Shape and Variation analysis

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

```
df = pd.read_csv("/content/FuelConsumptionCo2.csv")
df
```

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FU
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	
4	2014	ACURA	RDX AWD	SUV - SMALL	3.5	6	AS6	Z	
...	...	...	...	...	...	...	...	...	
1062	2014	VOLVO	XC60 AWD	SUV - SMALL	3.0	6	AS6	X	
1063	2014	VOLVO	XC60 AWD	SUV - SMALL	3.2	6	AS6	X	
1064	2014	VOLVO	XC70 AWD	SUV - SMALL	3.0	6	AS6	X	
1065	2014	VOLVO	XC70 AWD	SUV - SMALL	3.2	6	AS6	X	
1066	2014	VOLVO	XC90 AWD	SUV - STANDARD	3.2	6	AS6	X	

1067 rows × 13 columns

```
df.head()
```

	MODELYEAR	MAKE	MODEL	VEHICLECLASS	ENGINESIZE	CYLINDERS	TRANSMISSION	FUELTYPE	FUELC
0	2014	ACURA	ILX	COMPACT	2.0	4	AS5	Z	
1	2014	ACURA	ILX	COMPACT	2.4	4	M6	Z	
2	2014	ACURA	ILX HYBRID	COMPACT	1.5	4	AV7	Z	
3	2014	ACURA	MDX 4WD	SUV - SMALL	3.5	6	AS6	Z	

```
df.describe()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1067 entries, 0 to 1066
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   MODELYEAR                 1067 non-null   int64  
 1   MAKE                      1067 non-null   object 
 2   MODEL                     1067 non-null   object 
 3   VEHICLECLASS              1067 non-null   object 
 4   ENGINESIZE                1067 non-null   float64
 5   CYLINDERS                 1067 non-null   int64  
 6   TRANSMISSION              1067 non-null   object 
 7   FUELTYPE                  1067 non-null   object 
 8   FUELCONSUMPTION_CITY      1067 non-null   float64
 9   FUELCONSUMPTION_HWY       1067 non-null   float64
 10  FUELCONSUMPTION_COMB      1067 non-null   float64
 11  FUELCONSUMPTION_COMB_MPG  1067 non-null   int64  
 12  CO2EMISSIONS              1067 non-null   int64  
dtypes: float64(4), int64(4), object(5)
memory usage: 108.5+ KB
```

```
cdf = df["CO2EMISSIONS"]
vvb = df["CYLINDERS"]
```

```
type(cdf)
```

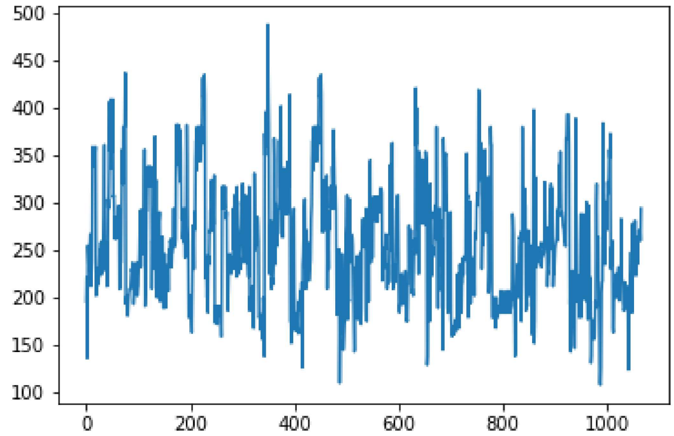
```
pandas.core.series.Series
```

```
cdf
```

```
0       196
1       221
2       136
3       255
4       244
       ... 
1062    271
1063    264
1064    271
1065    260
1066    294
Name: CO2EMISSIONS, Length: 1067, dtype: int64
```

```
plt.plot(cdf)
```

```
[<matplotlib.lines.Line2D at 0x7f968ebe9be0>]
```




```
cdf.mean()
```

```
256.2286785379569
```

```
cdf.median()
```

```
251.0
```

Degrees of freedom refer to the number of independent pieces of information that are used to calculate a statistic

```
# standard deviation
# default ddof = 1
# divded by n - 1
cdf.std()
```

```
63.372304442800065
```

```
# standard deviation
# ddof = 0
# divded by n
cdf.std(ddof=0)
```

```
63.34260099404252
```

```
# variance with ddof = 0
# sum((x_i - x_mean)^2) / n
cdf.var(ddof = 0)
```

```
4012.2851006904766
```

```
# variance with ddof = 1
# sum((x_i - x_mean)^2) / (n-1)
cdf.var(ddof = 1)
```

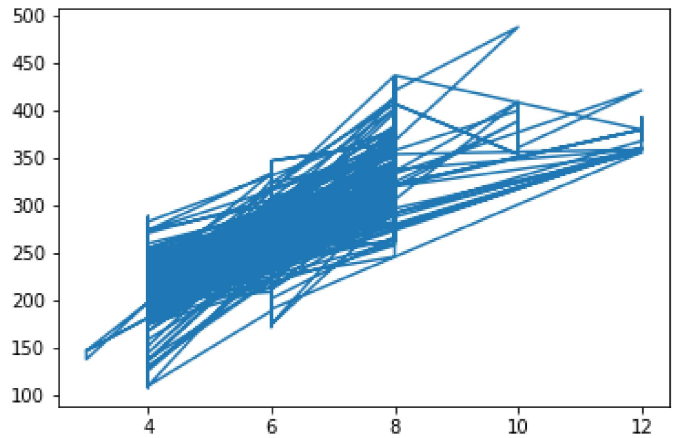
```
4016.048970390937
```

```
# mean (average) absolute deviation
cdf.mad()
```

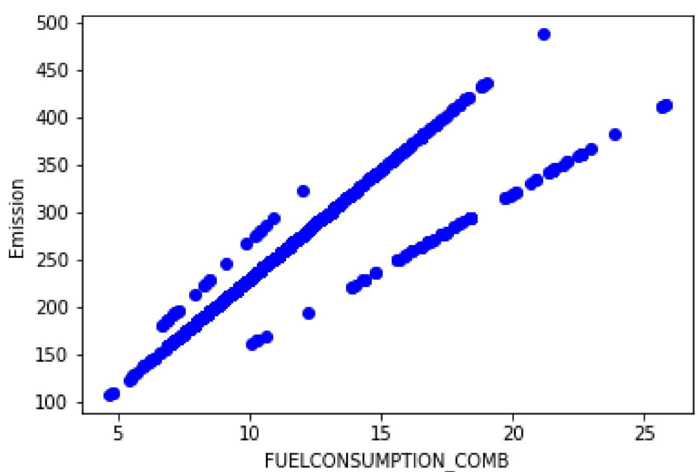
```
51.04539613470135
```

```
plt.plot(vvb, cdf )
```

```
[<matplotlib.lines.Line2D at 0x7f968c2ef730>]
```

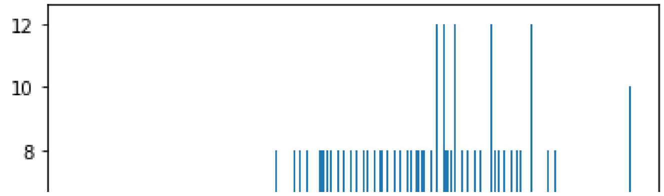



```
plt.scatter(df.FUELCONSUMPTION_COMB, df.CO2EMISSIONS, color = 'blue')
plt.xlabel("FUELCONSUMPTION_COMB")
plt.ylabel("Emission")
plt.show()
```



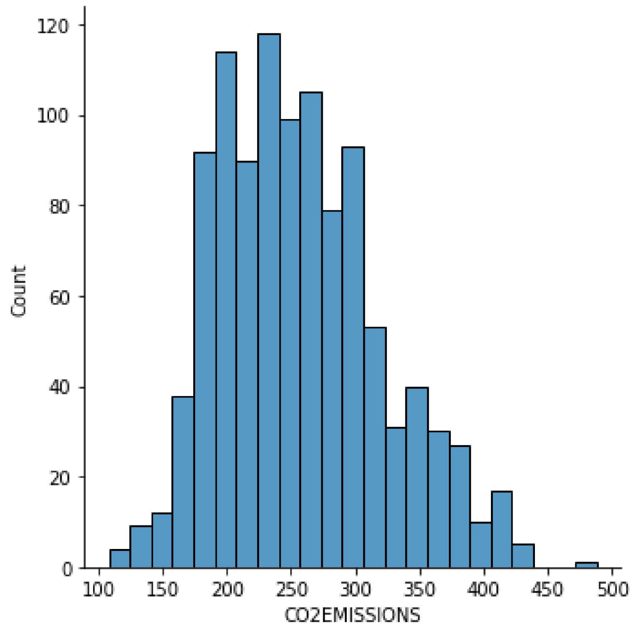

```
plt.bar(cdf, vvb)
```

```
<BarContainer object of 1067 artists>
```



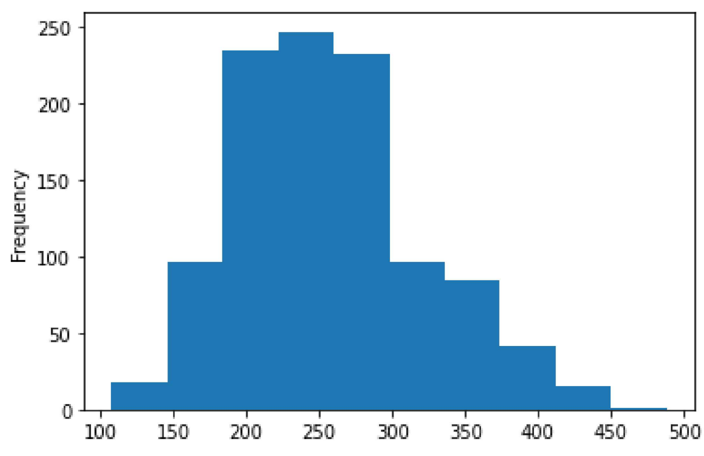

```
sns.displot(cdf)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f968b933fd0>
```




```
cdf.plot(kind="hist")
```

```
<AxesSubplot:ylabel='Frequency'>
```



from the visual analysis we get to know that the chosen data is almost a positively skewed

Skewness

```
cdf.skew()
```

```
0.5195146330615628
```

```
# calculate the third moment of the distribution
third_moment = np.sum((cdf - cdf.mean())**3) / len(cdf)

# calculate the standard deviation of the distribution
std_dev = np.std(cdf, ddof=0)

# calculate the moment of coefficient of skewness
moment_of_coefficient_of_skewness = third_moment / (std_dev**3)

print(moment_of_coefficient_of_skewness)
```

```
0.5187840084529483
```

We implemented the above code to find the moent of coefficient of skewness which gives a (+ve) value and this confirms our earlier assumptions from visual analysis of histogram that, this is a (+ve)ly skewed data.

==> Data is Modedrately skewed.

```
cdf.kurtosis()
```

```
-0.10955312596150169
```

We get a negative kurtosis, it means that our distribution is flatter and has lighter tails than a normal distribution. This type of distribution is called a platykurtic distribution.

Outliers and BoxPlots

Create a box plot using Matplotlib's boxplot function:

```
plt.boxplot(cdf)
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x7f968b79ebb0>,
  <matplotlib.lines.Line2D at 0x7f968b79ee80>],
 'caps': [<matplotlib.lines.Line2D at 0x7f968b7af190>,
  <matplotlib.lines.Line2D at 0x7f968b7af460>],
```

To find outliers, we can use the interquartile range (IQR) method. First, calculate the IQR:

```
 'fliers': [<matplotlib.lines.Line2D at 0x7f968b7afa00>],
```

```
q1, q3 = np.percentile(cdf, [25, 75])
iqr = q3 - q1
iqr
```

```
87.0
```

Then, define the upper and lower bounds:

```
upper_bound = q3 + (1.5 * iqr)
lower_bound = q1 - (1.5 * iqr)
```

Finally, find any data points that fall outside the bounds:

```
outliers = [x for x in cdf if x < lower_bound or x > upper_bound]
print("Outliers:", outliers)
```

```
Outliers: [437, 432, 435, 488, 432, 435]
```

Summary: Mean = 256.2286785379569 Median = 251.0 std deviation = 63.372304442800065 varaince = 4016.048970390937 mean absolute deviation=51.04539613470135 Skewness = 0.5187840084529483, Kurtosis = -0.10955312596150169 (+ve)ly skewed data and is platykurtic distribution.