Descriptive Statistics and Python Implementation

Task

Write a Jupyter Notebook explaining all the Descriptive Statistics.

```
Mean
Median
Mode
Variance
Standard Deviation
Correlation
Normal Distribution (use references)
Feature of Normal Distribution
Positively Skewed & Negatively Skewed Normal Distribution
Effect on Mean, Median and Mode due to Skewness
Explain QQ Plot and show the implementation of the same
Explain Box Cox and show the implementation of the same
```

Explain each topic (mentioned above) with the help of images, code examples (with and without library functions) and formulas.

In [1]:

```python
## Importing necessary Libraries and packages

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

import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```python
## Importing CSV file using pandas
df=pd.read_csv('C:\\Users\\Shubham Shinde\\OneDrive\\Desktop\\Innomatics\\data.csv')
```

1. Shape of Data

In [3]:
```
df.shape   ## Gives shape of data i.e total number of rows and column
```

Out[3]:
```
(50, 7)
```

Total 50 rows and 7 attributes are present in the dataset.

2.Head and Tail Values

In [4]:
```
## gives Head values (by default gives first 5 values)
df.head()
```

Out[4]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	Highest_Qualified_Member	No_of_Earning_Meml
0	5000	8000	3	2000	64200	Under-Graduate	
1	6000	7000	2	3000	79920	Illiterate	
2	10000	4500	2	0	112800	Under-Graduate	
3	10000	2000	1	0	97200	Illiterate	
4	12500	12000	2	3000	147000	Graduate	

In [5]:
```
## Gives tail(last)values (by default gives last 5 values)
df.tail()
```

Out[5]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	Highest_Qualified_Member	No_of_Earning_Men
45	90000	48000	7	0	885600	Post-Graduate	
46	98000	25000	5	0	1152480	Professional	
47	100000	30000	6	0	1404000	Graduate	
48	100000	50000	4	20000	1032000	Professional	
49	100000	40000	6	10000	1320000	Post-Graduate	

3. Checking Data Info Using info function

In [6]:
```
## Checking Data info using info() Function
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 7 columns):
 #   Column                   Non-Null Count  Dtype 
---  ------                   --------------  ----- 
 0   Mthly_HH_Income          50 non-null     int64 
 1   Mthly_HH_Expense         50 non-null     int64 
 2   No_of_Fly_Members        50 non-null     int64 
 3   Emi_or_Rent_Amt          50 non-null     int64 
 4   Annual_HH_Income         50 non-null     int64 
 5   Highest_Qualified_Member  50 non-null     object
 6   No_of_Earning_Members    50 non-null     int64 
dtypes: int64(6), object(1)
memory usage: 2.9+ KB
```

This Function gives the total number attributes,non null value count and data type

In [7]:
```
## Checking Data type using dtype function
df.dtypes
```

Out[7]:
```
Mthly_HH_Income              int64
Mthly_HH_Expense             int64
No_of_Fly_Members            int64
Emi_or_Rent_Amt              int64
Annual_HH_Income             int64
Highest_Qualified_Member    object
No_of_Earning_Members        int64
dtype: object
```

Total 7 Attributes are present in the dataset.

only 1 categorical attribute = Highest_Qualified_Member

6 Numerical Attributes = Mthly_HH_Income , Mthly_HH_Expense , No_of_Fly_Members , Emi_or_Rent_Amt , Annual_HH_Income , No_of_Earning_Members

4.Unique value count of categorical Data

In [8]:
```
## It gives the unique values present in the respected data
df['Highest_Qualified_Member'].unique()
```

Out[8]:
```
array(['Under-Graduate', 'Illiterate', 'Graduate', 'Post-Graduate',
```

```
        'Professional'], dtype=object)
```

In [9]:
```
## It gives unique value count present in the respected data
df['Highest_Qualified_Member'].value_counts()
```

Out[9]:
```
Graduate          19
Professional      10
Under-Graduate    10
Post-Graduate      6
Illiterate         5
Name: Highest_Qualified_Member, dtype: int64
```

5.Measure of Central Tendency

Measures of central tendency are a set of “middle” values representative of the data points.

Central tendency describes the distribution of data focusing on the central location around which all other data are clustered.

It is the opposite of dispersion that measures how far the observations are scattered with respect to the central value.

5.1.Mean

Mean is the average of some data points.

$$\boxed{\overline{X} = \frac{\sum X}{N}}$$

In [10]:
```
## Mean gives the Average value of respected data
df.mean()
```

Out[10]:
```
Mthly_HH_Income            41558.00
Mthly_HH_Expense           18818.00
No_of_Fly_Members              4.06
Emi_or_Rent_Amt             3060.00
Annual_HH_Income          490019.04
No_of_Earning_Members          1.46
dtype: float64
```

5.2.Median

Median is the number at the center of a series after they are ordered (ascending or descending).

In [11]:
```
## Median give the middle value of respected data
df.median()
```

Out[11]:
```
Mthly_HH_Income           35000.0
Mthly_HH_Expense          15500.0
No_of_Fly_Members             4.0
Emi_or_Rent_Amt               0.0
Annual_HH_Income         447420.0
No_of_Earning_Members         1.0
dtype: float64
```

5.3.Mode

[1,2, 3, 4, 5,5] — the most frequently occurring one is 5; that's mode.

A distribution can have more than one mode as in the list [2, 2, 3, 4, 4]; it's called bimodal distribution of a discrete variable.

Along this logic, a distribution with more than two modes are called multimodal distribution.

In [12]:
```
## Mode gives the most occured/repeated value in the respected data.
df.mode()
```

Out[12]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	Highest_Qualified_Member	No_of_Earning_Meml
0	45000	25000	4	0	590400	Graduate	

6.Describe

In [13]:
```
df.describe()
```

Out[13]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	No_of_Earning_Members
count	50.000000	50.000000	50.000000	50.000000	5.000000e+01	50.000000
mean	41558.000000	18818.000000	4.060000	3060.000000	4.900190e+05	1.460000
std	26097.908979	12090.216824	1.517382	6241.434948	3.201358e+05	0.734291

min	5000.000000	2000.000000	1.000000	0.000000	6.420000e+04	1.000000
25%	23550.000000	10000.000000	3.000000	0.000000	2.587500e+05	1.000000
50%	35000.000000	15500.000000	4.000000	0.000000	4.474200e+05	1.000000
75%	50375.000000	25000.000000	5.000000	3500.000000	5.947200e+05	2.000000
max	100000.000000	50000.000000	7.000000	35000.000000	1.404000e+06	4.000000

Describe function gives the following information

```
1.Count - It gives the total count of values present in the each attribute.
2.mean  - It gives the mean value of each attribute.
3.std   - it gives the Standard Deviation of of each attribute.
4.min   - It gives the minimum value present in the data.
5.25%   - It gives the the value which present at 25 percentile.
6.50%   - It gives the the value which present at 50 percentile.
7.75%   - It gives the the value which present at 75 percentile.
8.max   - It gives the maximum value present in the data.
```

7. Measures of Dispersion

Dispersion is a way of describing how data is spread around an average value.

data set have large differences between data values, then data set is said as widely scattered data set.

data values are close to each other the data set is said to be tightly clustered data set.

```
E.g.
  55,57,55,58,56
  10,22,31,45,27
```

7.1 Range

The difference between largest and smallest data value in given data set is known as Range of given data set.

For example, if we have data set as1,3,4,2,7,8,12,6.

Range = 12–1 = 11.

In [14]:
```
for i in df.columns:
    if df[i].dtypes =='object':
        pass
    else:
        print(f'Range of {i} is {df[i].max()-df[i].min()}')
```

```
Range of Mthly_HH_Income is 95000
Range of Mthly_HH_Expense is 48000
Range of No_of_Fly_Members is 6
Range of Emi_or_Rent_Amt is 35000
Range of Annual_HH_Income is 1339800
Range of No_of_Earning_Members is 3
```

7.2 Mean Absolute Deviation

Average of absolute differences from the mean is known as mean deviation.

$$M.D. = \frac{\sum_{i=1}^{N} |x_i - \bar{x}|}{N}$$

((value-mean)/(total count of values))

In [15]:
```
for i in df.columns:
    if df[i].dtypes =='object':
        pass
    else:
        print(f'Mean Absolute Deviation of {i} is {df[i].mad()}')
```

```
Mean Absolute Deviation of Mthly_HH_Income is 20288.96
Mean Absolute Deviation of Mthly_HH_Expense is 9247.44
Mean Absolute Deviation of No_of_Fly_Members is 1.1991999999999998
Mean Absolute Deviation of Emi_or_Rent_Amt is 3866.4
Mean Absolute Deviation of Annual_HH_Income is 238469.56800000006
Mean Absolute Deviation of No_of_Earning_Members is 0.6072000000000002
```

7.3 Variance

The Variance is the average of squared differences from the mean.

$$Variance = \sigma^2 = \frac{\sum_{i=1}^{N} |x_i - \bar{x}\,|^2}{N}$$

((value-mean)^2/(total count of values))

In [16]:
```
for i in df.columns:
    if df[i].dtypes =='object':
        pass
    else:
        print(f'Variance of {i} is {df[i].var()}')
```

```
Variance of Mthly_HH_Income is 681100853.0612245
Variance of Mthly_HH_Expense is 146173342.85714287
Variance of No_of_Fly_Members is 2.302448979591837
Variance of Emi_or_Rent_Amt is 38955510.20408163
Variance of Annual_HH_Income is 102486925397.91666
Variance of No_of_Earning_Members is 0.5391836734693878
```

7.4 Standard Deviation

The Standard Deviation is the square root of variance.

$$\text{Standard Deviation} = \sigma = \sqrt{Variance}$$

In [17]:
```
for i in df.columns:
    if df[i].dtypes =='object':
        pass
    else:
        print(f'Standard Deviation of {i} is {df[i].std()}')
```

```
Standard Deviation of Mthly_HH_Income is 26097.908978713687
Standard Deviation of Mthly_HH_Expense is 12090.216824240286
```

```
Standard Deviation of No_of_Fly_Members is 1.5173822786601394
Standard Deviation of Emi_or_Rent_Amt is 6241.434947516607
Standard Deviation of Annual_HH_Income is 320135.79212252516
Standard Deviation of No_of_Earning_Members is 0.7342912729083656
```

8.Correlation

Variables within a dataset can be related for lots of reasons.

For example:

1. One variable could cause or depend on the values of another variable.
2. One variable could be lightly associated with another variable.
3. Two variables could depend on a third unknown variable.

It can be useful in data analysis and modeling to better understand the relationships between variables. The statistical relationship between two variables is referred to as their correlation.

A correlation could be positive, meaning both variables move in the same direction, or negative, meaning that when one variable's value increases, the other variables' values decrease. Correlation can also be neutral or zero, meaning that the variables are unrelated.

1. Positive Correlation: both variables change in the same direction.
2. Neutral Correlation: No relationship in the change of the variables.
3. Negative Correlation: variables change in opposite directions.

The performance of some algorithms can deteriorate if two or more variables are tightly related, called multicollinearity. An example is linear regression, where one of the offending correlated variables should be removed in order to improve the skill of the model.

We may also be interested in the correlation between input variables with the output variable in order provide insight into which variables may or may not be relevant as input for developing a model.

The structure of the relationship may be known, e.g. it may be linear, or we may have no idea whether a relationship exists between two variables or what structure it may take. Depending what is known about the relationship and the distribution of the variables, different correlation scores can be calculated.

For More info -https://machinelearningmastery.com/how-to-use-correlation-to-understand-the-relationship-between-variables/

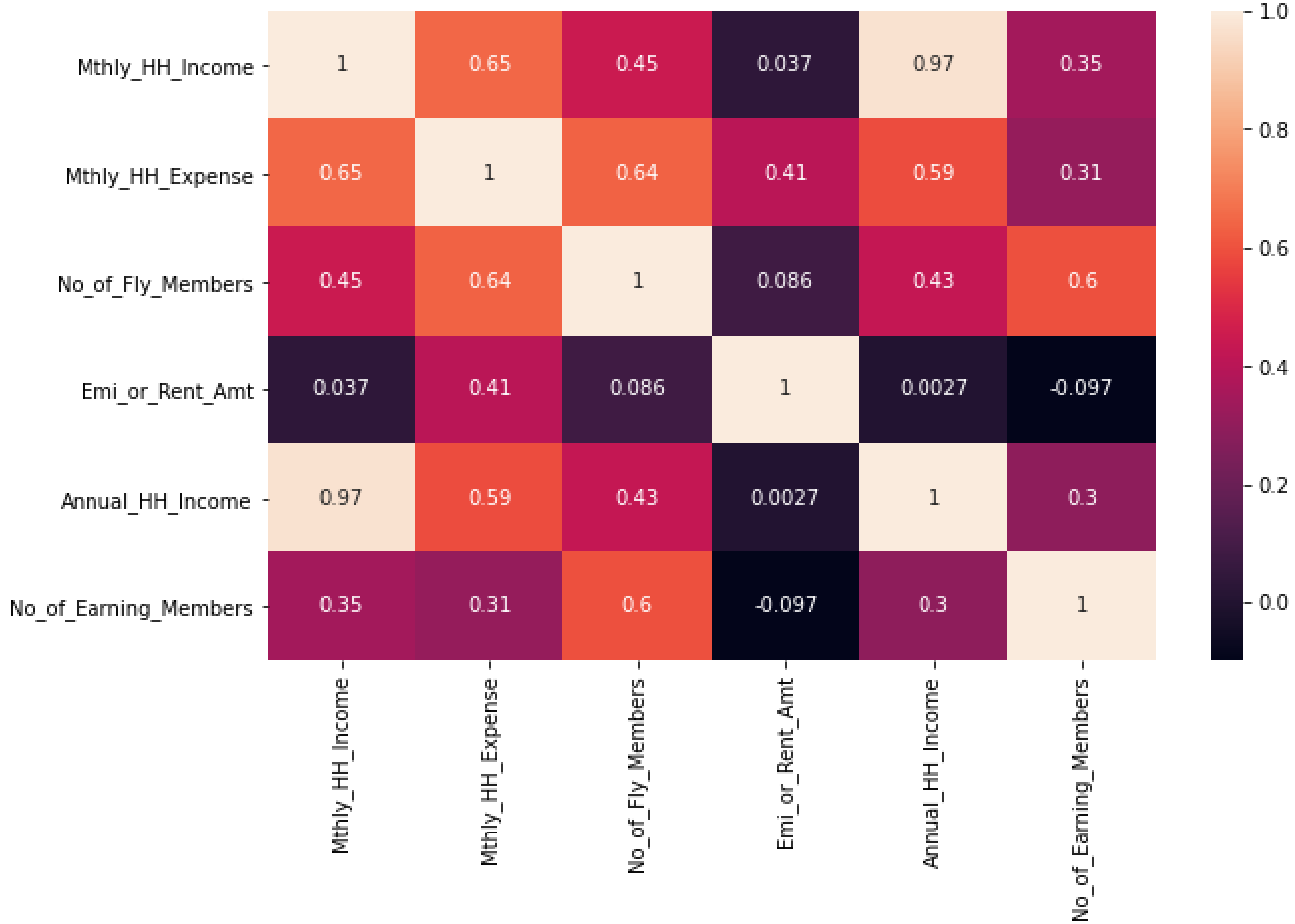
In [18]:
```
##This Function check the correlation between the independent variables.
df.corr()
```

Out[18]:

	Mthly_HH_Income	Mthly_HH_Expense	No_of_Fly_Members	Emi_or_Rent_Amt	Annual_HH_Income	No_of_Earning_Members
Mthly_HH_Income	1.000000	0.649215	0.448317	0.036976	0.970315	0.347883
Mthly_HH_Expense	0.649215	1.000000	0.639702	0.405280	0.591222	0.311915
No_of_Fly_Members	0.448317	0.639702	1.000000	0.085808	0.430868	0.597482
Emi_or_Rent_Amt	0.036976	0.405280	0.085808	1.000000	0.002716	-0.097431
Annual_HH_Income	0.970315	0.591222	0.430868	0.002716	1.000000	0.296679
No_of_Earning_Members	0.347883	0.311915	0.597482	-0.097431	0.296679	1.000000

In [19]:
```
##Plotting the heatmap of of correlation to get deep insights of correlations
plt.figure(figsize=(10,6))
sns.heatmap(df.corr(),annot=True,data=df)
```

Out[19]:
```
<AxesSubplot:>
```



Mthly_HH_Income and Annual_HH_Income are highly positively correlated with each other.

9.Skewness

Skewness is the measure of symmetry or asymmetry of data distribution.A distribution or data set is said to be symmetric if it looks the same to the left and right points of the center.

Type

```
1.Right skewness or Positive skewness
2.Left skewness or Negative skewness
```

Positive Skew

This is the case when the tail on the right side of the curve is bigger than that on the left side. For these distributions, mean is greater than the mode.

Negative Skew

This is the case when the tail on the left side of the curve is bigger than that on the right side. For these distributions, mean is smaller than the mode.

The most commonly used method of calculating Skewness is

$$Skewness = \frac{3\,(Mean - Median)}{Std \quad Deviation}$$

In [20]:
```
## This Function gives the skewness value of each attribute.
df.skew()
```

Out[20]:
```
Mthly_HH_Income          0.924615
Mthly_HH_Expense         1.199461
No_of_Fly_Members        0.113674
Emi_or_Rent_Amt          3.403680
Annual_HH_Income         1.192949
No_of_Earning_Members    1.593301
dtype: float64
```

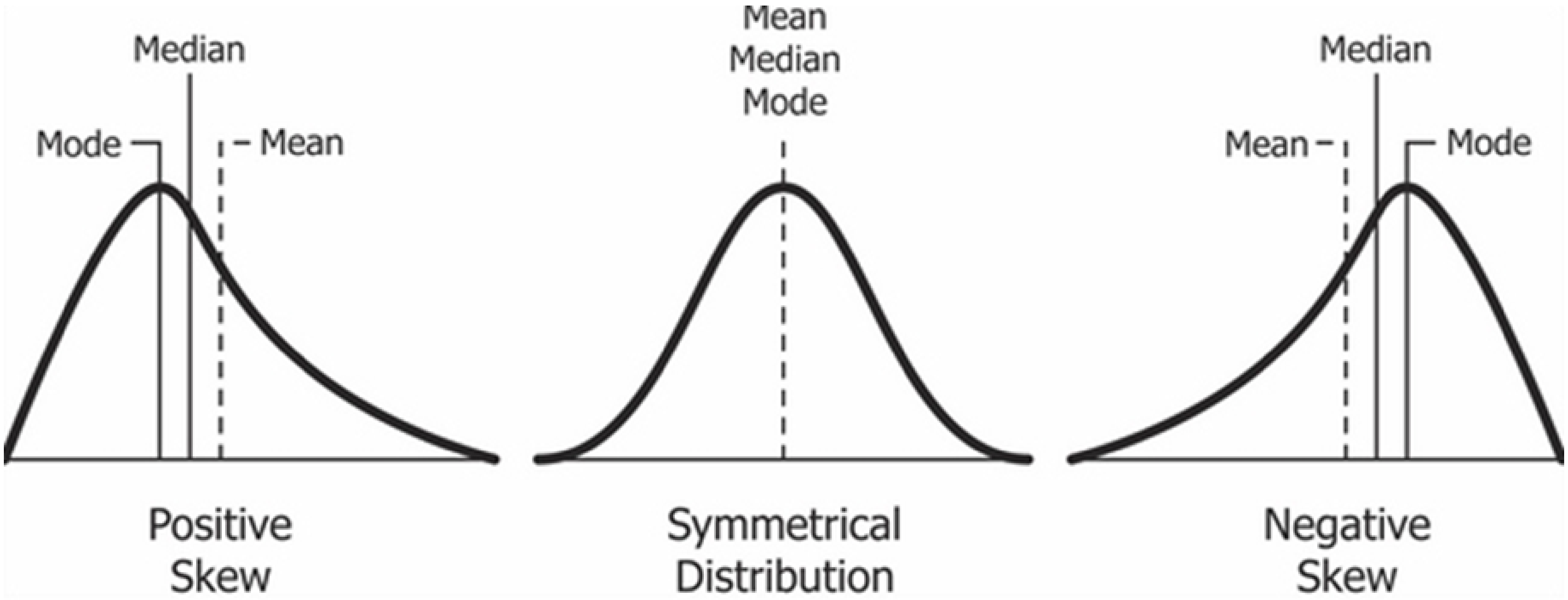
Right skewness

A right-skewed distribution will have a long tail in the right direction on the number line such that the mean of the total value of all data points will eventually go up. 

Left skewness

A left-skewed distribution will have a long tail in the left direction on the number line such that the mean of the total intrinsic value of all data points will eventually go down. 

If the skewness is zero, the distribution is symmetrical. If it is negative, the distribution is Negatively Skewed and if it is positive, it is Positively Skewed.



Positive Skew (Right Skewed) = Mean => Median => Mode

Symmetrical (Normal) Distribution = Mean = Median = Mode

Negative Skew (Left Skewed) = Mode => Median => Mean

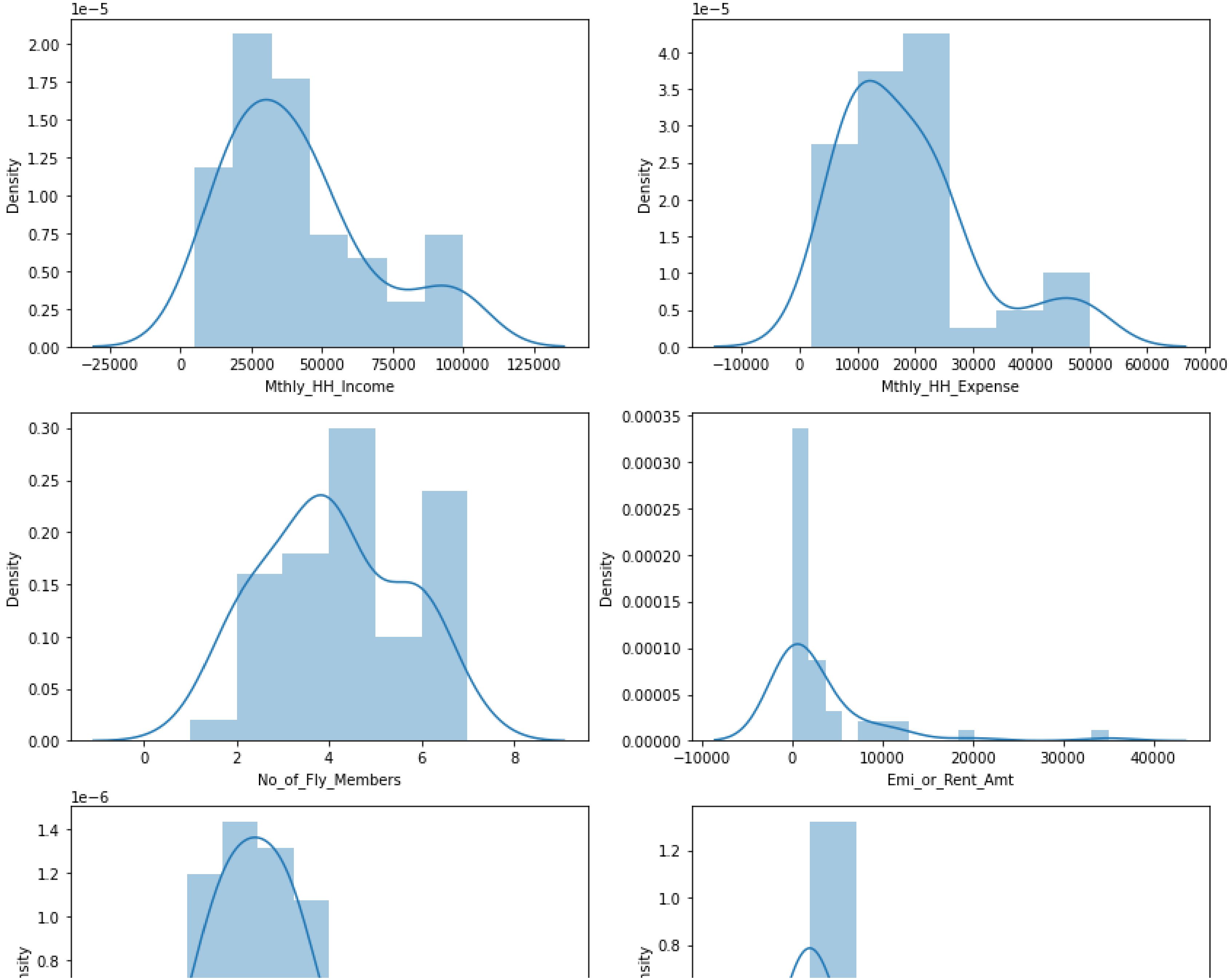
for more info=https://www.analyticsvidhya.com/blog/2020/04/statistics-data-science-normal-distribution/

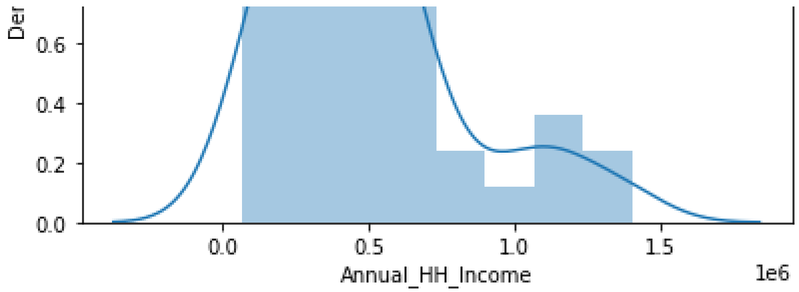
10.Distribution of Data

In [21]:

```
## Checking the Distribution of Data by plotting Distribution plot.
i=1
plt.figure(figsize=(14,34))
for col in df.columns:
    if df[col].dtypes!='object':
        plt.subplot(7,2,i)
        sns.distplot(df[col])
        i=i+1
    else:
```

```
        pass
plt.show()
```





Observation

```
1.Data is Right skewed i.e Positively Distributed.
```

11.Kurtosis

Kurtosis is the characteristic of being flat or peaked. It is a measure of whether data is heavy-tailed or light-tailed in a normal distribution.

Percentile coefficient of Kurtosis

```
 Ku=Q / (P90 — P10)
```

Where,

```
Q= Quartile deviation
P90=90th percentile
P10=10th percentile
```

A large value of kurtosis is often considered as riskier because data may tend to give an outlier value as an outcome with greater distance from the mean if applied to any machine learning algorithm.

Types of Kurtosis

```
1.Mesokurtic
2.Leptokurtic
```

3.Platykurtic

1.Mesokurtic

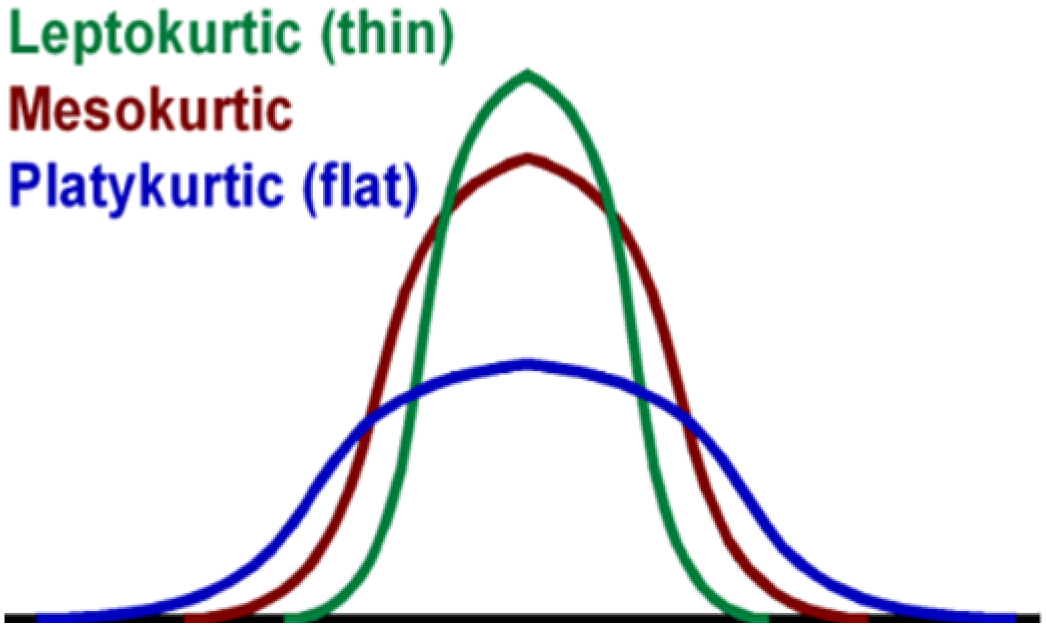
This distribution has the tails often similar to normal distribution.

2.Leptokurtic

This distribution will be having very long and skinny tails. This means there are more chances of the presence of outliers.

3.Platykurtic

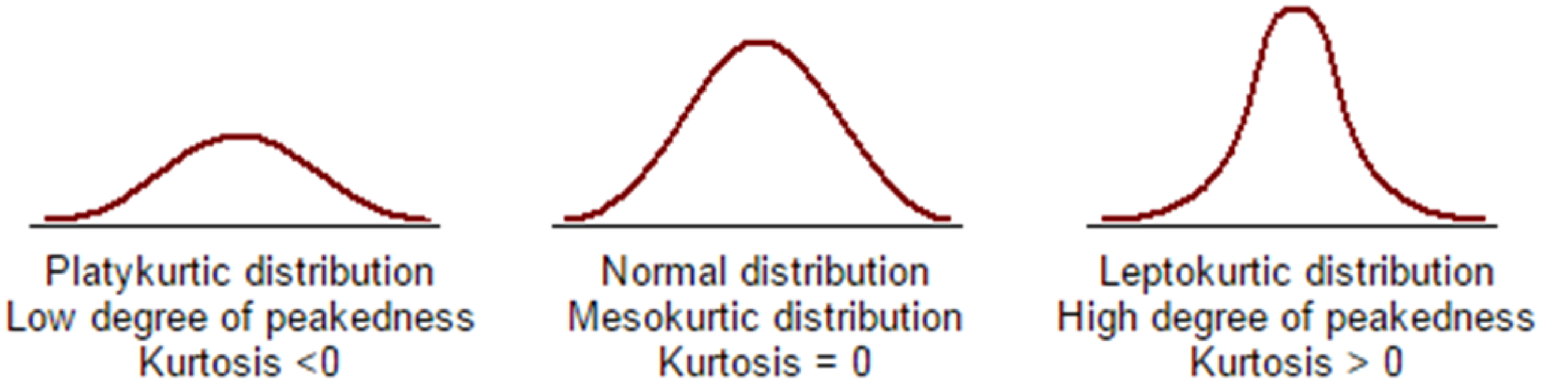
This distribution will be having very low and stretched around center tails which means most of the data points are present in high proximity with mean.



Mesokurtic — This is the case when the kurtosis is zero, similar to the normal distributions.

Leptokurtic — This is when the tail of the distribution is heavy (outlier present) and kurtosis is higher than that of the normal distribution.

Platykurtic — This is when the tail of the distribution is light(no outlier) and kurtosis is lesser than that of the normal distribution.



In [22]:
```
df.kurtosis()
```

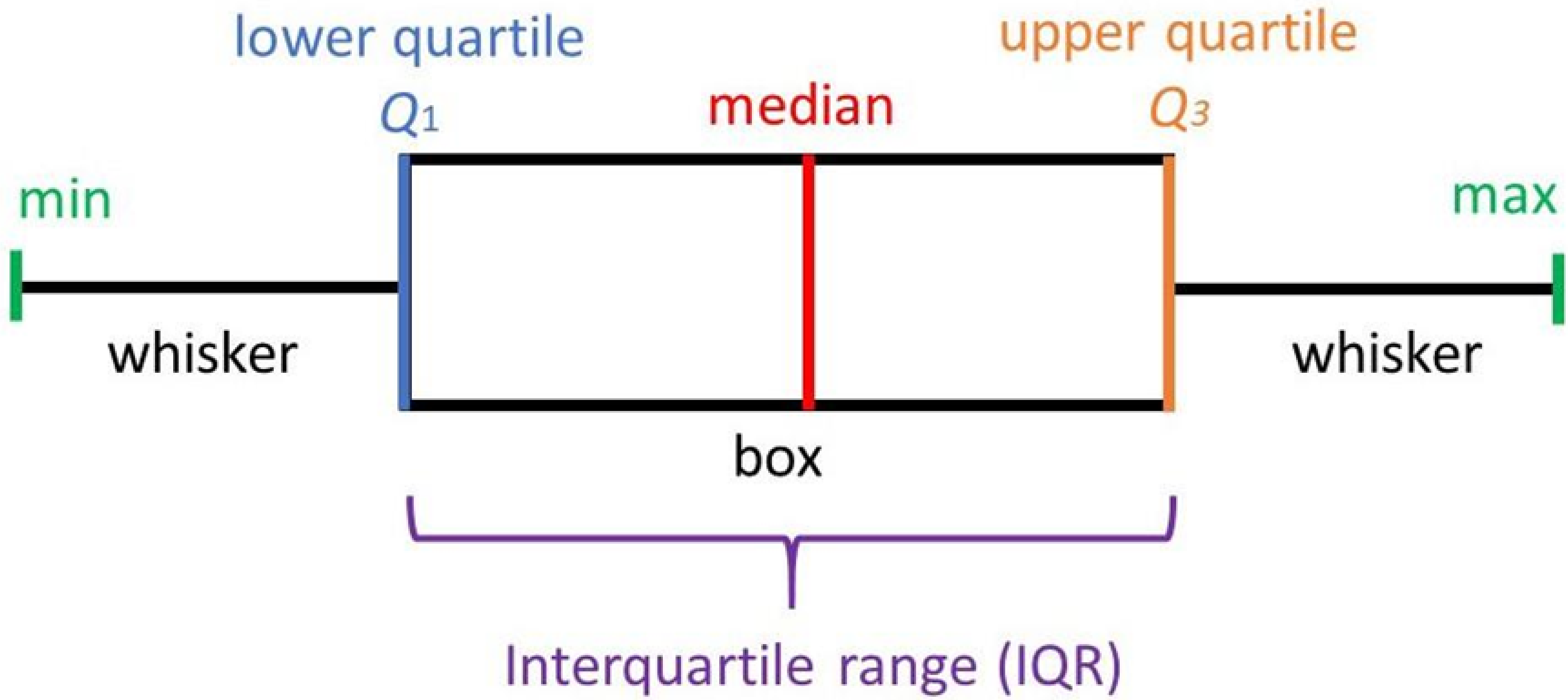
Out[22]:
```
Mthly_HH_Income            0.115550
Mthly_HH_Expense           0.942490
No_of_Fly_Members         -0.851445
Emi_or_Rent_Amt           14.202523
Annual_HH_Income           1.101291
No_of_Earning_Members      2.093212
dtype: float64
```

'No_of_Fly_Members' has Platykurtic Distribution.

'Emi_or_Rent_Amt' has Leptokurtic Distribution.I has very high Kurtosis value.It May Contains More Outliers Than any other attribut.

Other Attribute Has also Leptokurtic Distribution but they have low kurti=osis value which means less outliers are present in the data.

12.Boxplot



1.Minimum Value: It is the least numerical value of a dataset. This defines the left boundary.

2.Median (Q2): It is the number that divides the data into two halves. If the number of numbers in the data is ODD, the median number is the middle data value. If the number of numbers in the data is EVEN, the median number is the mean of the two middle data values. This is only applicable if the numbers in the dataset are arranged in increasing/ascending order.

3.Lower Half: It is the set of values that lie to the left of the Median (Q2)

4.Upper Half: It is the set of values that lie to the right of the Median (Q2)

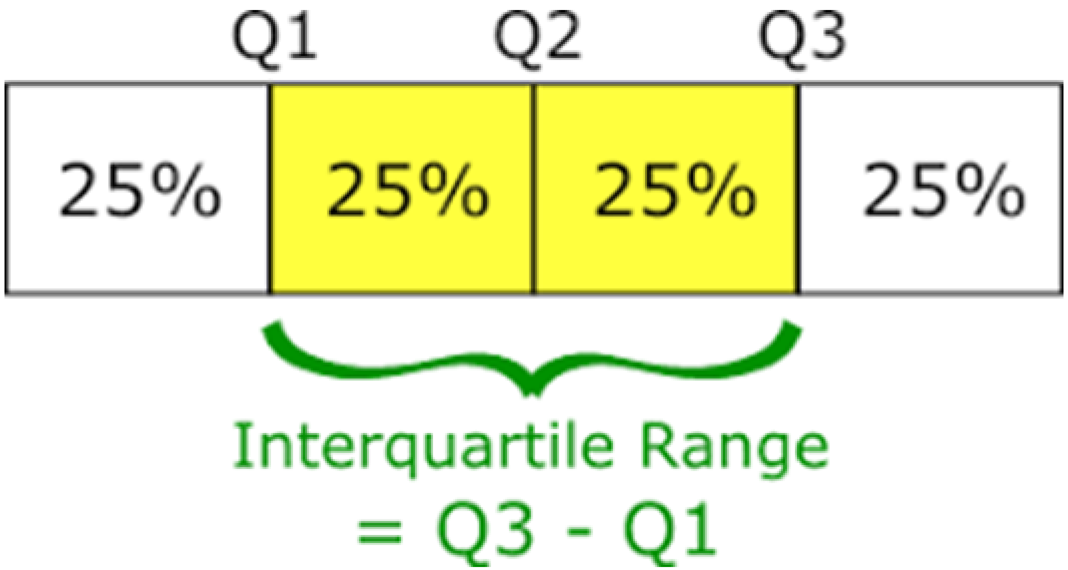
5.Q1 (25th Percentile): It is the median of the lower half of the dataset.

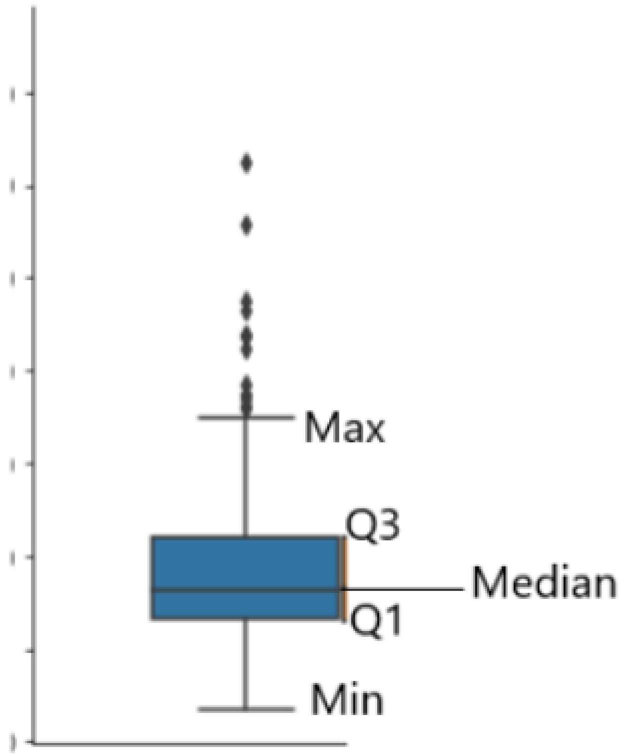
6.Q3 (75th Percentile): It is the median of the upper half of the dataset.

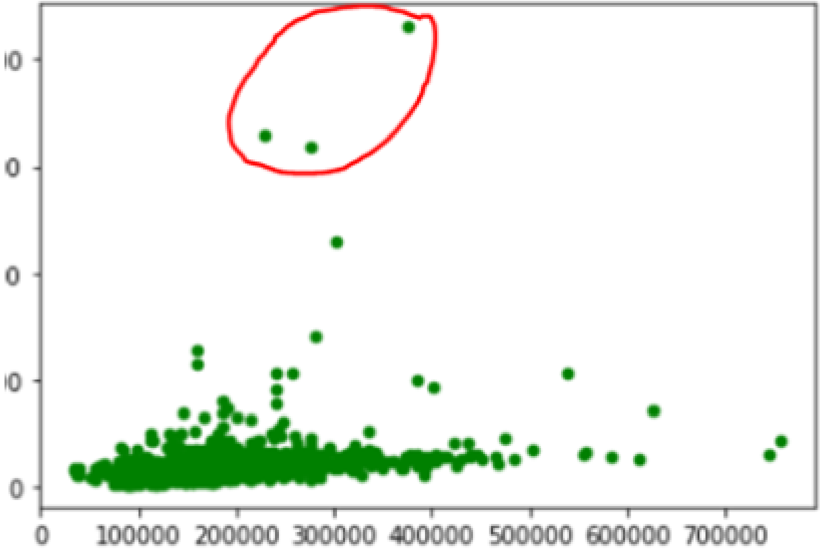
7.Maximum Value: It is the greatest numerical value of a dataset. This defines the right boundary.

8.Range: It is the difference between the maximum and the minimum value.

9.Interquartile Range (IQR): It is the difference between Q3 and Q1.




Max
Q3
Median
Q1
Min

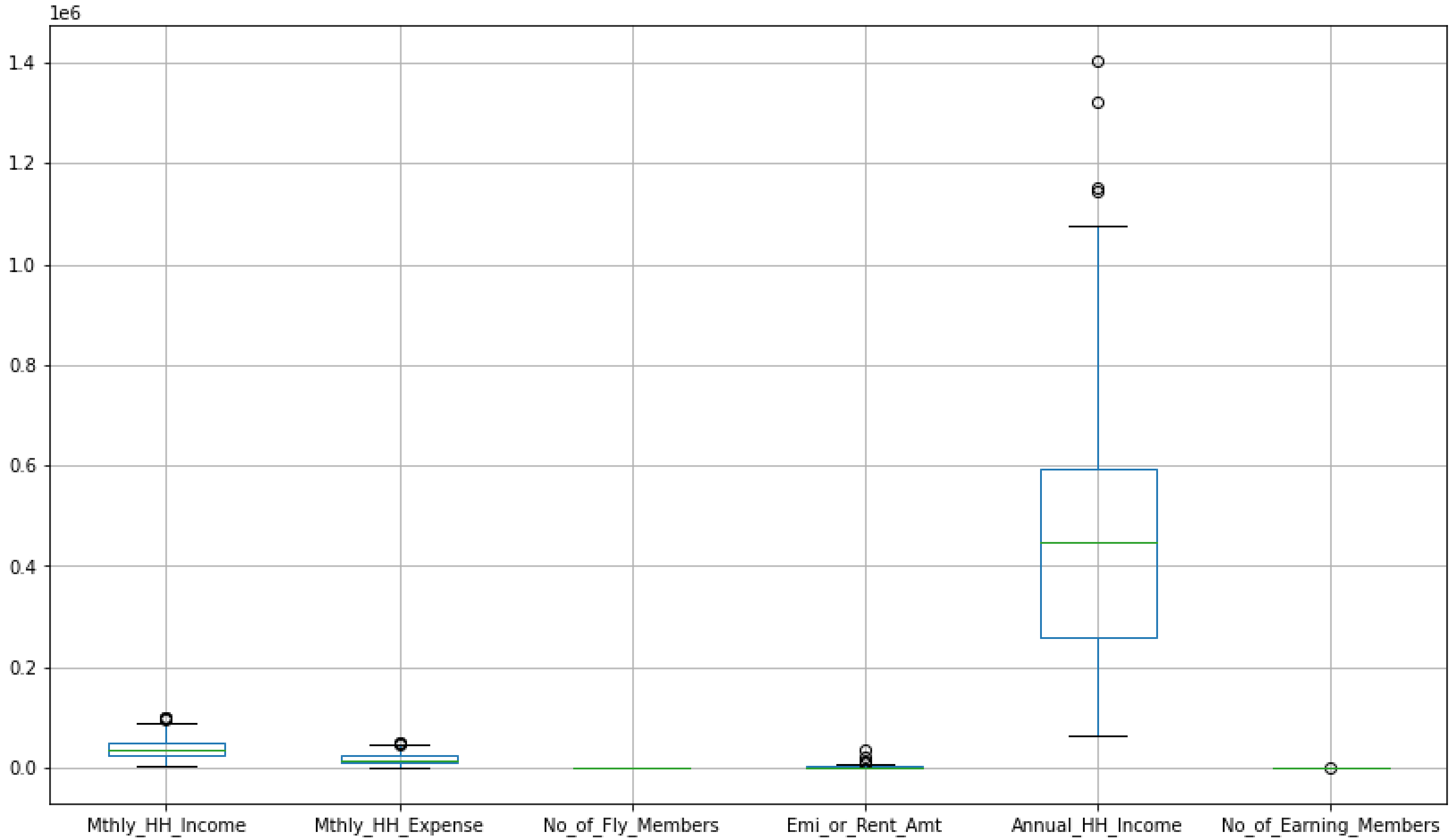


The point in the red marks are known as outliers.

In [23]:
```
## Dectecting Outliers Using Boxplot

df.boxplot(figsize=(14,8))
```

Out[23]:
```
<AxesSubplot:>
```



Annual HH Income has maximum outliers as compare to others.

13.Q-Q(quantile-quantile) Plots

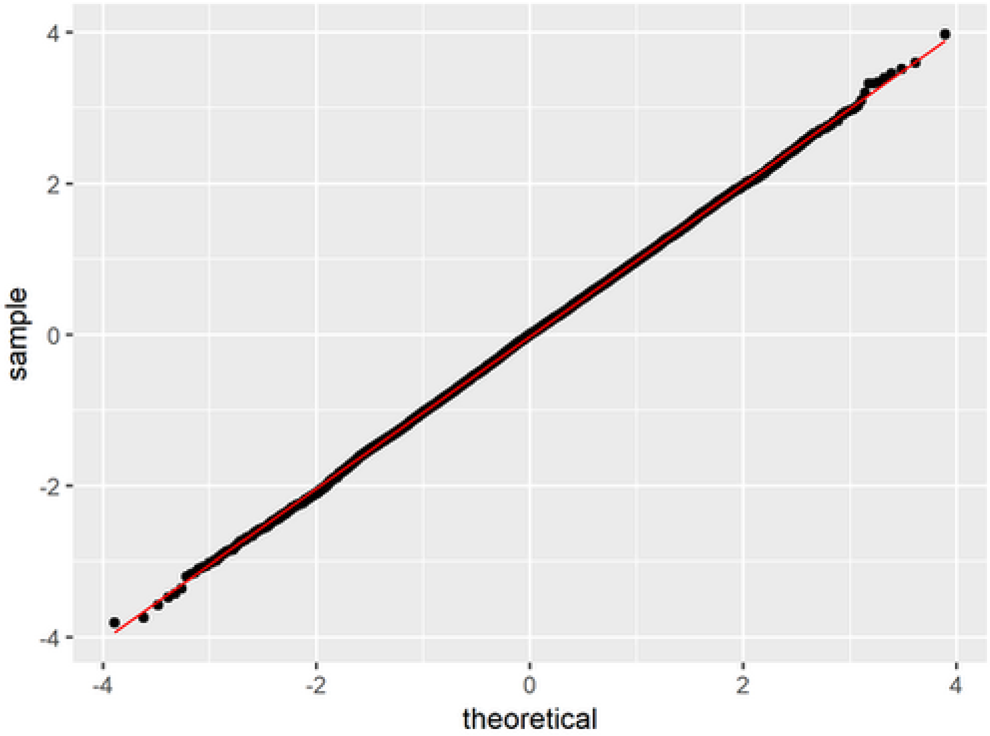
Before understanding QQ plots first understand what is a Quantile?

A quantile defines a particular part of a data set, i.e. a quantile determines how many values in a distribution are above or below a certain limit. Special quantiles are the quartile (quarter), the quintile (fifth), and percentiles (hundredth).

An example:

If we divide a distribution into four equal portions, we will speak of four quartiles. The first quartile includes all values that are smaller than a quarter of all values. In a graphical representation, it corresponds to 25% of the total area of distribution. The two lower quartiles comprise 50% of all distribution values. The interquartile range between the first and third quartile equals the range in which 50% of all values lie that are distributed around the mean.

In Statistics, A Q-Q(quantile-quantile) plot is a scatterplot created by plotting two sets of quantiles against one another. If both sets of quantiles came from the same distribution, we should see the points forming a line that’s roughly straight(y=x).



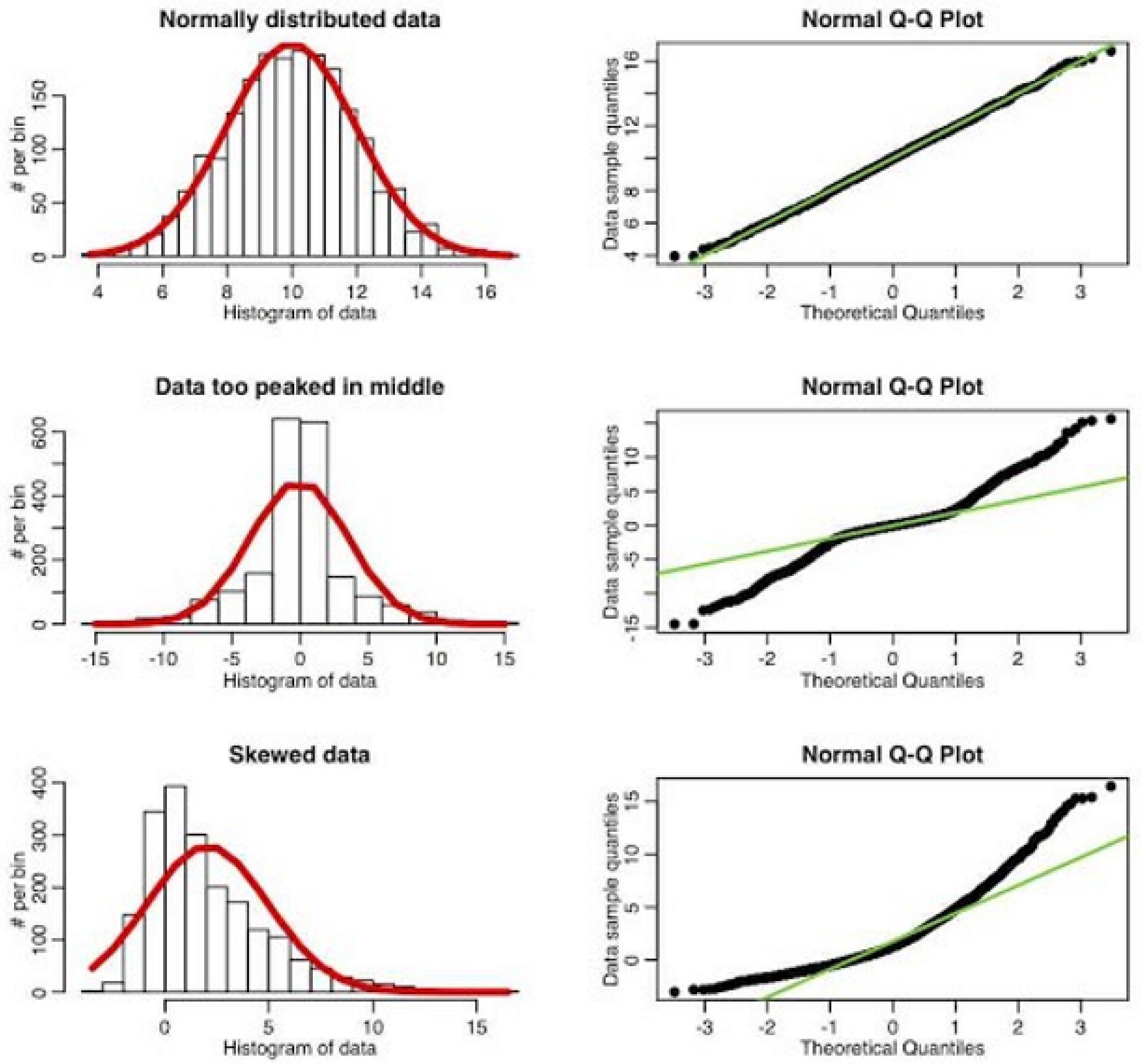
For example, the median is a quantile where 50% of the data fall below that point and 50% lie above it. The purpose of Q Q plots is to find out if two sets of data come from the same distribution. A 45-degree angle is plotted on the Q Q plot; if the two data sets come from a common distribution, the points will fall on that reference line.

It’s very important for you to know whether the distribution is normal or not so as to apply various statistical measures on the data and interpret it in much more human-understandable visualization and their Q-Q plot comes into the picture. The most fundamental question answered by the Q-Q plot is if the curve is Normally Distributed or not.

Normally distributed, but why?

The Q-Q plots are used to find the type of distribution for a random variable whether it is a Gaussian Distribution, Uniform Distribution, Exponential Distribution, or even Pareto Distribution, etc.

You can tell the type of distribution using the power of the Q-Q plot just by looking at the plot. In general, we are talking about Normal distributions only because we have a very beautiful concept of the 68–95–99.7 rule which perfectly fits into the normal distribution So we know how much of the data lies in the range of the first standard deviation, second standard deviation and third standard deviation from the mean. So knowing if a distribution is Normal opens up new doors for us to experiment with



Skewed Q-Q plots

Q-Q plots can find skewness(measure of asymmetry) of the distribution.

If the bottom end of the Q-Q plot deviates from the straight line but the upper end is not, then the distribution is Left skewed(Negatively skewed).

Now if upper end of the Q-Q plot deviates from the staright line and the lower is not, then the distribution is Right skewed(Positively skewed).

Tailed Q-Q plots

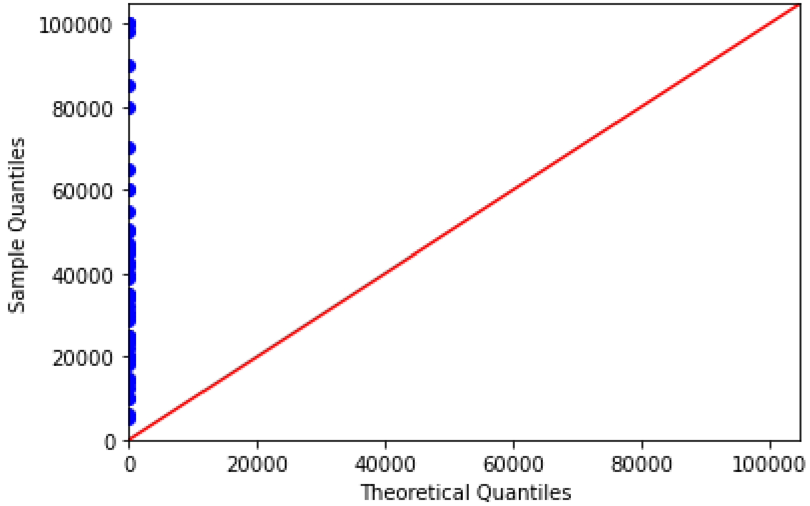
Q-Q plots can find Kurtosis(measure of tailedness) of the distribution.

The distribution with the fat tail will have both the ends of the Q-Q plot to deviate from the straight line and its centre follows the line, where as a thin tailed distribution will term Q-Q plot with very less or negligible deviation at the ends thus making it a perfect fit for normal distribution.

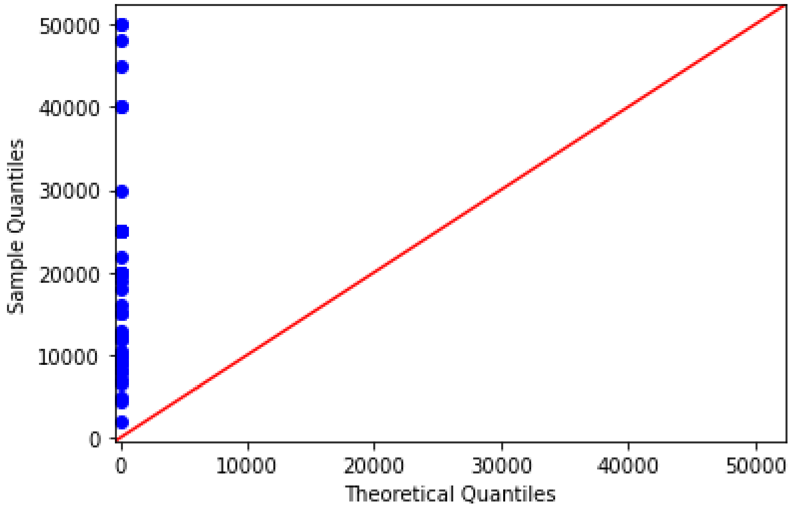
for more info - https://machinelearningmastery.in/2021/09/30/advanced-statistical-concepts-in-data-science/

In [24]:
```
import statsmodels.api as sm
```

In [25]:
```
sm.qqplot(df['Mthly_HH_Income'], line='45')
plt.show()
```

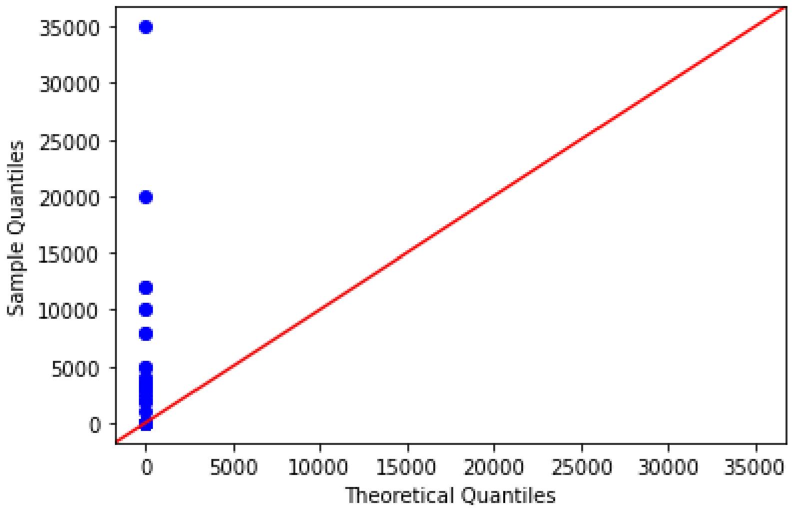


In [26]:
```
sm.qqplot(df['Mthly_HH_Expense'], line='45')
plt.show()
```



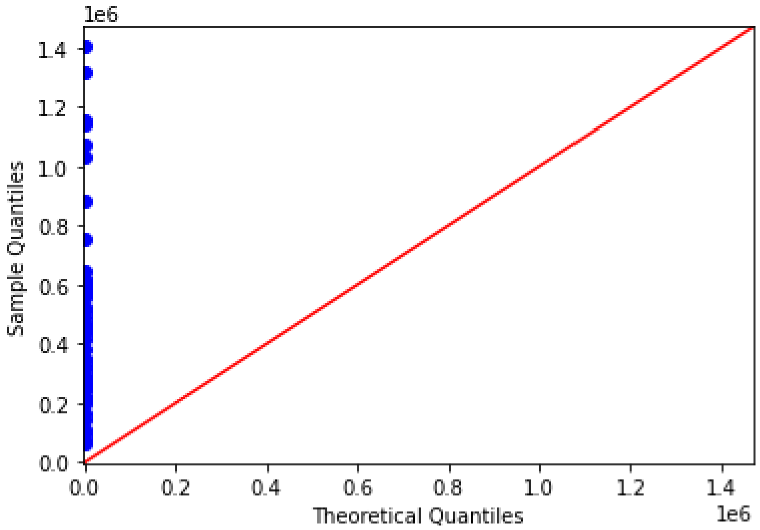
In [27]:

```
sm.qqplot(df['Emi_or_Rent_Amt'], line='45')
plt.show()
```

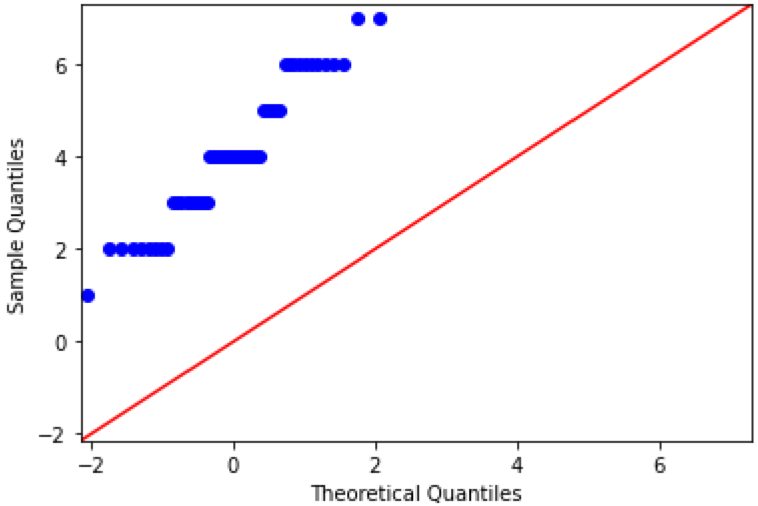


In [28]:

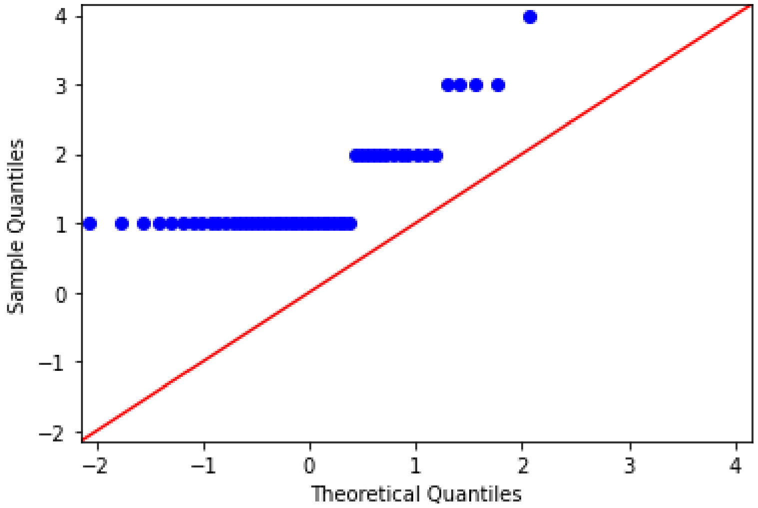
```
sm.qqplot(df['Annual_HH_Income'], line='45')
plt.show()
```



In [29]:
```
sm.qqplot(df['No_of_Fly_Members'], line='45')
plt.show()
```



In [30]:
```
sm.qqplot(df['No_of_Earning_Members'], line='45')
plt.show()
```



By Above plots we can say that Data is not normally Distributed.

END

In [ ]: