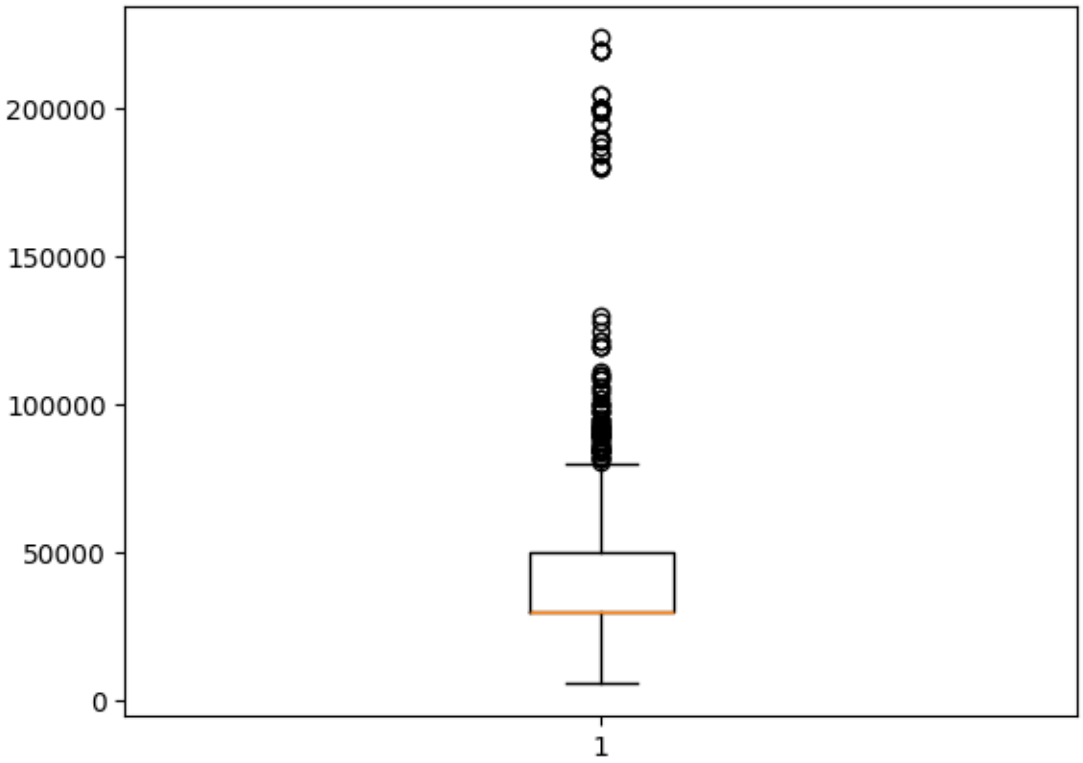


It shows 5 values 1. min() 2. max() 3. median()

[69]:
```
df[df["brand"]=="TVS"]["price"].max()
```

[69]:
```
np.float64(224000.0)
```

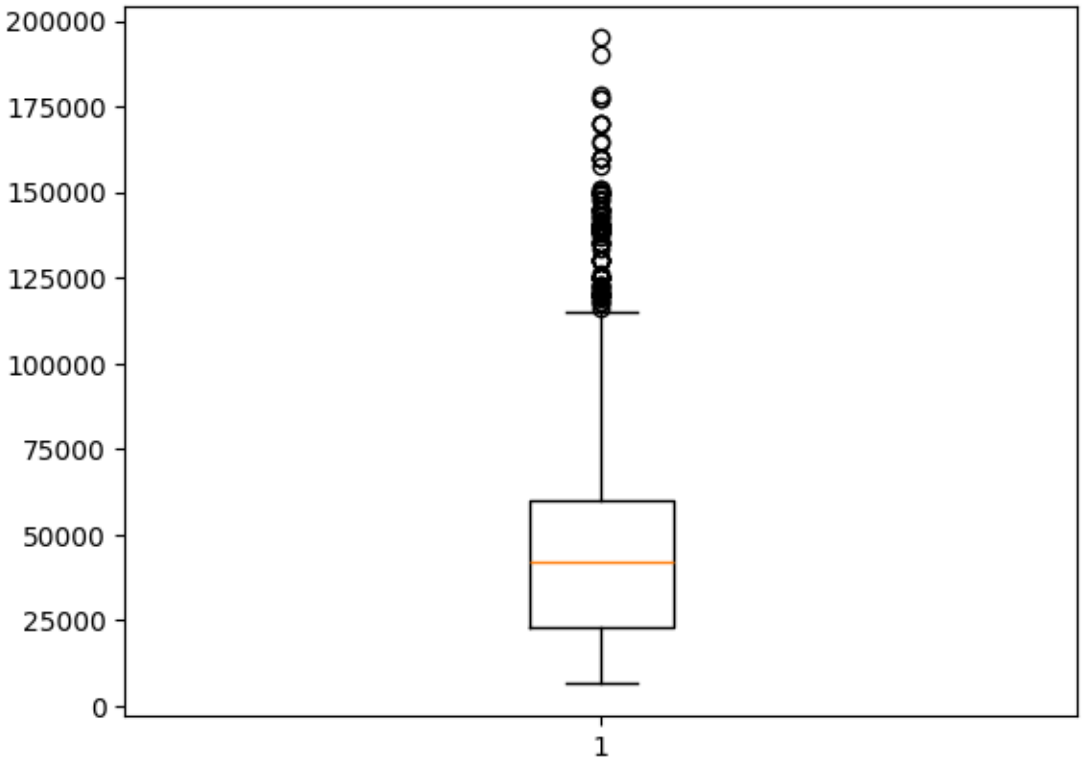
[67]:
```
plt.boxplot(df[df["brand"]=="TVS"]["price"])
plt.show()
```



here these small balls are called as outliers.

[76]:
```
plt.boxplot(df[df["brand"]=="Bajaj"]["price"])
plt.show
```

[76]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```


200000
175000
150000
125000
100000
75000
50000
25000
0
1