INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

Descriptive Statistics

Agenda

- Classification of Data
- Understanding the variables
  - Numerical Variable
  - Categorical Variable
- Descriptive Statistics
  - Measure of Central Tendency
  - Distribution of the data
  - Measure of Dispersion
  - Skewness and Kurtosis
  - Covariance and Correlation
- Case Study

Exploratory Data Analysis (EDA)

- EDA is a process of analyzing the datasets to summarize their main features using numerical and visual methods

- In this session, we shall cover the numerical methods to analyze the data

- The numerical methods includes descriptive statistics

The Data

What is Data?

Data is units of information in structured or unstructured format.

Examples:

- Collection of relevant tweets
- Records of yield in a farm over a period of time
- Records of stock price every minute
- Records of performance of a sports person

Classification of Data

In general, there are four types of data. They are:

- Time Series Data
- Cross sectional data
- Pooled Data
- Panel Data

Time series data

- Time series data is the set of observations on a variable at different time points
- The data may be collected daily, weekly, monthly, or annually

Date	Prev.Close	Open.Price	High.Price	Low.Price
01-Jan-1996	408	407	407.9	405
02-Jan-1996	407.9	407	409	406.25
03-Jan-1996	406.25	409	409	409
04-Jan-1996	409	405	407	405
05-Jan-1996	406.3	401.5	401.5	401.5
08-Jan-1996	401.5	401.5	405	402
09-Jan-1996	404	404	400	395
10-Jan-1996	399.5	399.5	397	395
11-Jan-1996	396	399	405	396
12-Jan-1996	405	403	403	400

Sample data of daily stock price of a company

Cross-sectional data

Cross-sectional data are set of observations on two or more variables at the same time point

Produce in the 2019 in (quintals)			
Region	**Rice**	**Wheat**	**Maize**
Region 1	28.76	86.66	8.23
Region 2	84.03	65.45	41.94
Region 3	71.75	28.99	38.43
Region 4	14	68.87	7.04
Region 5	43.97	54.27	86.51

Sample data of a farm produce in five regions in the year 2019

Pooled data

Pooled data is combination of both time series data and cross sectional data.

	Produce in the 2011		Produce in the 2012	
Appliance	Production	Profit (in '000 Rs)	Production	Profit (in '000 Rs)
Radio	280	13	121	8
Television	840	645	1456	5192
Washing Machine	775	2899	152	5706
Computer	876	6887	1005	6349
Air Conditioner	439	5427	328	8095

Sample data of the number of appliances produced in an industry for two years

Panel data

- Panel data is type of pooled data
- The same cross-sectional unit is surveyed over time
- Also known as longitudinal or micro-panel data

Appliance	Year	Production	Profit (in '000 Rs)
Radio	2010	280	13
Television	2010	840	645
Washing Machine	2010	775	2899
Computer	2010	876	6887
Air Conditioner	2010	439	5427
Radio	2011	234	123
Television	2011	835	645
Washing Machine	2011	564	2899
Computer	2011	874	6887
Air Conditioner	2011	435	5427

Sample data of the number of appliances produced and profits earned by a company over a period of two years

Reading Data from Sources

Read the data

- Data is available in different sources
- In python we can read data from different sources
- We shall see how to read data from csv, xlsx, tsv, json, html file formats, zip file and read data from an URL

Read a csv file

```
# import the required library
import pandas as pd

# store the data in a variable
df_mpg_csv = pd.read_csv('mpg.csv')
```

```
# display head of the data
df_mpg_csv.head()
```

	mpg	cylinders	displacement	horsepower	performance	weight	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	Good	3504	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	Excellent	3693	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	Average	3436	11.0	70	usa	plymouth satellite
3	16.0	8	304.0	150.0	Excellent	3433	12.0	70	usa	amc rebel sst
4	17.0	8	302.0	140.0	Excellent	3449	10.5	70	usa	ford torino

Read an xlsx file

```
# import the required library
import pandas as pd

# store the data in a variable
df_mpg_xlsx = pd.read_excel('mpg.xlsx')
```

```
# display head of the data
df_mpg_xlsx.head()
```

	mpg	cylinders	displacement	horsepower	performance	weight	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	Good	3504	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	Excellent	3693	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	Average	3436	11.0	70	usa	plymouth satellite
3	16.0	8	304.0	150.0	Excellent	3433	12.0	70	usa	amc rebel sst
4	17.0	8	302.0	140.0	Excellent	3449	10.5	70	usa	ford torino

Read a tsv file

```
# import the required library
import pandas as pd

# store the data in a variable
df_mpg_tsv = pd.read_csv('mpg.tsv', sep = '\t')
```

```
# display head of the data
df_mpg_tsv.head()
```

	mpg	cylinders	displacement	horsepower	performance	weight	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	Good	3504	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	Excellent	3693	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	Average	3436	11.0	70	usa	plymouth satellite
3	16.0	8	304.0	150.0	Excellent	3433	12.0	70	usa	amc rebel sst
4	17.0	8	302.0	140.0	Excellent	3449	10.5	70	usa	ford torino

Read a json file

```
# import the required library
import pandas as pd

# store the data in a variable
df_mpg_json = pd.read_json('mpg.json')
```

```
# display head of the data
df_mpg_json.head()
```

	mpg	cylinders	displacement	horsepower	performance	weight	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	Good	3504	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	Excellent	3693	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	Average	3436	11.0	70	usa	plymouth satellite
3	16.0	8	304.0	150.0	Excellent	3433	12.0	70	usa	amc rebel sst
4	17.0	8	302.0	140.0	Excellent	3449	10.5	70	usa	ford torino

Read a html file

```
# read the html file
# indicates that the row to use to create the column names
# index_col indicates that the column to use to create the column names
# the following code returns a list
df_mpg_html = pd.read_html('mpg.html', header=1, index_col=0)

# we extract the first element to the list which is our requiured DataFrame
df_mpg_html = df_mpg_html[0]

# display first three rows
df_mpg_html.head(3)
```

	mpg	cylinders	displacement	horsepower	performance	weight	acceleration	model_year	origin	name
1										
2	18.0	8	307.0	130.0	Good	3504	12.0	70	usa	chevrolet chevelle malibu
3	15.0	8	350.0	165.0	Excellent	3693	11.5	70	usa	buick skylark 320
4	18.0	8	318.0	150.0	Average	3436	11.0	70	usa	plymouth satellite

Read data from an url

We use the pima indians diabetes dataset

Store the url. This url contains the csv file of the data

Read the csv

```
# import pandas
import pandas
# store the url
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.csv"
# create a list of column names
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
# read the data
data = pandas.read_csv(url, names=names)
# print the data
data.head()
```

	preg	plas	pres	skin	test	mass	pedi	age	class
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

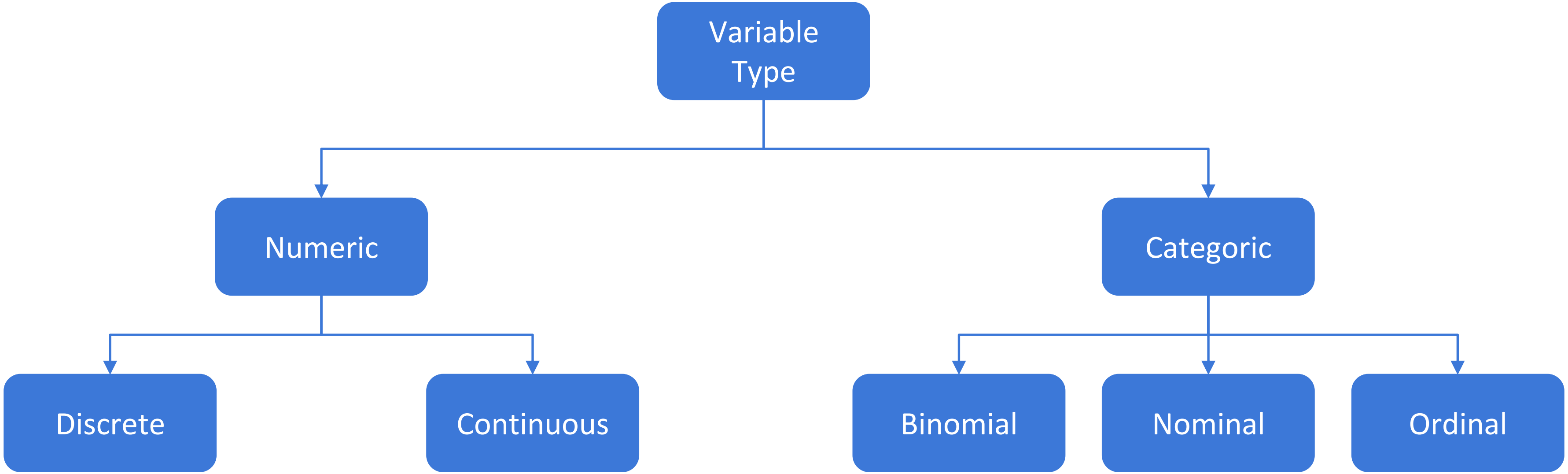
Summary

File Type	Function
CSV File	read_csv(file_address)
TSV File	read_csv(file_address, sep = ‘\t’)
JSON File	read_json(file_address)
HTML File	read_html(file_address)
URL	read_csv(url)

Understanding the variable

Understand the variable

To understand the descriptive statistics, we need to first understand the type of each variable or column



Numerical Variable

- A variable which takes a numeric value is called a numeric variable
- Also known as quantitative variable
- A discrete numeric variable is a random variable which takes discrete values, i.e. values from the set of whole numbers only. It can take countably finite values

Examples:

1. Number of cars passing by a toll-gate every one minute
2. Number of defective items in a box

Numerical Variable

A continuous numeric variable is a variable which can have infinite number of values within a range

Examples:

1. The amount of rainfall in millimeters
2. Price of a stock

Categorical Variable

- Categorical variable has two or more levels
- Also known as qualitative variable

Examples:

1. Colors: red, orange, yellow
2. Gender: male, female
3. Economic status: low, medium, high

Categorical Variable

Nominal Data:

- Nominal data has no order and has two or more than two categories
- It represents discrete units and are used to label variables
- Example: Housing type - Apartment, Bungalow, Penthouse

Binary Data:

- Binary data has no order and has strictly two categories
- Also known as dichotomous variable
- Example: Presence or absence of a disease

Categorical Variable

Ordinal Data:

- Ordinal data is ordered nominal data
- It represents discrete and ordered units
- Example: Education - Primary, Secondary, High School, College and University which are ordered

PLEASE!! NOTE

- At times data may appear to be numeric but is actually categorical
- Consider the adjoining table. The 'Gender' column has value 1 and 0
- The numeric values used to represent a categorical variable cannot be used for numerical computation

Name	Gender	Age
Ria	1	45
Arya	1	23
Sam	0	34
Joe	0	54
Harry	0	12
Sarah	1	43

Data Analysis

- It is used to only describe the sample or summarize information about the sample
- Also known as Descriptive Statistics

- It is used to make inferences and generalizations about the broader population
- Also know an Inferential Statistics

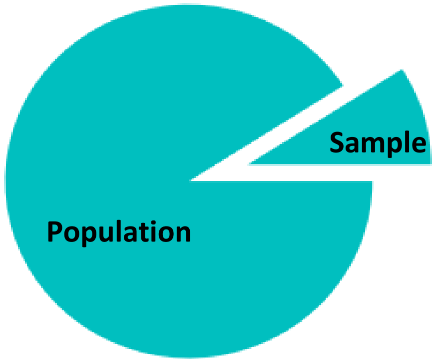
Descriptive Statistics

What is descriptive statistics?

- Descriptive statistics is the term given to the analysis of the data that helps describe, visualize or summarize the data
- Helps exhibit the patterns in the data
- However, descriptive statistics only help in understanding the data and not make conclusions regarding any hypothesis
- We can not generalise the facts, they are confined only to the data at hand (that may be the sample)

Population & Sample

- Population is a collection of all the individuals or objects
- Sample is a subset of the population which is a representative of the population



Example:

Suppose a company producing electric bulbs wants to know the average life of a bulb. If the details on average life of all bulbs are available, then this information is regarded as the population.

It would be challenging for the company to test each and every bulb produced. In such a scenario, the company would draw a sample from the produced bulbs to test.

Descriptive statistics

The major characteristics we observe in each variable of the data set are:

1. Measures of central tendency
2. Distribution of the data
3. Measures of dispersion

The above characteristics help us in understanding the data and are part of the exploratory data analysis

Measures of Central Tendency

Measures of central tendency

- A measure of central tendency is a single value that identifies the central position of the data
- It includes mean, median, mode and partition values

Mean

- Mean is a central tendency of the data
- It is defined as the sum of all observations divided by the number of observations:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{N}$$

where N is the total number of observations

- The whole data is spread out around the mean

Mean

Merits:

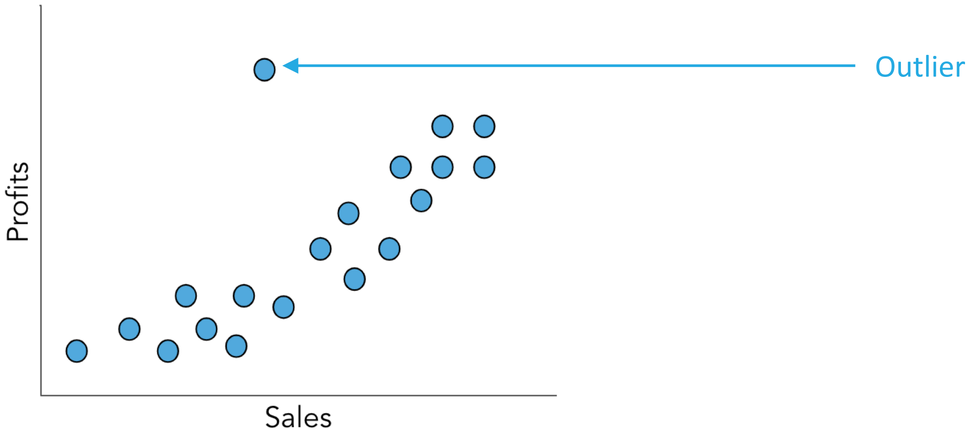
- Based on all observation

Demerits:

- Affected by extreme observations (outliers)
- Arithmetically it is not possible to obtain the mean of a data with missing values
- Mean can not be calculated for categorical data

Outlier

- An outlier is a value that behaves differently than other observations
- It does not not follow the usual pattern



Obtain the mean

Each column of a DataFrame is a pandas Series.

Let us create a series and find its mean.

```
prices = pd.Series([0,0,35,40,10,54,87,12,95,64,56,4,45,76,34,56,
    87,34,56,32,56,48,89,42,65,100,99,98,100,96,99])
```

```
# obtain the mean of 'prices'
prices.mean()
```

```
57.064516129032256
```

Obtain the mean

Arithmetically, it is not possible to compute mean of missing data. But, python ignores the missing values and calculates the mean of the available data in the series

Introduce missing values in the previous series

```
# let us introduce some missing values in the series
# the remaining values are same as the series 'prices'
prices_missing = pd.Series([0,0,35,40,10,54,87,np.NaN,12,95,64,56,4,45,76,34,np.NaN,56,
               87,34,56,32,56,48,np.NaN,89,42,65,100,99,98,100,96,99])
```

```
# obtain the mean of prices_missing
prices_missing.mean()
```

```
57.064516129032256
```

PLEASE!! NOTE

Mean of series with missing values

We notice that the means obtained are the same

This is because the function drops the missing values before calculating the mean value

```
# obtain the mean of 'prices'
prices.mean()
```

```
57.064516129032256
```

```
# obtain the mean of price_missing
prices_missing.mean()
```

```
57.064516129032256
```

Trimmed mean

- Arithmetic mean obtained by ignoring lowest and highest α% observations is called as the α% - trimmed mean
- Using a trimmed mean, it is possible to eliminate the influence of outliers
- Note the observations are arranged either in increasing or decreasing order
- Also called as truncated mean

Example: The trimmed means are used in scoring the performance of an athlete during olympic games, to minimize the extreme scoring from possible biased judgements.

Obtain the trimmed mean

While working in python, we can directly use the built in function, trim_mean(), available in the library scipy.

Import the required library

```
import scipy
from scipy import stats

# obtain a 20% trimmed mean
scipy.stats.trim_mean(prices, proportiontocut =0.20)
```

```
58.89473684210526
```

Parameter ‘proportiontocut’ denotes the α% values to exclude. It takes value between 0 to 0.51

Median

- Median is the middle most observation in the data when it is arranged either in ascending or descending order of their values
- It divides the data into two equal parts. Thus, it is a positional average
- Median will be the middle term, if the number of observations is odd
- Median will be average of middle two terms, if number of observations is even

Median

Merits:

- It is not affected by extreme values

Demerits:

- It is not based on all observations

Obtain the median

```
prices.median()
```

```
56.0
```

Interpretation: The median value is 56. It implies that the value 56 divides the data in two equal parts. There are 50% of the observations above and below this price.

Using mean and median

- For a numeric variable, the missing value is replace by the mean if there are no outliers present

- In case if outliers are present the missing values are replaced by the median

- For symmetric data median = mean, so either of the value can be used

Mode

- Mode of the data is the value which has the highest frequency
- In other words, it is most repeated observation
- A set of observations having two modes is said to be bimodal
- A set of observations may have more than two modes. Such data is said to be multimodal

Mode

Merits:

- It is applicable to both numeric and categorical data
- It is not affected by extreme observations

Demerits:

- It is not based on all observations

Obtain the mode

```
prices.mode()

0      56
dtype: int64
```

0 is the index

Notice that the output is a Series

56 is the mode value

Interpretation: The mode value is 56. It implies that the value 56 occurs the most number of times
Also note that there is only one mode. The data is unimodal

Obtain the mode

```
# create a series of pen prices
prices_pens = pd.Series([0,35,40,10,54,35,100,100])

# display the value counts of the
prices_pens.value_counts()
```

```
100     2
35      2
54      1
40      1
10      1
0       1
dtype: int64
```

Number of times each value is appearing in the data

Notice values 35 and 100 each occur 2 times

```
# print the mode
prices_pens.mode()
```

```
0     35
1    100
dtype: int64
```

The mode values

Interpretation: The mode values are 35 and 100. It implies that these values occur equal number of times and have the highest frequency.

Since there are two modes, the data is bimodal.

Obtain the mode

```
# create a series of pen prices
prices_pens = pd.Series([0,0,35,40,10,54,76,34,56,87,34,56,100,99,100,99])

# display the value counts of the
prices_pens.value_counts()
```

```
56     2
99     2
100    2
34     2
0      2
76     1
10     1
40     1
87     1
54     1
35     1
dtype: int64
```

Number of times each value is appearing in the data

Notice values 0, 34, 56, 99, 100 each occur 2 times

```
# print the mode
prices_pens.mode()
```

```
0      0
1     34
2     56
3     87
4     99
5    100
dtype: int64
```

The mode values

Interpretation: The mode values are 0, 34, 56, 87, 99, 100. It implies that these values occur equal number of times and have the highest frequency.

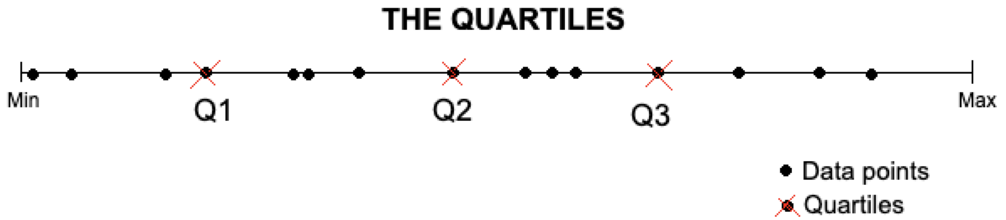
Also note that there are 6 mode values. Data is multimodal.

Partition values

- Partition values are the values which divide the dataset into equal parts
- Note that partition values may not be equispaced
- Also known as quantiles
- We have seen that the median divides the data into two equal halves. Similarly in order to make four parts we use quartiles, for ten parts we use deciles

Quartiles

- The values that divide the dataset into four equal parts are called quartiles



- 25% of the data lies below the first quartile (Q1) and 75% above it
- The second quartile divides the data points in two equal halves. Q2 is the median
- The third quartile (Q3) divides the the observation in 75%-25%

Obtaining the quartiles

The partition values are obtained using the quantile function specifying the percentage of data below the required partition value.

```
# to get the first quartile
prices.quantile(.25)
```

```
34.5
```

```
# to get the second quartile
prices.quantile(.50)
```

```
56.0
```

```
# to get the third quartile
prices.quantile(.75)
```

```
88.0
```

Deciles

- The values that divide the dataset into ten equal parts are called deciles
- 10% of the data lies below the first decile (D1) and 90% above it
- Similarly, 20% of the data lies below the second decile (D2) and 80% above it and so on
- The fifth decile is the same as the second quartile i.e. D5 = Q2 = Median
- There are in all 9 deciles

Obtain the nine deciles (D9) for the variable ‘prices’ defined before.

Similarly we define percentiles as values which divides the data into 100 equal parts.

Obtain the 25th (P25), 50th (P50) and 75th (P75) percentile.

Summary

- Mean, median and mode can be used to impute the missing values in the data
- For a symmetric distribution, mean = median = mode
- For any data,

$$Q2 = D5 = Median = P50$$

- Mean is affected by the presence of outliers
- Trimmed mean is useful when the outliers are present in the data
- Median and mode are not affected by the presence of outliers

Distribution of the Data

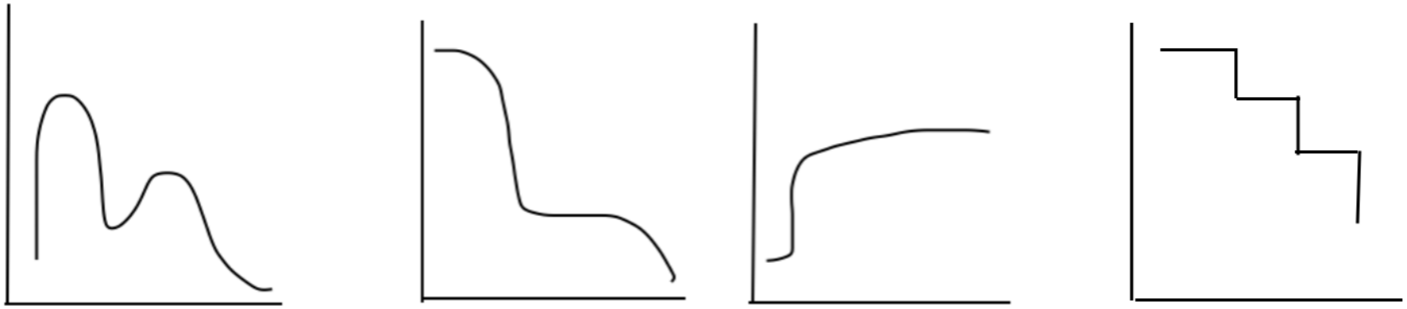
Distribution of the data

- The distribution is a summary of the frequency of values taken by a variable
- The distribution of the data gives information on the shape and spread of the data
- On plotting the histogram or a frequency curve for a variable, we actually look at how the data is distributed over its range

Distribution of the data

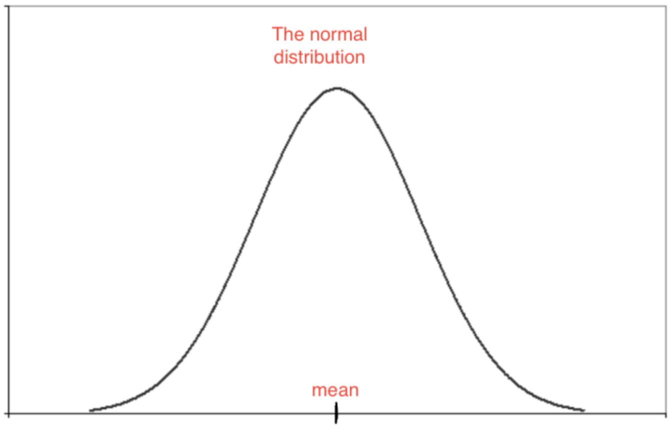
- The data may be distributed in a random matter
- However there are standard distributions in statistics like the normal distribution, binomial distribution, student’s-t distribution and so on

Some distributions



Normal distribution

- It is the most important distribution
- The distribution is bell shaped
- Also called as the gaussian distribution
- The frequent observations are found in the middle of the distribution, and further decrease away towards the tails



Normal distribution

- It is given by the function:

Value taken by the variable X

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad \text{where} \quad -\infty < x, \mu < \infty, \sigma > 0$$

x and μ take value between -∞ to +∞

σ is strictly positive

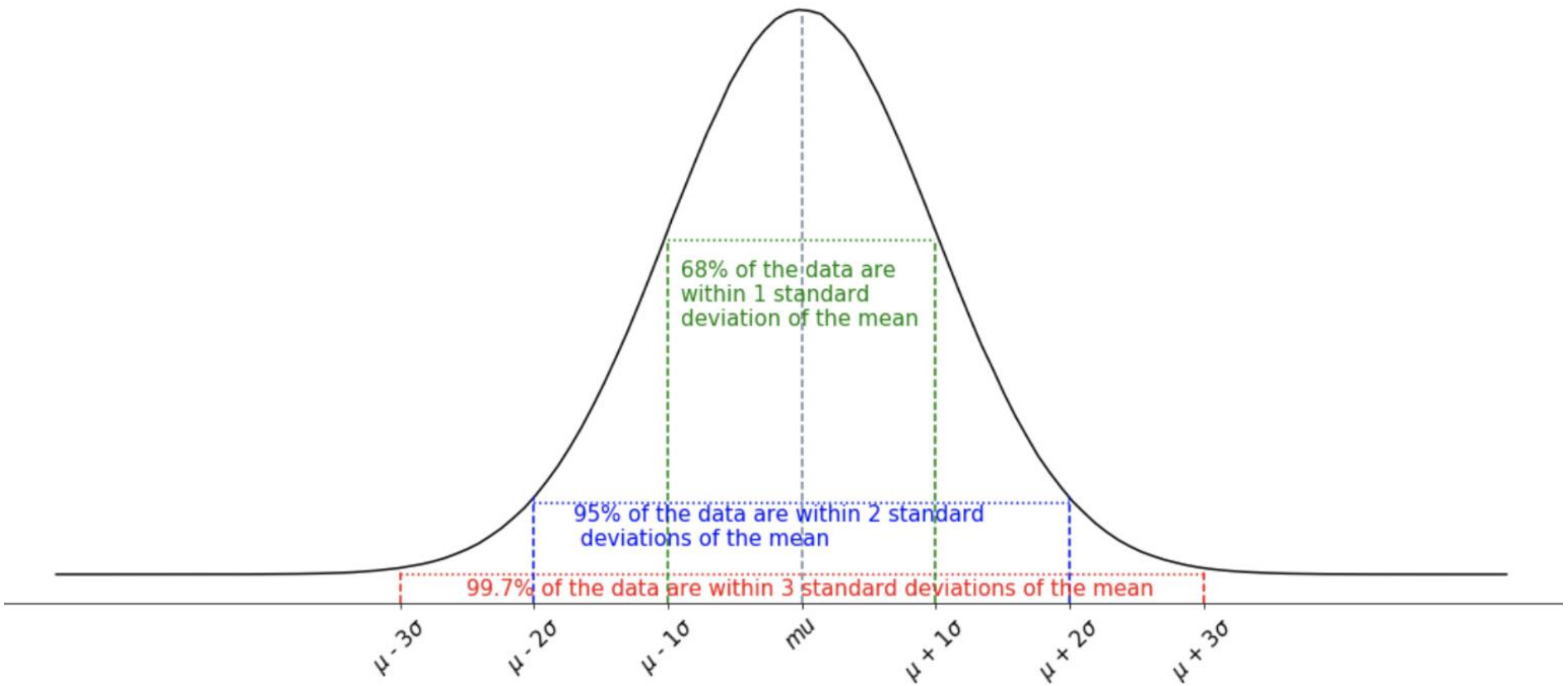
μ is the mean and σ^2 is the variance

- It is said that $X \sim N(\boldsymbol{\mu}, \sigma^2)$, i.e the variable X follows normal distribution with parameter $\boldsymbol{\mu}$ and σ^2

Characteristics of normal distribution

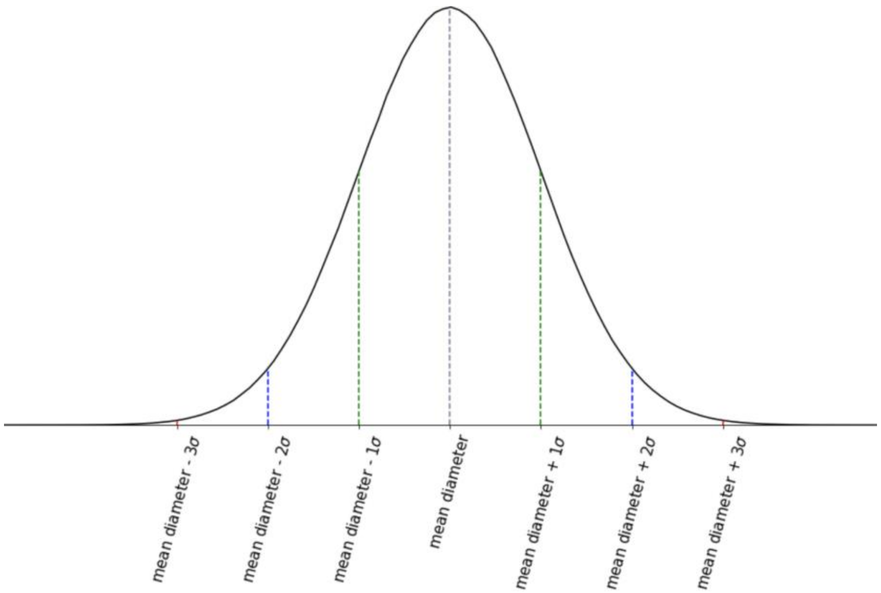
- The distribution is symmetric about the mean
- For a normal distribution mean = median = mode
- The standard normal distribution has mean 0 and variance 1
- For $X \sim N(\mu, \sigma^2)$, $Z = (x - \mu)/\sigma$ then $Z \sim N(0, 1)$ that is the standard normal distribution

The normal distribution spread



Normal distribution - example

Consider a manufacturer producing bolts. He records the diameter of bolts produced.



Most of the observations will be around the mean diameter. As we move away from the mean diameter the number of observations gradually reduce.

Summary

- The distribution of the data gives information on the shape and spread of the data

- The normal/gaussian distribution is bell shaped where
  mean = median = mode

- The standard normal distribution has mean 0 and variance 1

Measures of Dispersion

Measures of Dispersion

- The measure of dispersion refers to the variability within the data
- Variability is the measure of how close or far the data lie from the central value
- Reliability of a measure of central tendency is more if dispersion in the data is less
- It includes range, variance, standard deviation, coefficient of variation and interquartile range

Range

- Range is the difference between the largest and the smallest observation

- It is defined as

$$\text{Range} = X_n - X_1$$

where X_n is the largest value and X_1 is the smallest value

Range

Merits:

- It is a basic way to measure the variability

Demerits:

- If either the smallest or the largest value turns out to be an outlier then the range value is unreliable
- The other in-between observations do not affect the range value
- It does not give a clear picture of variability in the data

Obtain the range

```
# Series 1
year_establishment = pd.Series((1955, 1900, 1980, 1990, 2000))
Range = year_establishment.max() - year_establishment.min()
Range
```

```
100
```

```
# Series 2
year_establishment = pd.Series((1920, 1990, 1925, 1930, 2020))
Range = year_establishment.max() - year_establishment.min()
Range
```

```
100
```

Use the formula for range

Interpretation: The range of both the variables is 100. However, it seen that the first series has observations towards the maximum and the second series has observations towards the minimum.
Thus, range is not a good measure of dispersion.

Variance

Variance is the arithmetic mean of squares of deviations taken from the mean. It is given as

$$\text{Variance } (\sigma^2) = \frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N}$$

Interpretation: The variance looks at how spread out the observations are from the mean. The higher the variance more the data is spread out.

Obtain the variance

```
# obtain the variance for the variable 'prices'
prices.var()
```

```
1054.3290322580644
```

More the data is spread out away from the mean, higher the variance.

PLEASE!! NOTE

The unit of measurement of variance

- The unit of measurement of the variance is not the same as that of the original data
- Consider an industry producing screws. The diameter of each screw is noted in millimeters
- The variance in the diameter is measured in squared millimeters
- Thus a usual practice is to take square root of the variance which would have the same unit as the original data
- The value thus obtained is called the standard deviation of the data

Standard deviation

The standard deviation of the variable is the square root of the variance.

$$\text{Standard deviation } (\sigma) = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \bar{x})^2}{N}}$$

Obtain the standard deviation

```
prices.std()
```

```
32.470433200960905
```

Interpretation: The standard deviation looks at how spread out the observations are from the mean. The higher the standard deviation more the data is spread out.

Coefficient of variation

- The coefficient of variation is a statistical measure of dispersion of data points around the mean

$$\text{Coefficient of Variation (C.V.)} = \frac{\textit{Standard Deviation}}{|\textit{Mean}|} \; X \; 100 = \frac{\sigma}{|\bar{x}|} \; X \; 100$$

- It is always expressed in percentage and is unit free
- Generally, used to compare dispersion of two or more groups
- According to Karl Pearson, C.V. is the percentage variation in the mean and S.D. is the total variation in mean

Obtain the coefficient of variation

```
# importing variation from scipy.stats
from scipy.stats import variation

scipy.stats.variation(prices)
```

```
0.5597598237090757
```

The variation() from the scipy library computes the coefficient of variation.

Note: The name of the function can be quite misleading; do not confuse with 'variance'.

Obtain the coefficient of variation

Consider two financial portfolios, A and B. Obtain the the better portfolio in which an investor should invest.

Portfolio	Expected Return (%)	Volatility (%)
A	25	30
B	10	15

Obtain the coefficient of variation

Obtain the coefficient of variation for the two portfolios

```
# the expected return of portfolio A (in %)
expected_return_A = 25

# the volatility of portfolio A (in %)
volatility_A = 10

# obtain the coefficient of variation
cv =  volatility_A/expected_return_A

# print the coefficient of variation
cv
```

```
0.4
```

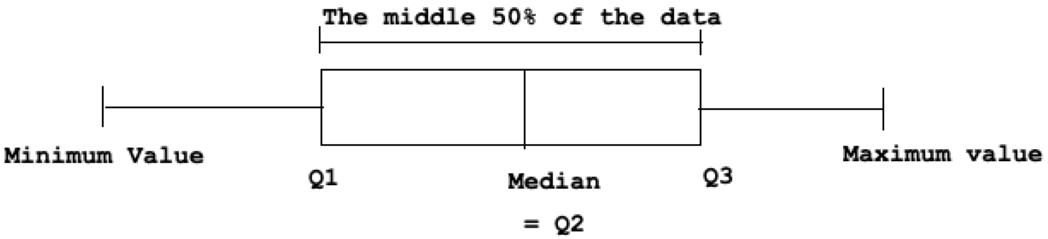
```
# the expected return of portfolio B (in %)
expected_return_B = 30

# the volatility of portfolio B (in %)
volatility_B = 15

# obtain the coefficient of variation
cv =  volatility_B/expected_return_B

# print the coefficient of variation
cv
```

```
0.5
```

Interpretation: The coefficient of variation for portfolio A is lower than that of portfolio B. So, the investor should invest in portfolio A because it is comparatively more consistent in its returns.

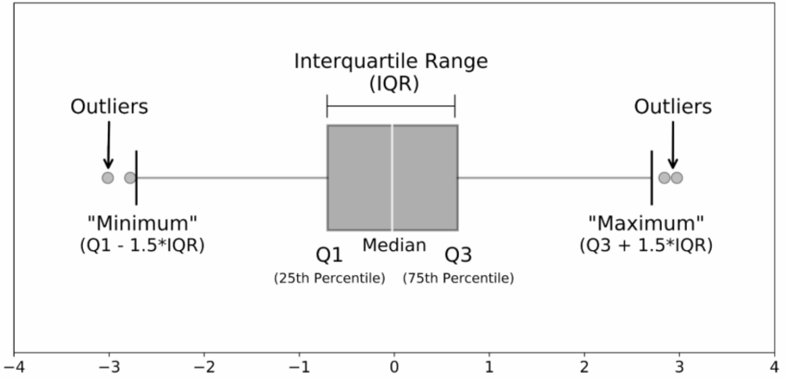
Quartiles

- We have seen quartiles are kind of partition values
- With the help of the quartiles we understand the spread of the data
- Quartiles when plotted with box plot help in identifying the outliers



Understanding box plot

- A box plot is a way of displaying the distribution of data based on a five number summary (minimum, first quartile (Q1), median, third quartile (Q3), and maximum)
- The whiskers give an idea of the spread of the data
- The dots outside of the whiskers are the outliers



Interquartile range

- The interquartile range is defined as the difference between the third and the the first quartile

- It gives the range in which the middle 50% of the data lies

- The value of IQR is used to remove outliers in the data

$$\mathrm{IQR} = \mathrm{Q}_3 - \mathrm{Q}_1$$

Obtain interquartile range

```
# obtain the first quartile
Q1 = prices.quantile(0.25)

# obtain the quartile
Q3 = prices.quantile(0.75)

# obtain the IQR
IQR = Q3 - Q1

# print IQR
IQR
```

```
53.5
```

Interpretation: The interquartile range of the prices is 53.5

Z-score

- Every value in the dataset has the z-score
- To find the z-score of a value, subtract the mean from it and divide it by the standard deviation
- The z-scores are extensively used in statistics especially in hypothesis testing
- It is also used to normalize the data, ie is it scales the data to get the mean of the scaled data as 0 and variance as 1

$$\text{Z- score} = \frac{x_i - \bar{x}}{\sigma}$$

Z-score

The z-score tells how much is the value away from the mean

Z-score Value	What it means?
0	The corresponding value is same as the mean
< 0	The corresponding value is less than the mean
> 0	The corresponding value is greater than the mean
< -3 or > 3	The corresponding value is potential outlier

Summary

- Range is not a reliable measure as it do not consider the in-between observations
- The variables with a near-zero standard deviation have less significance in the analysis
- Coefficient of variation (CV) is the unitless quantity
- Box-plot is used to visualize five-number summary (minimum, Q1, Q2, Q3, and maximum)
- IQR can be used to detect and remove the outliers in the data

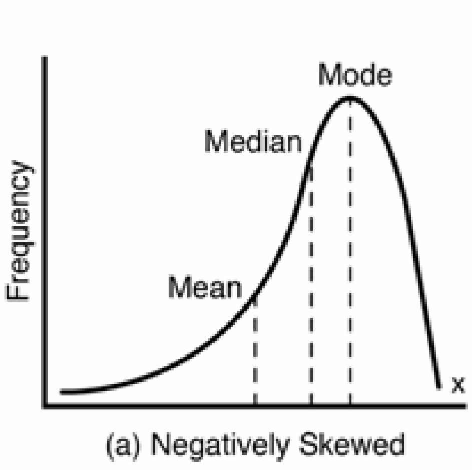
Skewness

Skewness

- Skewness is a lack of symmetry or departure from symmetry
- If the distribution of the data is elongated on either side then the data is said to be skewed
- If the distribution of the data is elongated on the left side then the data is said to be left skewed
- If the distribution of the data is elongated on the right side then the data is said to be right skewed

Skewness

The graph of skewed distribution



mean < median < mode   mean = median = mode   mean > median > mode

Calculate the skewness

The Bowley's Coefficient of Skewness is defined as

$$S_b = \frac{Q_3 - 2\,Q_2 + Q_1}{Q_3 - Q_1}$$

where Q_1, Q_2 and Q_3 are first, second and third quartiles respectively

Value of skewness	Interpretation
$S_b < 0$	The distribution is negatively skewed
$S_b = 0$	The distribution is not skewed (symmetric distribution)
$S_b > 0$	The distribution is positively skewed

Calculate the skewness

The Bowley's coefficient of skewness:

```
# calculate Bowley's coefficient of skewness
# calculate first quartile
Q1 = prices.quantile(0.25)

# calculate second quartile
Q2 = prices.quantile(0.5)

# calculate third quartile
Q3 = prices.quantile(0.75)

s_b = (Q3 - 2*Q2 + Q1) / (Q3 - Q1)

# Bowley's coefficient of skewness
print(s_b)
```

```
0.19626168224299065
```

Interpretation: The coefficient is positive but near zero. Thus, the variable 'prices' is said to be near symmetric.

Calculate the skewness

The Karl Pearson’s Coefficient of Skewness is defined as

$$S_k = \frac{3(mean - median)}{\sigma}$$

Where σ is the standard deviation

Value of skewness	Interpretation
$S_k < 0$	The distribution is negatively skewed
$S_k = 0$	The distribution is not skewed (symmetric distribution)
$S_k > 0$	The distribution is positively skewed

Calculate the skewness

The Karl Pearson's coefficient of skewness:

```
# calculate Karl Pearson's coefficient of skewness
# calculate the mean
price_mean = prices.mean()

# calculate median
price_med = prices.median()

# calculate standard deviation
price_sd = prices.std()

s_k = (3*(price_mean - price_med)) / price_sd

# Karl Pearson's coefficient of skewness
print(s_k)
```

```
0.09835250325524642
```

Interpretation: The coefficient is positive but near zero. Thus, the variable 'prices' is said to be near symmetric.

Calculate the skewness

- The adjusted Fisher-Pearson Coefficient of Skewness is defined as

$$G_1 = \sqrt{\frac{N(N-1)}{N-2}}\,\frac{m_3}{m_2^{3/2}}$$

Where, m_2 and m_3 are the 2nd and 3rd central moments respectively.

$$m_2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})^2 \quad \text{and} \quad m_3 = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})^3$$

- This coefficient of skewness is based on the central moments of the distribution

Calculate the skewness

Value of skewness	Interpretation
$G_1 < 0$	The distribution is negatively skewed
$G_1 = 0$	The distribution is not skewed (symmetric distribution)
$G_1 > 0$	The distribution is positively skewed

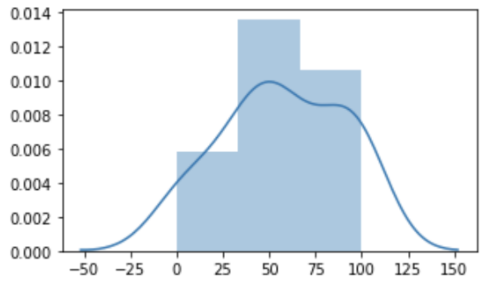
Obtain the skewness

In python, the ‘skew()’ returns the value of adjusted Fisher-Pearson coefficient of skewness.

```
# use the 'skew()' to calculate the skewness
prices.skew()
```

```
-0.20497188990720197
```

```
# import the required libraries
import seaborn as sns
import matplotlib.pyplot as plt

# set the plot size
plt.figure(figsize=(5,3))

# plot a distribution plot
sns.distplot(prices)

# display the plot
plt.show()
```



Interpretation: The variable prices is slightly negative skewed. It can be said it is close to symmetric.

PLEASE!! NOTE

Coefficient of skewness

- The magnitude of the coefficient of skewness explains the extent of skewness in the distribution
- The +/- sign of a coefficient provides the direction of skewness
- The coefficient compares the distribution of the sample to the normal distribution. Thus, the zero value represents the symmetric distribution
- A higher magnitude of the coefficient indicates that the distribution highly differs from the normal distribution

Summary

Bowley's Coefficient of Skewness

- Based on quartiles
- Value lies between -1 to +1
- Can be used to calculate skewness for open-ended distribution
- Preferred if the data has extreme values (outliers)

Karl Pearson's Coefficient of Skewness

- Based on mean, median and standard deviation
- Usually value lies between -3 to +3
- Can not be used for open-ended distribution, as we can not calculate the mean in such case

Fisher - Pearson Coefficient of Skewness

- Based on central moments
- Mostly used measure of skewness by different softwares
- Can not be used for open-ended distribution, as we can not calculate the mean in such case

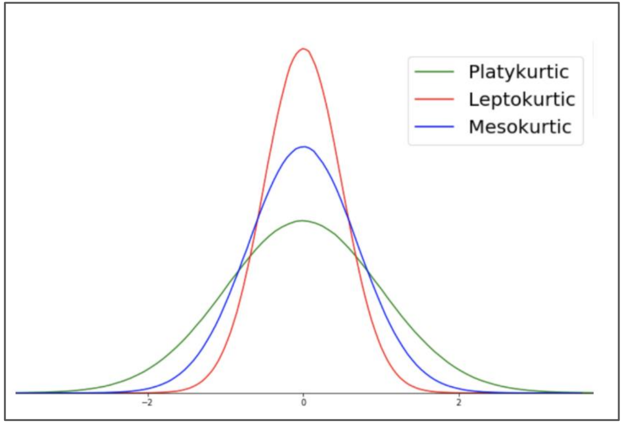
Kurtosis

Kurtosis

- Kurtosis measures the peakedness of the distribution
- In other words, it is a statistical measure that defines how the tails of the distribution differ from the normal distribution
- Kurtosis identifies whether the tails of a given distribution contain extreme values
- A histogram is an effective way to show both the skewness and kurtosis of a data set

Kurtosis

Value	Thickness of Tails	Interpretation
Kurtosis < 0	Thin	The distribution is platykurtic
Kurtosis = 0	Normal	The distribution is mesokurtic
Kurtosis > 0	Thick	The distribution is leptokurtic



Obtain the kurtosis

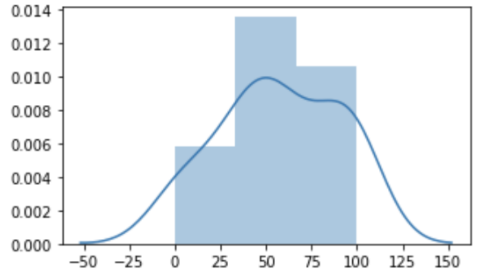
We use kurt() to find the kurtosis

Interpretation: The variable ‘prices’ is platykurtic since -1.036 < 0. This means there are more dispersed prices with lighter tails. i.e. less extreme values.

```
prices.kurt()
```

```
-1.0366224980536156
```

```
# import the required libraries
import seaborn as sns
import matplotlib.pyplot as plt

# set the plot size
plt.figure(figsize=(5,3))

# plot a distribution plot
sns.distplot(prices)

# display the plot
plt.show()
```



Summary

- A mesokurtic distribution is basically a normal distribution
- A leptokurtic distribution has heavy tails on both the sides that indicates the presence of large outliers. Also the distribution is concentrated towards the mean than the normal distribution
- A platykurtic distribution has flat/lighter tails on both the sides and it indicates that there are small amount of outliers in the distributions. Also, the values are more dispersed and fewer values are close to the mean

Covariance

Covariance

- Covariance is a measure of how much two random variables vary together
- Covariance is similar to variance, but variance explains how a single variable varies, and covariance explains how two variables vary together
- It cannot tell whether the value indicates a strong relationship or weak relationship, since the covariance can take any value

Covariance

The covariance for two variables X and Y are given by

$$COV(X,Y) = \frac{\sum_{i=1}^{n}\left(X_i - \overline{X}\right)\left(Y_i - \overline{Y}\right)}{n-1}$$

Xi = values taken by variable X , $\forall$ X $\in$ [1, n]

Yi = values taken by variable Y , $\forall$ Y $\in$ [1, n]

$\overline{X}$ = mean of Xi

$\overline{Y}$ = mean of Yi

Obtain the covariance

```
# create imports series
imports_raw_material = pd.Series([10,11,14,14,20,22,16,12,15,13])

# create exports series
exports_finished_products = pd.Series([12,14,15,16,21,26,21,15,16,14])

# calculate the covarinace
imports_raw_material.cov(exports_finished_products)
```

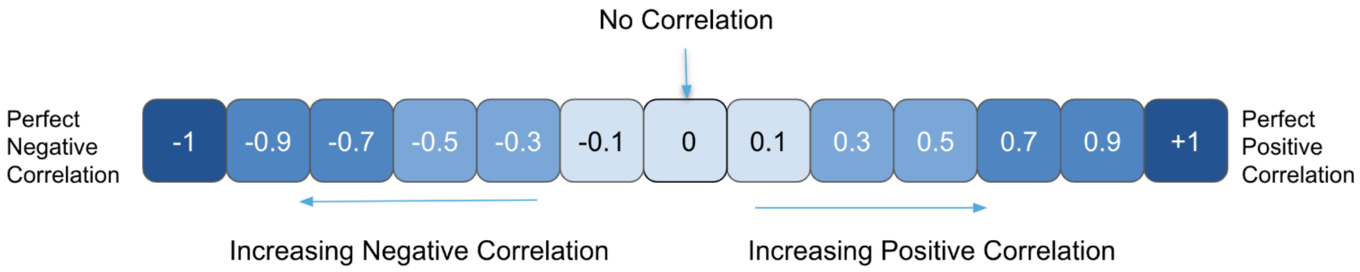
```
15.444444444444443
```

Interpretation: There is a positive covariance between imports of raw material and export of finished goods

Correlation

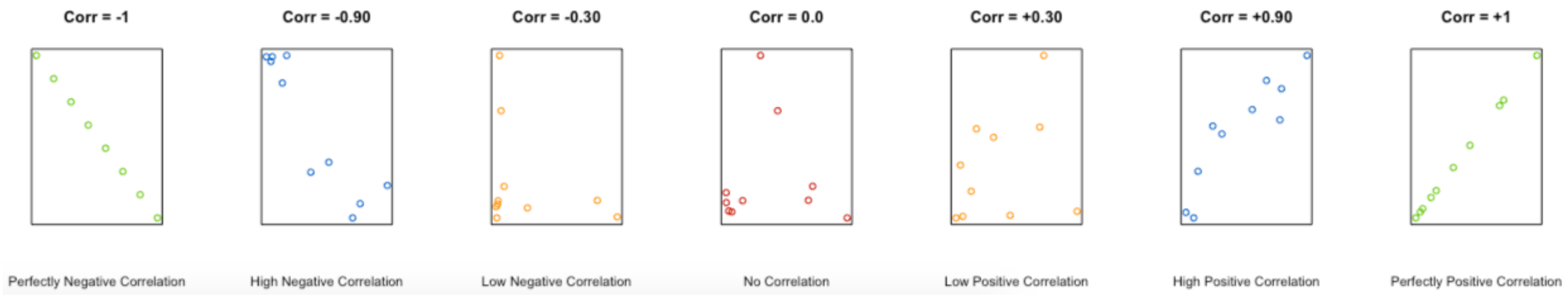
Correlation

- Correlation is the extent of linear relationship among numeric variables
- It indicates the extent to which two variables increase or decrease in parallel
- The value of a correlation coefficient ranges between -1 and 1



Correlation

- A positive correlation exists if variable increase/decrease simultaneously
- A negative correlation exists if one variable increases, while the other variable decreases



Obtain the correlation

```
# create imports series
imports_raw_material = pd.Series([10,11,14,14,20,22,16,12,15,13])

# create exports series
exports_finished_products = pd.Series([12,14,15,16,21,26,21,15,16,14])

# calculate the correlation
imports_raw_material.corr(exports_finished_products)
```

```
0.9458496510416226
```

Interpretation: There is a high positive correlation between imports of raw material and export of finished goods

Correlation

- Correlation is normalized form of covariance

$$Corr(X,Y) = \frac{Cov(X,Y)}{\sigma_x \sigma_y}$$

where σ_x is the standard deviation of X and σ_y is the standard deviation of Y

- The two terms conceptually they are almost identical

Covariance and correlation

Covariance

- tells whether two variables are related by measuring how the variables change in relation to each other
- Gives the direction of the relationship

Correlation

- Similar to covariance
- Tells how strong is the relationship

Case Study

The mpg dataset

- Let us now do a some practise on the dataset
- We will work on mpg dataset
- mpg stands for miles per gallon
- The data is generally used to predict the miles per gallon of a car
- Let us try and get some useful insights from the data

Import the dataset

Let us import the data and display its first 5 rows.

```
# import the mpg data set
df_mpg = pd.read_excel('mpg.xlsx')
```

```
# display head of the data set
# "head()" displays the first five rows
df_mpg.head()
```

	mpg	cylinders	displacement	horsepower	performance	weight	acceleration	model_year	origin	name
0	18.0	8	307.0	130.0	Good	3504	12.0	70	usa	chevrolet chevelle malibu
1	15.0	8	350.0	165.0	Excellent	3693	11.5	70	usa	buick skylark 320
2	18.0	8	318.0	150.0	Average	3436	11.0	70	usa	plymouth satellite
3	16.0	8	304.0	150.0	Excellent	3433	12.0	70	usa	amc rebel sst
4	17.0	8	302.0	140.0	Excellent	3449	10.5	70	usa	ford torino

Check the size and variable type of the data

```
# check the size of the data
df_mpg.shape
```

```
(398, 10)
```

```
# check the variable type
df_mpg.dtypes
```

```
mpg             float64
cylinders         int64
displacement    float64
horsepower      float64
performance      object
weight            int64
acceleration    float64
model_year        int64
origin           object
name             object
dtype: object
```

Interpretation: The dataset has 398 observation and 10 variables.

The variables performance, origin and name are categorical while others are numeric

Check for duplicates and the null values

Interpretation: The dataset has no duplicate rows.

However, variable horsepower has 6 missing values

```
# check duplicates
df_mpg.duplicated().sum()
```

```
0
```

```
# check for missing values
df_mpg.isnull().sum()
```

```
mpg             0
cylinders       0
displacement    0
horsepower      6
performance     0
weight          0
acceleration    0
model_year      0
origin          0
name            0
dtype: int64
```

Descriptive Statistics

The DataFrame.describe() gives the summary statistics of the numeric variables in the dataset

```
# summary statistics of numeric data
df_mpg.describe()
```

	mpg	cylinders	displacement	horsepower	weight	acceleration	model_year
count	398.000000	398.000000	398.000000	392.000000	398.000000	398.000000	398.000000
mean	23.514573	5.454774	193.425879	104.469388	2970.424623	15.568090	76.010050
std	7.815984	1.701004	104.269838	38.491160	846.841774	2.757689	3.697627
min	9.000000	3.000000	68.000000	46.000000	1613.000000	8.000000	70.000000
25%	17.500000	4.000000	104.250000	75.000000	2223.750000	13.825000	73.000000
50%	23.000000	4.000000	148.500000	93.500000	2803.500000	15.500000	76.000000
75%	29.000000	8.000000	262.000000	126.000000	3608.000000	17.175000	79.000000
max	46.600000	8.000000	455.000000	230.000000	5140.000000	24.800000	82.000000

Interpretation: There are 6 missing values in the horsepower variable.

The vehicles under study have an average of 23.5 mpg with standard deviation of 7.8 mpg.

Descriptive Statistics

The DataFrame.describe(include = ‘object’) gives the summary statistics of the categorical variables in the dataset

```python
# summary statistics of numeric data
df_mpg.describe(include = 'object')
```

	performance	origin	name
count	398	398	398
unique	3	3	305
top	Good	usa	ford pinto
freq	161	249	6

Interpretation: From the counts of values, we understand that there are no missing values in the categorical variables.

The data has information on 305 models of cars, in which Ford Pinto occurs 6 times. Most of the cars have originated in the USA.

Skewness

We will now use the mpg dataset to obtain the skewness

```
# obtain the skewness
df_mpg.skew()
```

```
mpg             0.457066
cylinders       0.526922
displacement    0.719645
horsepower      1.087326
weight          0.531063
acceleration    0.278777
model_year      0.011535
dtype: float64
```

Interpretation: The variables mpg, cylinders, displacement, horsepower and weight are positively skewed.

The variable acceleration has near symmetric distribution.

The variable model_year has symmetric distribution.

(Refer slide no. 98)

Kurtosis

Use kurt() to obtain the kurtosis.

```
# obtain the kurtosis
df_mpg.kurt()
```

```
mpg            -0.510781
cylinders      -1.376662
displacement   -0.746597
horsepower      0.696947
weight         -0.785529
acceleration    0.419497
model_year     -1.181232
dtype: float64
```

Interpretation: The distribution of variables mpg, cylinders, displacement, weight and model_year is platykurtic. This implies that there are very less number of extreme observations in these variables.

The variables horsepower and acceleration are leptokurtic. This implies that the distribution of these variables is accumulated near mean, with the presence of more extreme observations.

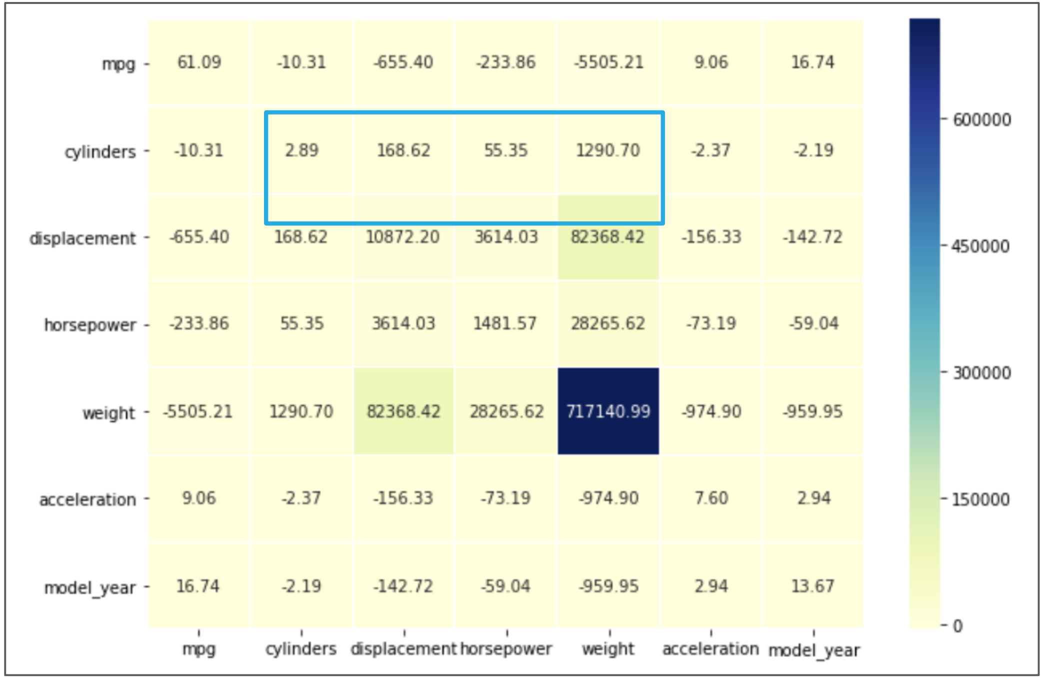
(Refer slide no. 104)

Covariance

```python
# obtain the covariance matrix
cov = df_mpg.cov()

# set the plot size
fig,ax = plt.subplots(figsize=(10, 7))

# plot a heatmap for the correlation matrix
# annot: print values in each cell
# linewidths: specify width of the line specifying the plot
# vmin: minimum value of the variable
# vmax: maximum value of the variable
# cmap: colour code of the plot
# fmt: set the decimal place of annot
sns.heatmap(cov, annot = True, linewidths = 0.05, cmap = "YlGnBu",fmt = '.2f')

# display the plot
plt.show()
```

Covariance

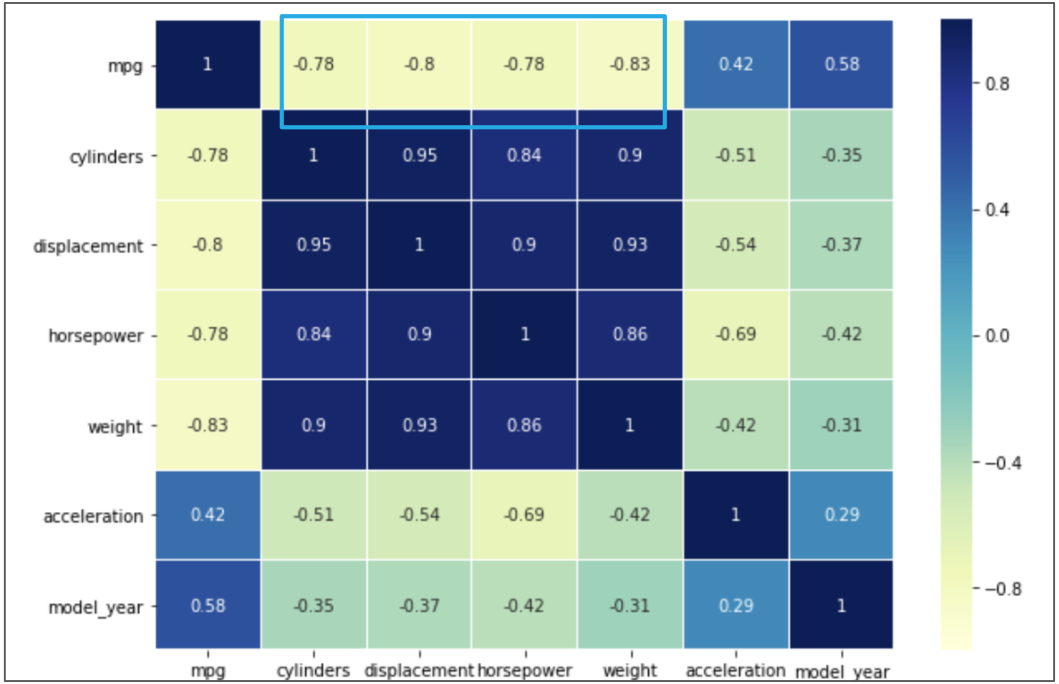
Interpretation: We see there are variable having negative covariance with mpg, except for variables acceleration and model_year have positive correlation.



Correlation

```python
# obtain the correlation matrix
corr = df_mpg.corr()

# set the plot size
fig,ax = plt.subplots(figsize=(10, 7))


# plot a heatmap for the correlation matrix
# annot: print values in each cell
# linewidths: specify width of the line specifying the plot
# vmin: minimum value of the variable
# vmax: maximum value of the variable
# cmap: colour code of the plot
sns.heatmap(corr, annot = True, linewidths = 0.05, vmin = -1 , vmax = 1, cmap = "YlGnBu")

# display the plot
plt.show()
```

Correlation



Interpretation: We see there are variable having high negative correlation with mpg, except for variables acceleration and model_year have positive correlation.

What does the diagonal of the covariance matrix give?

Question:

What does the diagonal of the covariance matrix give?

Answer:

Covariance is a measure of how much two random variables vary together. The diagonal of the covariance matrix gives the covariance of a variable with itself, that is, it gives the variance of that variable

Thank you!

Happy Learning :)