

DESCRIPTIVE STATISTICS

ANAKHA R MENON

CH.EN.U4CSE20103

CSE-B

AIM : To understand and implement the concepts of descriptive statistics in python programming.

CODE :

```
In [ ]: %matplotlib inline
```

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

```
In [ ]: mtcars = pd.read_csv("/content/mtcars.csv")
mtcars = mtcars.rename(columns={'Unnamed: 0': 'model'})
mtcars.index = mtcars.model
del mtcars["model"]

mtcars.head()
```

```
Out[ ]:
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
model											
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

```
In [ ]: mtcars.mean()
```

```
Out[ ]: mpg      20.090625
      cyl      6.187500
      disp    230.721875
      hp      146.687500
      drat     3.596563
      wt       3.217250
      qsec    17.848750
      vs       0.437500
      am       0.406250
      gear     3.687500
      carb     2.812500
      dtype: float64
```

```
In [ ]: mtcars.mean(axis=1)
```

```
Out[ ]: model
      Mazda RX4      29.907273
      Mazda RX4 Wag  29.981364
      Datsun 710      23.598182
      Hornet 4 Drive  38.739545
      Hornet Sportabout 53.664545
      Valiant        35.049091
      Duster 360     59.720000
      Merc 240D      24.634545
      Merc 230       27.233636
      Merc 280       31.860000
      Merc 280C      31.787273
      Merc 450SE     46.430909
      Merc 450SL     46.500000
      Merc 450SLC    46.350000
      Cadillac Fleetwood 66.232727
      Lincoln Continental 66.058545
      Chrysler Imperial 65.972273
      Fiat 128       19.440909
      Honda Civic    17.742273
      Toyota Corolla 18.814091
      Toyota Corona  24.888636
      Dodge Challenger 47.240909
      AMC Javelin    46.007727
      Camaro Z28     58.752727
      Pontiac Firebird 57.379545
      Fiat X1-9      18.928636
      Porsche 914-2  24.779091
      Lotus Europa   24.880273
      Ford Pantera L 60.971818
      Ferrari Dino   34.508182
      Maserati Bora  63.155455
      Volvo 142E     26.262727
      dtype: float64
```

```
In [ ]: mtcars.median()
```

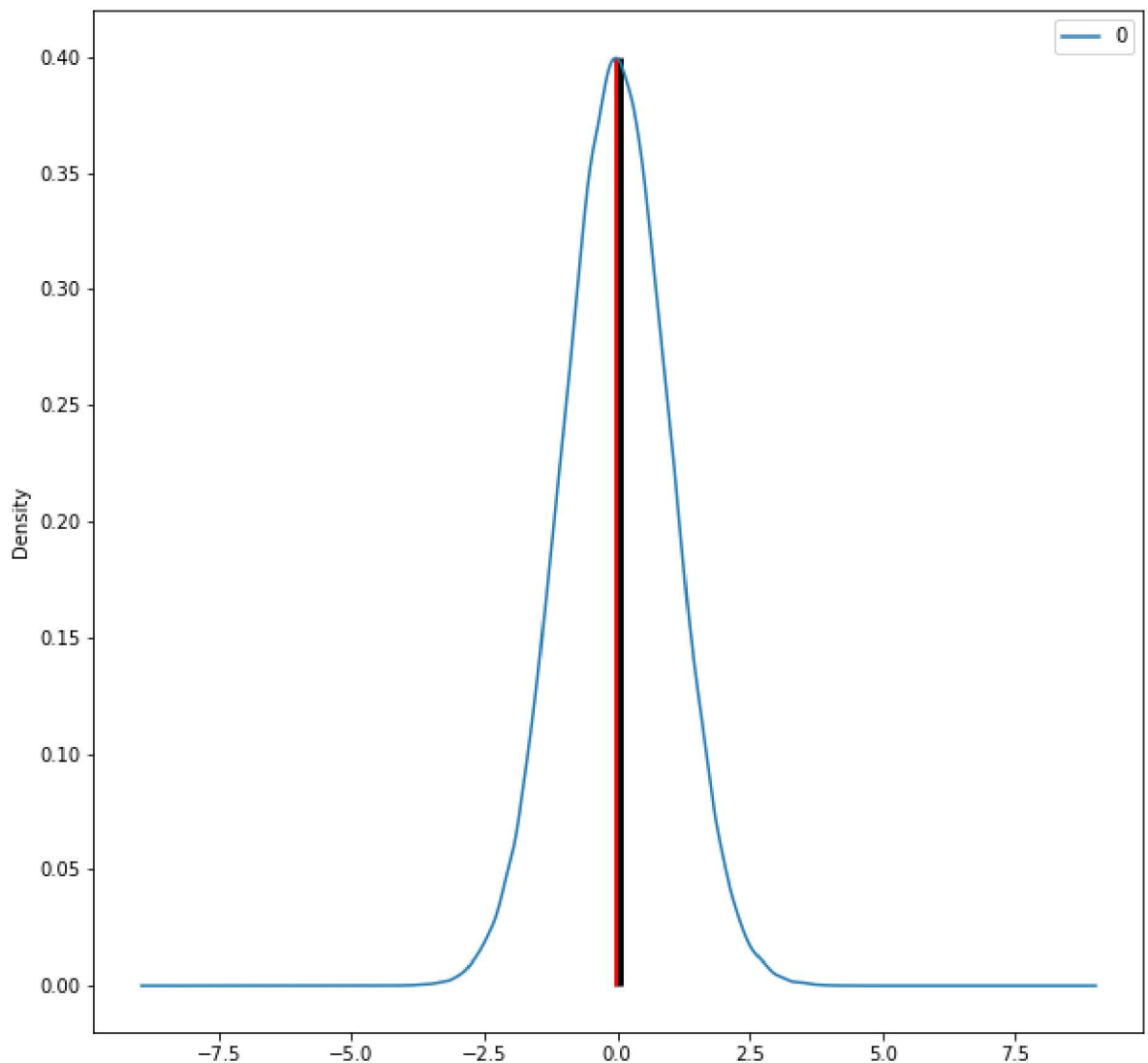
```
Out[ ]: mpg      19.200
      cyl      6.000
      disp    196.300
      hp      123.000
      drat     3.695
      wt       3.325
      qsec    17.710
      vs       0.000
      am       0.000
      gear     4.000
      carb     2.000
      dtype: float64
```

```
In [ ]: norm_data = pd.DataFrame(np.random.normal(size=100000))

norm_data.plot(kind="density",
               figsize=(10,10));

plt.vlines(norm_data.mean(),      # Plot black line at mean
           ymin=0,
           ymax=0.4,
           linewidth=5.0);

plt.vlines(norm_data.median(),   # Plot red line at median
           ymin=0,
           ymax=0.4,
           linewidth=2.0,
           color="red");
```

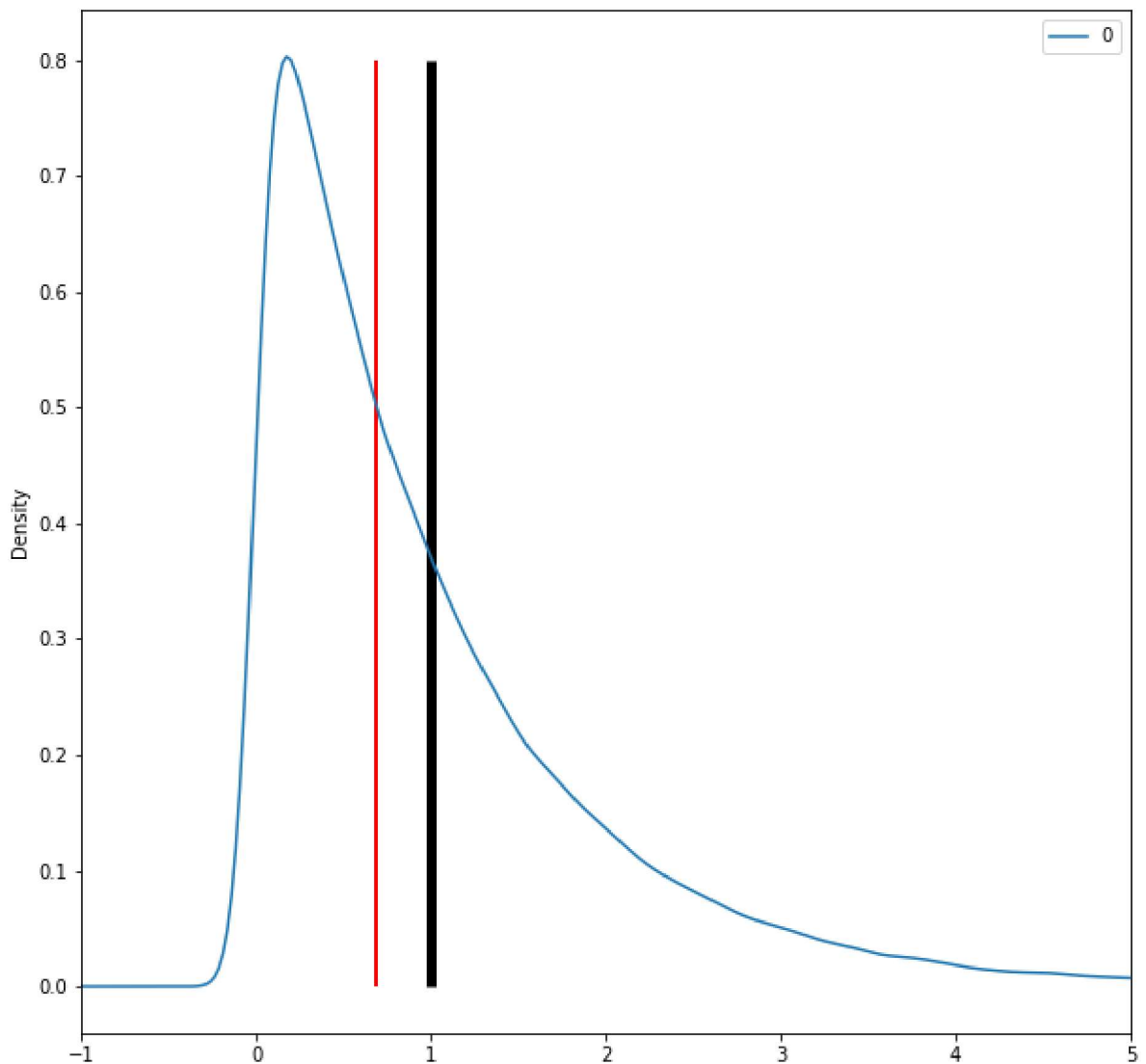


```
In [ ]: skewed_data = pd.DataFrame(np.random.exponential(size=100000))

skewed_data.plot(kind="density",
                 figsize=(10,10),
                 xlim=(-1,5));

plt.vlines(skewed_data.mean(),      # Plot black line at mean
           ymin=0,
           ymax=0.8,
           linewidth=5.0);
```

```
plt.vlines(skewed_data.median(), # Plot red line at median
           ymin=0,
           ymax=0.8,
           linewidth=2.0,
           color="red");
```

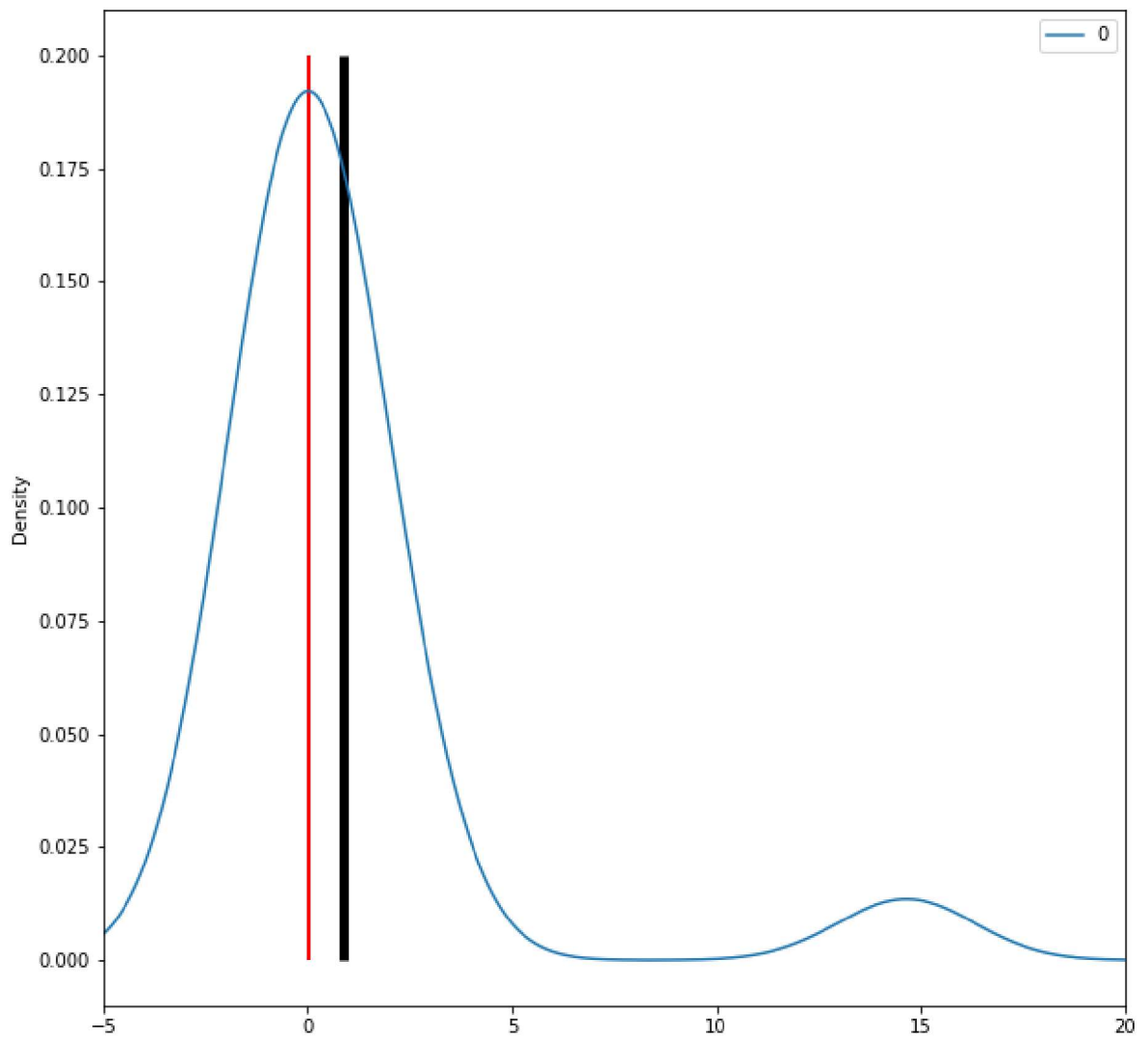


```
In [ ]: norm_data = np.random.normal(size=50)
outliers = np.random.normal(15, size=3)
combined_data = pd.DataFrame(np.concatenate((norm_data, outliers), axis=0))

combined_data.plot(kind="density",
                   figsize=(10,10),
                   xlim=(-5,20));

plt.vlines(combined_data.mean(), # Plot black line at mean
           ymin=0,
           ymax=0.2,
           linewidth=5.0);

plt.vlines(combined_data.median(), # Plot red line at median
           ymin=0,
           ymax=0.2,
           linewidth=2.0,
           color="red");
```



```
In [ ]: mtcars.mode()
```

```
Out[ ]:
```

	mpg	cyl	displ	hp	drat	wt	qsec	vs	am	gear	carb
0	10.4	8.0	275.8	110.0	3.07	3.44	17.02	0.0	0.0	3.0	2.0
1	15.2	NaN	NaN	175.0	3.92	NaN	18.90	NaN	NaN	NaN	4.0
2	19.2	NaN	NaN	180.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	21.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	21.4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	22.8	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	30.4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
In [ ]: max(mtcars["mpg"]) - min(mtcars["mpg"])
```

```
Out[ ]: 23.5
```

```
In [ ]: five_num = [mtcars["mpg"].quantile(0),
                    mtcars["mpg"].quantile(0.25),
                    mtcars["mpg"].quantile(0.50),
                    mtcars["mpg"].quantile(0.75),
                    mtcars["mpg"].quantile(1)]

five_num
```

```
Out[ ]: [10.4, 15.425, 19.2, 22.8, 33.9]
```

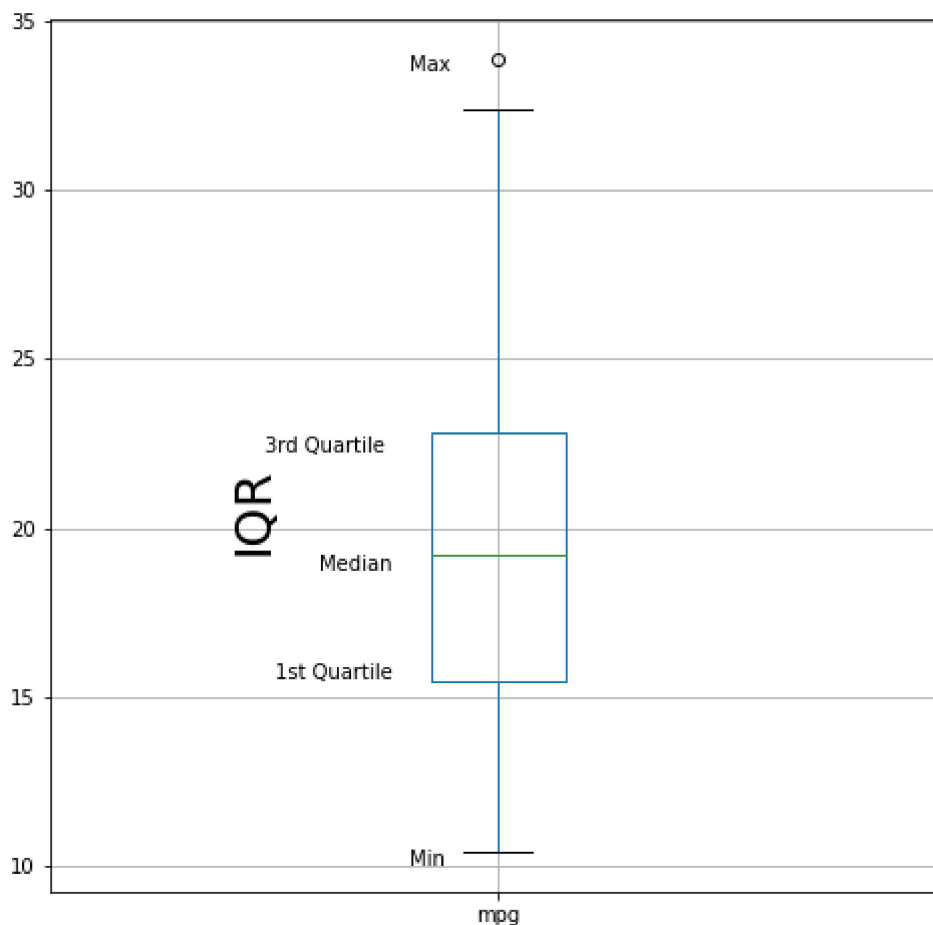
```
In [ ]: mtcars["mpg"].describe()
```

```
Out[ ]: count      32.000000  
mean       20.090625  
std        6.026948  
min        10.400000  
25%        15.425000  
50%        19.200000  
75%        22.800000  
max        33.900000  
Name: mpg, dtype: float64
```

```
In [ ]: mtcars["mpg"].quantile(0.75) - mtcars["mpg"].quantile(0.25)
```

```
Out[ ]: 7.375
```

```
In [ ]: mtcars.boxplot(column="mpg",  
                        return_type='axes',  
                        figsize=(8,8))  
  
plt.text(x=0.74, y=22.25, s="3rd Quartile")  
plt.text(x=0.8, y=18.75, s="Median")  
plt.text(x=0.75, y=15.5, s="1st Quartile")  
plt.text(x=0.9, y=10, s="Min")  
plt.text(x=0.9, y=33.5, s="Max")  
plt.text(x=0.7, y=19.5, s="IQR", rotation=90, size=25);
```



```
In [ ]: mtcars["mpg"].var()
```

```
Out[ ]: 36.32410282258064
```

```
In [ ]: mtcars["mpg"].std()
```

```
Out[ ]: 6.026948052089104
```

```
In [ ]: abs_median_devs = abs(mtcars["mpg"] - mtcars["mpg"].median())  
abs_median_devs.median() * 1.4826
```

```
Out[ ]: 5.4114900000000001
```

```
In [ ]: mtcars["mpg"].skew()
```

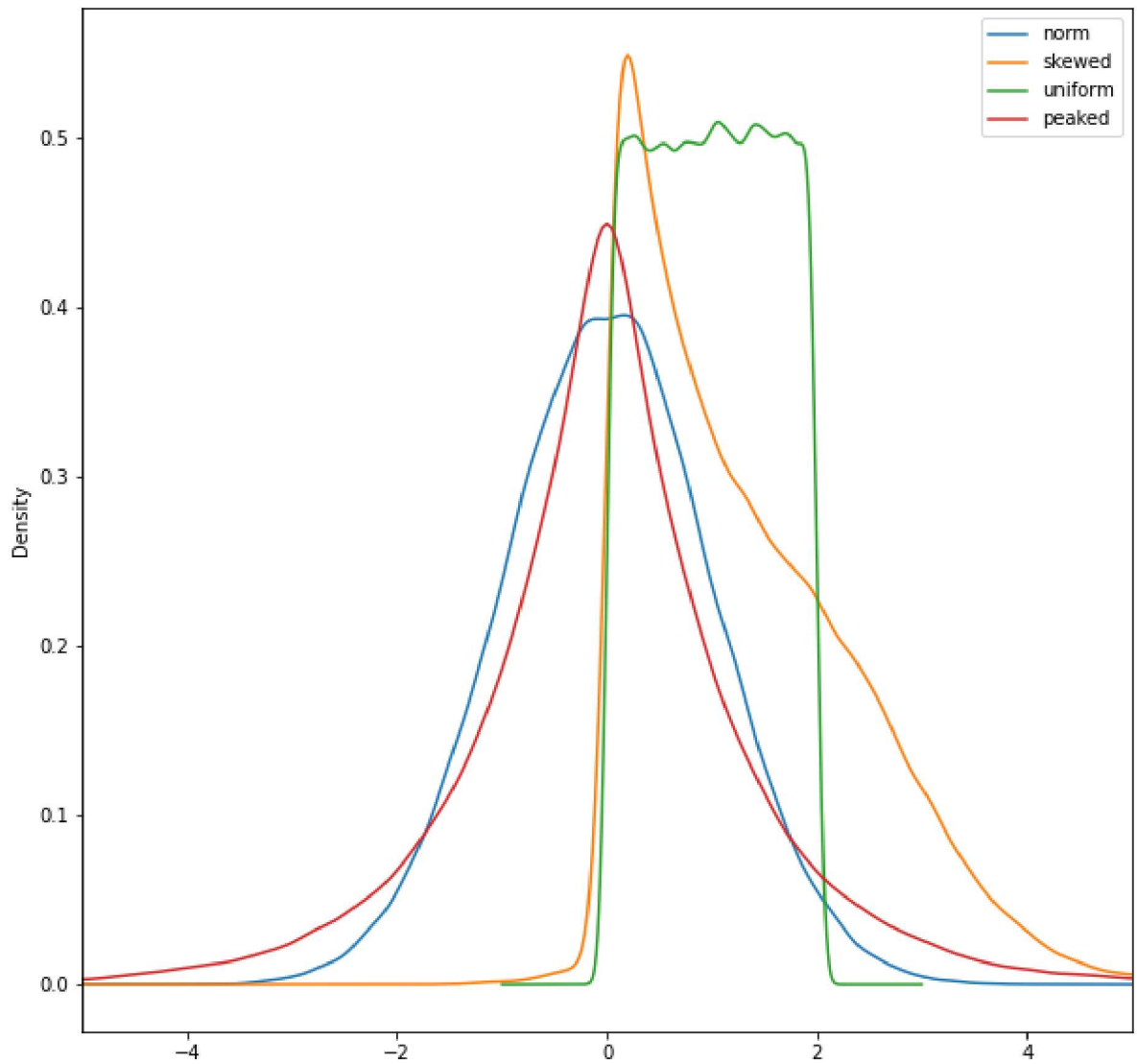
```
Out[ ]: 0.6723771376290805
```

```
In [ ]: mtcars["mpg"].kurt()
```

```
Out[ ]: -0.0220062914240855
```

```
In [ ]: norm_data = np.random.normal(size=100000)  
skewed_data = np.concatenate((np.random.normal(size=35000)+2,  
                               np.random.exponential(size=65000)),  
                               axis=0)  
uniform_data = np.random.uniform(0,2, size=100000)  
peaked_data = np.concatenate((np.random.exponential(size=50000),  
                               np.random.exponential(size=50000)*(-1)),  
                               axis=0)  
  
data_df = pd.DataFrame({"norm":norm_data,  
                        "skewed":skewed_data,  
                        "uniform":uniform_data,  
                        "peaked":peaked_data})
```

```
In [ ]: data_df.plot(kind="density",  
                    figsize=(10,10),  
                    xlim=(-5,5));
```



```
In [ ]: data_df.skew()
```

```
Out[ ]: norm      0.001087
        skewed    1.009027
        uniform   -0.007627
        peaked     0.019298
        dtype: float64
```

```
In [ ]: data_df.kurt()
```

```
Out[ ]: norm      0.015002
        skewed    1.297641
        uniform   -1.198904
        peaked     3.140110
        dtype: float64
```

RESULT : Thus we have successfully implemented the concepts of descriptive statistics in python programming.