Descriptive Statistics

- ***Statistics***:
  - A way to get information from data.
  - the science that deals with the collection, classification, analysis, and interpretation of numerical facts or data, and that, by use of mathematical theories of probability, imposes order and regularity on aggregates of more or less disparate elements.
- ***Descriptive Statistics***:
  - Deals with methods of organizing, summarizing, and presenting data in a convenient and informative way.
- ***Measures of Central Tendency:***

- Mean: $\mu = \frac{\sum_{i=1}^{N} x_i}{N}$
- Median: The median is calculated by placing all the observations in ascending or descending order.
  - n is Odd: The observation that falls in the middle is the median. $(\frac{n+1}{2})^{th}$ item.
  - n is Even: Median is determined by averaging the two observations in the middle, $\frac{(\frac{n}{2})^{th} + (\frac{n}{2}+1)^{th}}{2}$ item.
- Mode: Observations that has the highest frequency.

Markdown

[1]:
```
import pandas as pd
```

[2]:
```
s1=pd.Series([2,1,0,4,3,15])
s1
```

[2]:
```
0     2
1     1
2     0
3     4
4     3
5    15
dtype: int64
```

[3]:
```
#Compute arithmetic mean
s1.mean()
```

[3]:
```
4.166666666666667
```

[4]:
```
s2=pd.Series([2,1,3,5,0])
s2
```

[4]:
```
0    2
1    1
2    3
3    5
4    0
dtype: int64
```

```
s2.sort_values()
```

```
4    0
```

```
[5]: s2.sort_values()

[5]: 4    0
     1    1
     0    2
     2    3
     3    5
     dtype: int64

[6]: #compute median
     s2.median()

[6]: 2.0

[7]: s3=pd.Series([3,0,2,3,4,1])

[8]: s3.value_counts()   #use value_counts() and then choose the top one - because value_counts() gives descending ordered list

[8]: 3    2
     0    1
     2    1
     4    1
     1    1
     dtype: int64

     s3.mode()
```

Day1_Stats Last Checkpoint: last month

Edit View Run Kernel Settings Help

Markdown

JupyterLab Python 3 (ipyk

[9]:
```
s3.mode()
```

[9]:
```
0    3
dtype: int64
```

[10]:
```
s4=pd.Series([3,0,2,3,4,1,5, 10, 5, 6])
s4.mode()
```

[10]:
```
0    3
1    5
dtype: int64
```

[ ]:

[12]:
```
df=pd.read_excel('ABC attrition data.xlsx',sheet_name='Employee data class training')
df.head()
```

```
C:\Users\chetana.hegde\Anaconda3\lib\site-packages\openpyxl\worksheet\_read_only.py:79: UserWarning: Web Extension extension is not supported and w
removed
  for idx, row in parser.parse():
```

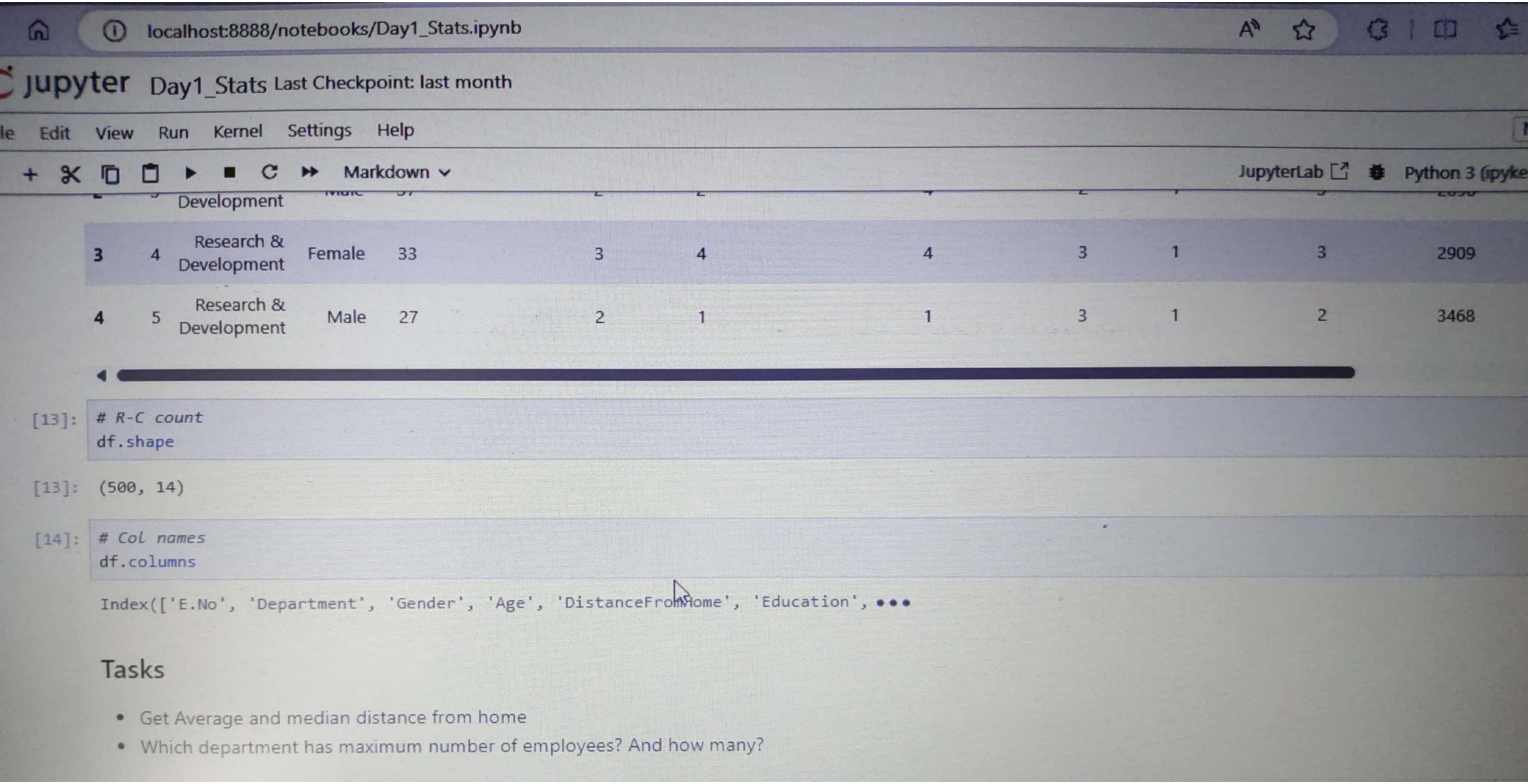
[12]:

	E.No	Department	Gender	Age	DistanceFromHome	Education	EnvironmentSatisfaction	JobInvolvement	JobLevel	JobSatisfaction	MonthlyIncome	NumCo
0	1	Sales	Female	41	1	2	2	3	2	4	5993	
1	2	Research & Development	Male	49	8	1	3	2	2	2	5130	
2	3	Research & Development	Male	37	2	2	4	2	1	3	2090	
		Research &										

Search

Activate Windo

Scanned with CamScanner



localhost:8888/notebooks/Day1_Stats.ipynb

jupyter Day1_Stats Last Checkpoint: last month

Edit View Run Kernel Settings Help

Markdown

JupyterLab Python 3 (ipykernel)

- Which department has maximum number of employees? And how many?

[15]:
```
# Your answer
df['DistanceFromHome'].mean()
```

[15]:
```
9.12
```

[16]:
```
df['DistanceFromHome'].median()
```

[16]:
```
6.0
```

[17]:
```
df['Department'].value_counts()
```

[17]:
```
Research & Development    333
Sales                     153
Human Resources            14
Name: Department, dtype: int64
```

[18]:
```
df['Department'].mode()
```

[18]:
```
0    Research & Development
Name: Department, dtype: object
```

[19]:
```
df.describe()  #statistical parameters for the numerical columns in the data
```

[19]:

	E.No	Age	DistanceFromHome	Education	EnvironmentSatisfaction	JobInvolvement	JobLevel	JobSatisfaction	MonthlyIncome	NumCompaniesW
count	500.000000	500.000000	500.000000	500.000000	500.000000	500.000000	500.00000	500.000000	500.000000	500.
mean	250.500000	36.896000	9.120000	2.888000	2.678000	2.730000	2.09600	2.804000	6598.644000	2.

File Edit View Run Kernel Settings Help

Markdown

Measures of Variability/Dispersion

- **Range** = Largest observation in the data - Smallest observation in the data
- **Variance** : How far each observation is from mean. These differences from the mean are called deviations.
- The average squared deviation from the mean is called the variance. $\sigma^2 = \frac{\sum_{i=1}^{N}(x_i-\mu)^2}{N}$
- **Standard Deviation**: Square root of variance is called as SD

[20]:
```
# Get the average, median, min, max, range, var and sd for the monthly income
print('Mean= ', df.MonthlyIncome.mean())
print('Median= ', df.MonthlyIncome.median())
print('Minimum= ', df.MonthlyIncome.min())
print('Maximum= ', df.MonthlyIncome.max())
print('Range= ', df.MonthlyIncome.max() - df.MonthlyIncome.min())
print('variance= ', df.MonthlyIncome.var())
print('Standard Deviation= ', df.MonthlyIncome.std())
```

```
Mean=  6598.644
Median=  4952.0
Minimum=  1102
Maximum=  19999
Range=  18897
variance=  23180200.710685384
Standard Deviation=  4814.582090969619
```

Let's understand SD more conceptually

Search

```
Standard Deviation=  4814.582090969619
```

Let's understand SD more conceptually

Standard deviation is also used as a measure or risk

You are trying to pick stock for investing in the equity market. Stock A has an annual return of 15%, with a std deviation of 30%. Stock B has an annual return of 12%, with a std deviation of 8%.

If you were risk averse, which would you choose?

Chebyshev's Inequality

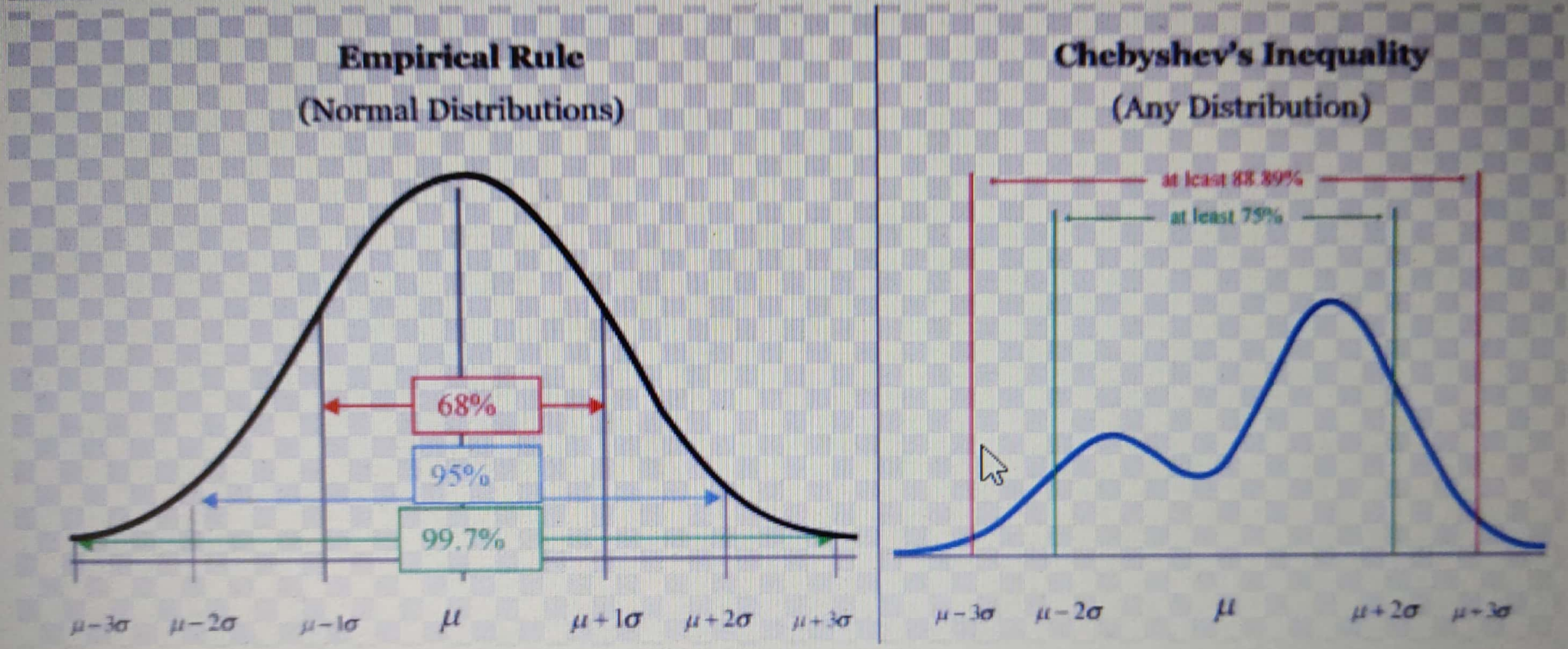
For any data set, it can be proved mathematically that

- Atleast 75% of all data points will lie within 2 standard deviations of the mean,
- and atleast 89% within 3 standard deviations

- and atleast 89% within 3 standard deviations

[21]:
```
from IPython.display import Image
Image(filename='ChebyshevInequality.png')
```

[21]:



Example:

Suppose we have a data series, with a min of 200, a max of 1500, a mean of 600, and a standard deviation of 80.

Then, atleast 75% of all the data points in the series will be within the range:

$$(600-2*80, 600+2*80) = (440, 760)$$

File Edit View Run Kernel Settings Help

Markdown

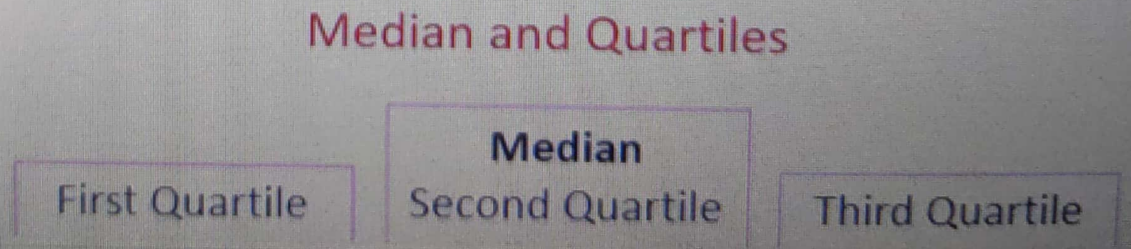
$$(600–2 * 80, 600 + 2 * 80) = (440, 760)$$

And, atleast 89% of all data points will be within the range:

$$(600–3 * 80, 600 + 3 * 80) = (360, 840)$$

Quantiles, Quartiles and Percentiles of data

- Quantile is a certain portion of the given data.
- Quartile is one portion among four equal divisions of the data.
  - That is, we need to divide the data in to 4 equal parts.
  - 1st Quartile (Q1) includes first 25% of the data (lowest value onwards).
  - 2nd Quartile (Q2) defines 50% of the data (this also equal to median)
  - 3rd Quartile (Q3) defines 75% of the data
- Percentile is one portion among 100 equal divisions of the data.
  - We can have P1 to P99.
  - P25 is equal to Q1
  - P50 is equal to Q2, which is same as median
  - P75 is equal to Q3

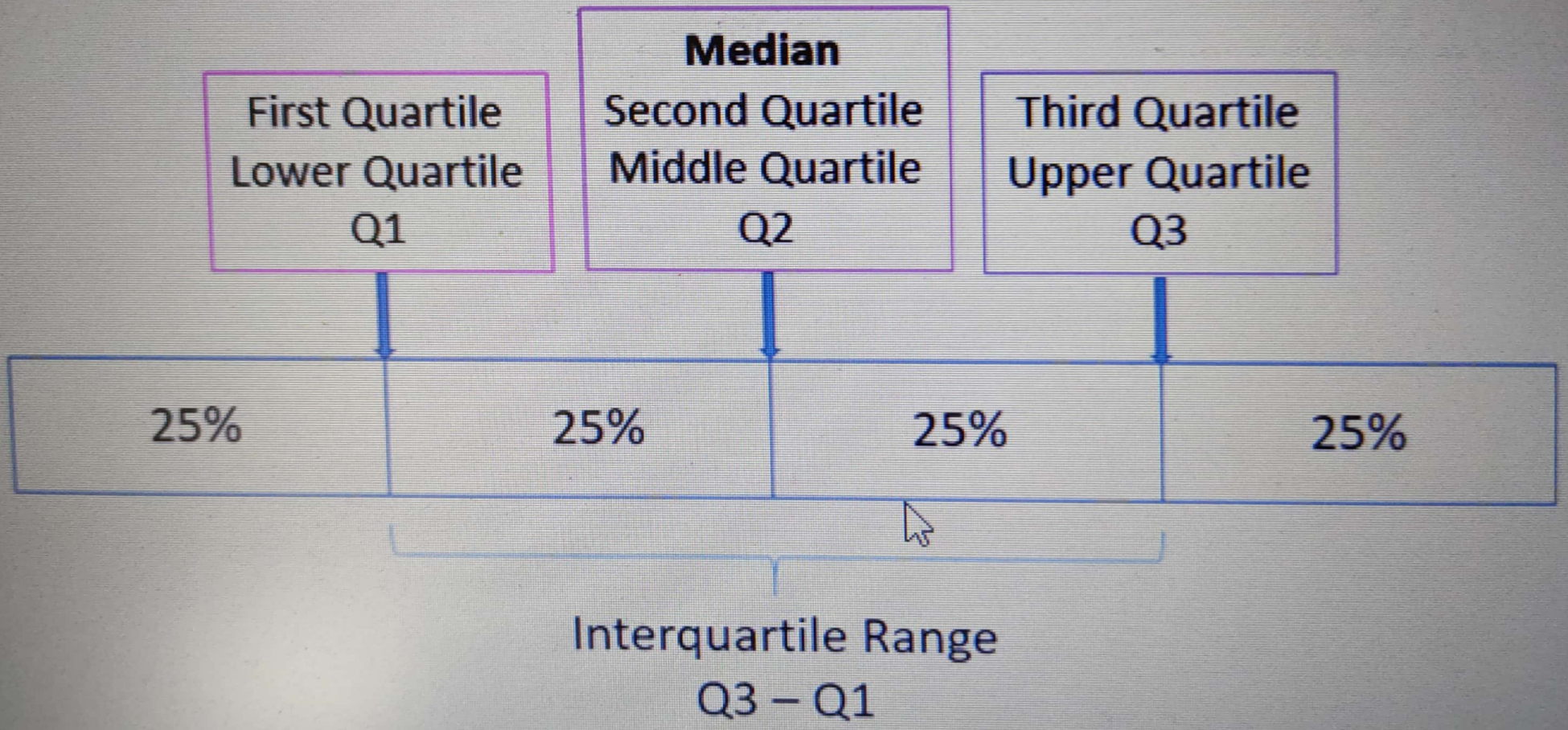


88°F Haze

Markdown

- P75 is equal to Q3

Median and Quartiles



- What is the 25th, 50th and 75th percentile of employee age?

22]:
```
print(df['Age'].quantile(0.25))   # This is Q1
print(df['Age'].quantile(0.5))    # This is Q2 and Median
print(df['Age'].quantile(0.75))   # This is Q3
```

```
30.0
36.0
43.0
```

Inference:

- There are 500 employees in the dataset. 25% (Q1) of the employees (which means, 25% of 500 =125 employees) are below (or equal) the age of 30.
- 50% of the employees = 250 employees are within 36 years.
- 75% of the employees = 375 employees are within 43 years of age.
- ***This clearly indicates that the organization has a young pool of employees***

IQR:

- The distance from the lower quartile to the upper quartile is known as the interquartile range (IQR).
- IQR = Q3-Q1

Here Q3 is 75th percentile and Q1 is 25th percentile

localhost:8888/notebooks/Day1_Stats.ipynb

jupyter Day1_Stats Last Checkpoint: last month

File Edit View Run Kernel Settings Help

Markdown

JupyterLab Python 3 (ipykernel)

[23]:
```
# IQR of Income
print('IQR of income = ', df.MonthlyIncome.quantile(0.75) - df.MonthlyIncome.quantile(0.25))
```

```
IQR of income =  5841.5
```

[24]:
```
print('Q1 of income = ', df.MonthlyIncome.quantile(0.25))
print('Q3 of income = ', df.MonthlyIncome.quantile(0.75) )
```

```
Q1 of income =  2900.25
Q3 of income =  8741.75
```

Inference:

- 50% of the employees (between Q1 and Q3 there is 50% of the data) = 250 people have salary in the range of Rs. 2900/- to Rs. 8742/- (assuming the income is in rupees)

[ ]:

Shape measures

1. Skewness

- Skewness is a measure of symmetry, or more precisely, the lack of symmetry.
- A distribution, or data set, is symmetric if it looks like a bell-shaped curve (normal/Gaussian distributio
- If the skewness is between
  - -0.5 and 0.5, the data are nearly symmetrical.
  - -1 and -0.5 then the distribution is slightly negatively skewed.
  - 0.5 and 1, the distribution is slightly positive skewed.
- If the skewness is lower than -1 or greater than 1, the data are extremely skewed.

Search

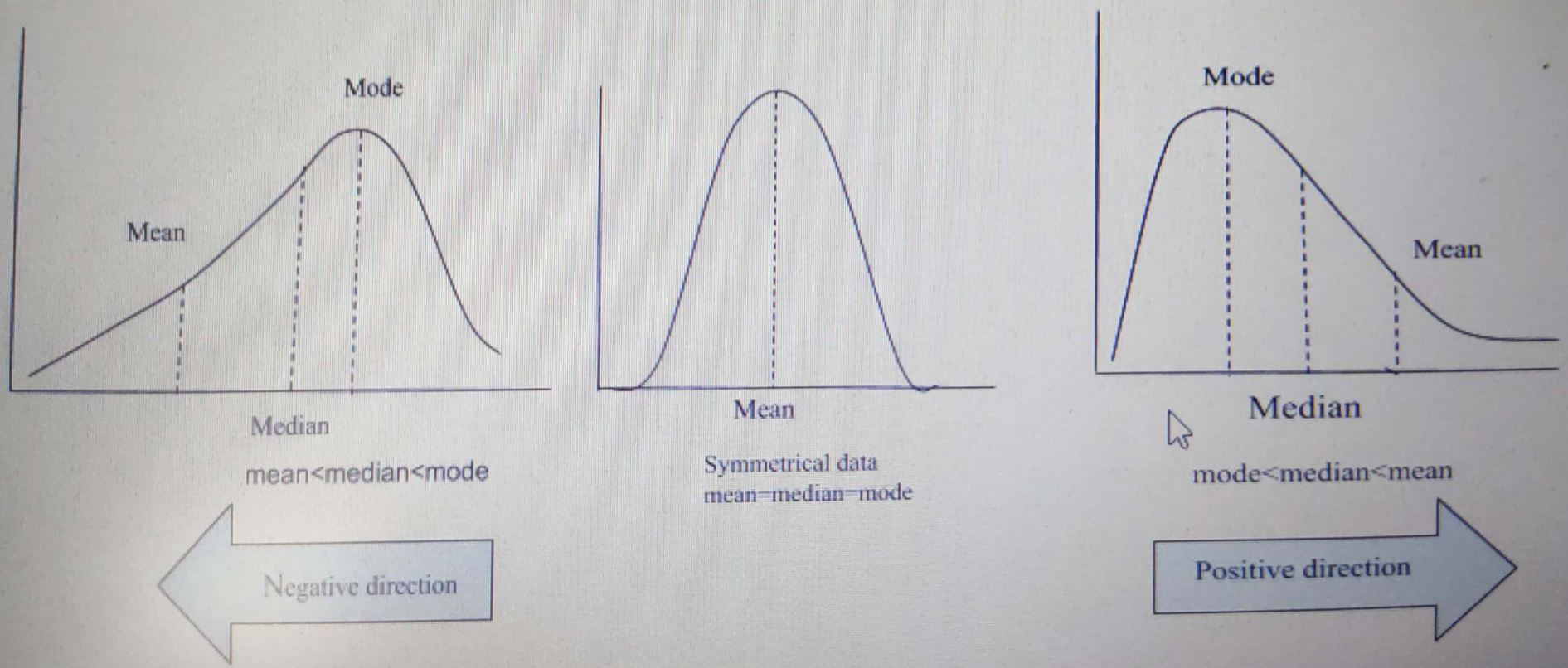
Edit View Run Kernel Settings Help

Markdown

- Skewness is a measure of symmetry, or more precisely, the lack of symmetry.
- A distribution, or data set, is symmetric if it looks like a bell-shaped curve (normal/Gaussian distribution)
- If the skewness is between
  - -0.5 and 0.5, the data are nearly symmetrical.
  - -1 and -0.5 then the distribution is slightly negatively skewed.
  - 0.5 and 1, the distribution is slightly positive skewed.
- If the skewness is lower than -1 or greater than 1, the data are extremely skewed.

[26]:
```
Image(filename='skewness.png')
```

[26]:

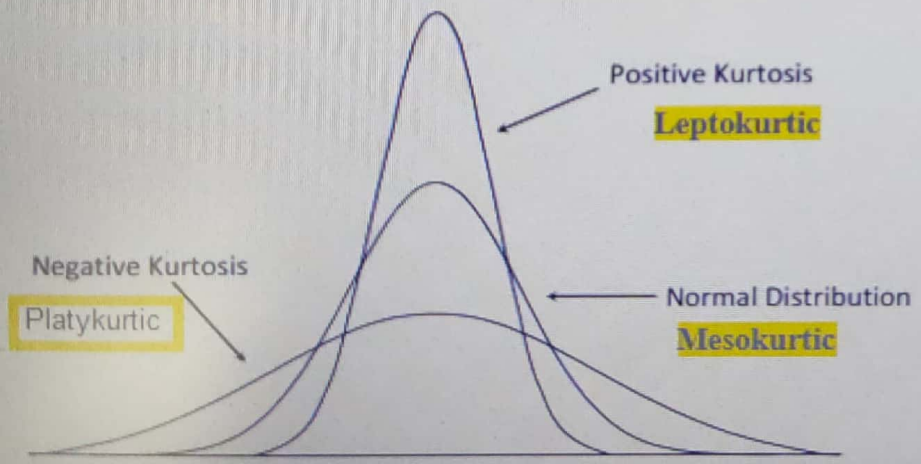


2. Kurtosis

- Kurtosis is a measure of whether the data are heavy-tailed or light-tailed relative to a normal distribution.
- That is, data sets with high kurtosis tend to have heavy tails, or outliers. Data sets with low kurtosis tend to have light tails, or lack of outliers.
- Kurtosis for normal distribution (to be more specific, standard normal distribution) is 3.
- Leptokurtic distributions have kurtosis >3
- Platykurtic distributions have kurtosis <3

[27]:
```
Image(filename='kurtosis.png')
```

[27]:



```
import pandas as pd
df=pd.read_excel('ABC attrition data.xlsx',sheet_name='Employee data class training')
df.head()
```

```
C:\Users\chetana.hegde\Anaconda3\lib\site-packages\openpyxl\worksheet\_read_only.py:79: UserWarning: Web Extension extension is not supported and
```

```
# Skewness and Kurtosis of Age and Monthly Income
print('Skewness of Age =', df.Age.skew())
print('Skewness of MI =', df.MonthlyIncome.skew())
print('Kurtosis of Age =', df.Age.kurtosis())
print('Kurtosis of MI =', df.MonthlyIncome.kurtosis())
```

```
Skewness of Age = 0.4390695056460207
Skewness of MI = 1.326524112397537
Kurtosis of Age = -0.382279412791124
Kurtosis of MI = 0.8182437134388572
```

Inferences:

- Age is slightly positively skewed
- Monthly Income is Highly positively skewed - There are few people with very high salary compared to others.
- Both Age and Monthly Income are platykurtic. Which means, both are highly deviated fromt the mean, i.e. high standard deviation

Covariance And Correlation

- Covariance and correlation both primarily assess the relationship between variables.
- Using covariance, we can only gauge the direction of the relationship (whether the variables tend to move in tandem or show an inverse relationship). However, it does not indicate the strength of the relationship, nor the dependency between the variables.

- Using covariance, we can only gauge the direction of the relationship (whether the variables tend to move in tandem or show an inverse relationship). However, it does not indicate the strength of the relationship, nor the dependency between the variables.
- On the other hand, correlation measures the strength of the relationship between variables. Correlation is the scaled measure of covariance. It is dimensionless. In other words, the correlation coefficient is always a pure value and not measured in any units.

$$\text{Cov}(X, Y) = \frac{\sum(X_i - \bar{X})(Y_j - \bar{Y})}{n}$$

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

$$\rho(X, Y) = \frac{\text{Cov}(X, Y)}{\sigma_X \sigma_Y}$$

We can also write the correlation coefficient (r or ρ) as -

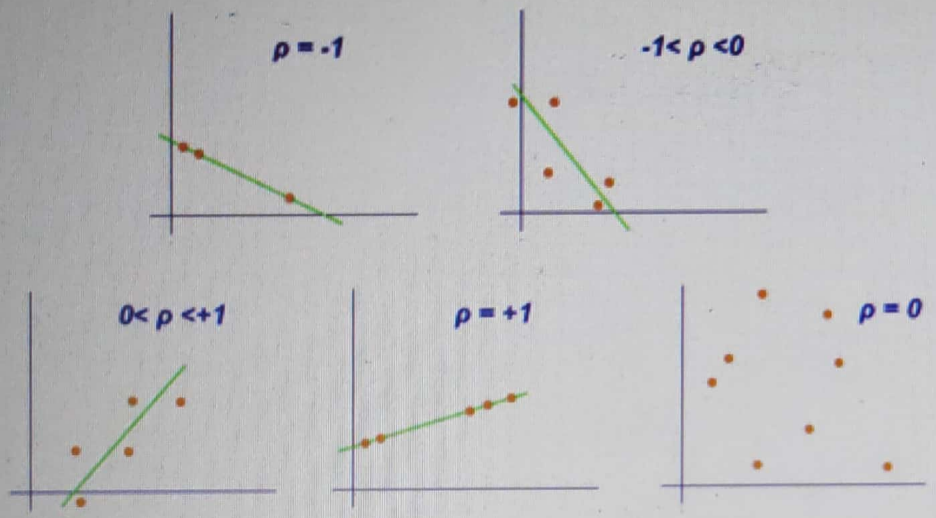
The value of correlation coefficient ranges from -1 to +1.

$\rho = -1$ $-1 < \rho < 0$

File Edit View Run Kernel Settings Help

Markdown

The value of correlation coefficient ranges from -1 to +1.



Example:

Consider the data related to age of the people and their glucose level. Find the correlation coefficient between these two variables.

[30]:
```
import numpy as np
Age = np.array([43, 21, 25, 42, 57, 59])
GlucoseLevel = np.array([99, 65, 79, 75, 87, 81])
```

[31]:
```
R=np.corrcoef(Age, GlucoseLevel)
print(R)
```

```
[[1.        0.5298089]
 [0.5298089 1.       ]]
```

Search

```
GlucoseLevel = np.array([99, 65, 79, 75, 87, 81])
```

[31]:
```
R=np.corrcoef(Age, GlucoseLevel)
print(R)
```

```
[[1.        0.5298089]
 [0.5298089 1.       ]]
```

[32]:
```
print('Correlation coefficient between Age and Glucose level is ', R[0][1])
```

```
Correlation coefficient between Age and Glucose level is  0.5298089018901743
```

Correlation between Age and MonthlyIncome in the employee dataset

[33]:
```
df['MonthlyIncome'].corr(df['Age'])
```

[33]:
```
0.4911856396084064
```

Inference: Age and Monthly Income are positively correlated. However, we cannot say that as the age increases, the income also increases. Because, there is no strong bond between them. If corr coeff was say, around 0.9, then we would have concluded that as the age increases, income increases. However, in our data, there are chances that young people having more salary and vice-versa.

[ ]:

[ ]: