2022

MyEduSolve

Kampus Merdeka
INDONESIA JAYA

Statistics For Data Science

By Group 2

OUR TEAM MEMBER

M. Kamal Jaza

M. Irvan Arfandi

Risma Ashali

Rizal Maulana K.

Shafira Aisyah

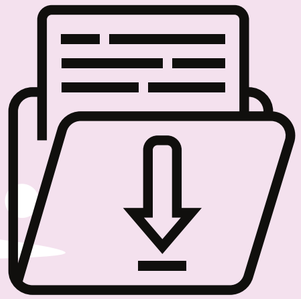
Outline

- Introduction of Statistics
- What is data?
- Types of Variables
- Probability Concept
- Descriptive Statistics
- Variance
- Standard Deviation
- Range & Quantiles
- Skewness & Correlation
- Distribution
- Outliers & Anomalies
- Handling Outliers
- Handling Missing Value
- Confidence Interval
- Hypothesis Testing

Let's get to know Statistics for Data Science

What is Statistics?

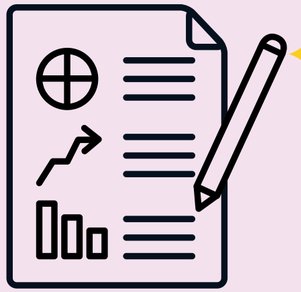
Statistics is a set of mathematical methods and tools that enable us to answer important questions about data.



COLLECTING

ORGANIZING

ANALYZING



INTERPRETING

PRESENTING

What is Data?

Data is a collection of facts, such as numbers, words, measurements, observations or just descriptions of things

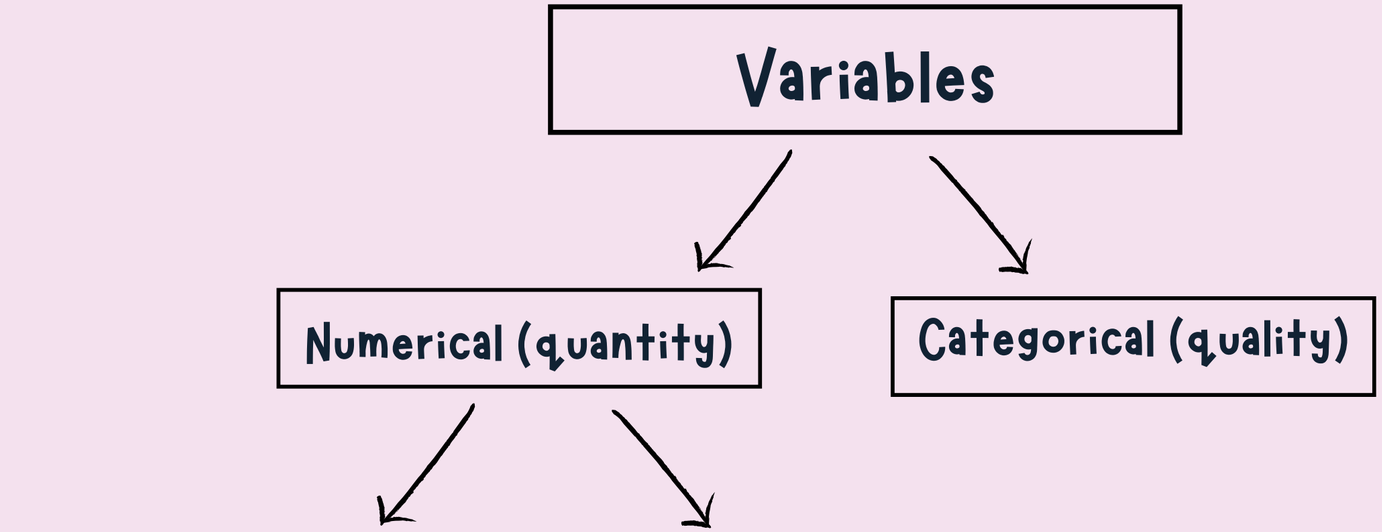
Qualitative Data

Qualitative data are measures of 'types' and may be represented by a name, symbol, or a number code. Qualitative data are data about categorical variables (e.g. what type).

Quantitative Data

Quantitative data are measures of values or counts and are expressed as numbers. Quantitative data are data about numeric variables (e.g. how many; how much; or how often).

Types of Variables



continuous variables are variables whose value is obtained by measuring, i.e., one which can take on an uncountable set of values.

Discrete variables are countable in a finite amount of time. For example, you can count the change in your pocket.

Types of Variables

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

discrete

ordinal

continuous

Quiz 1

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

#Numerical

Diskrit: PassengerId

Continuous: Age, Fare

#Categorical: Embarked, Sex, Survived, Name, Ticket, Cabin, SibSp, Parch

Ordinal: Pclass

Quiz 1

```
titan.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
titan['Age'].mean()
```

```
29.69911764705882
```

```
titan['Age'].median()
```

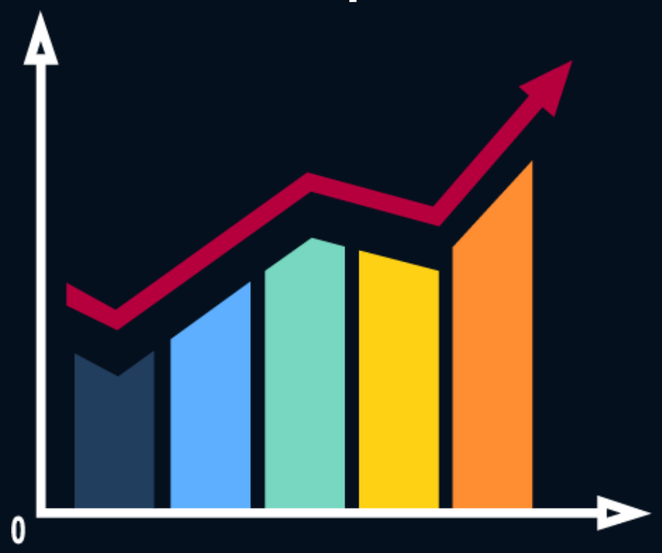
```
28.0
```

```
titan['Age'].mode().values[0]
```

```
24.0
```

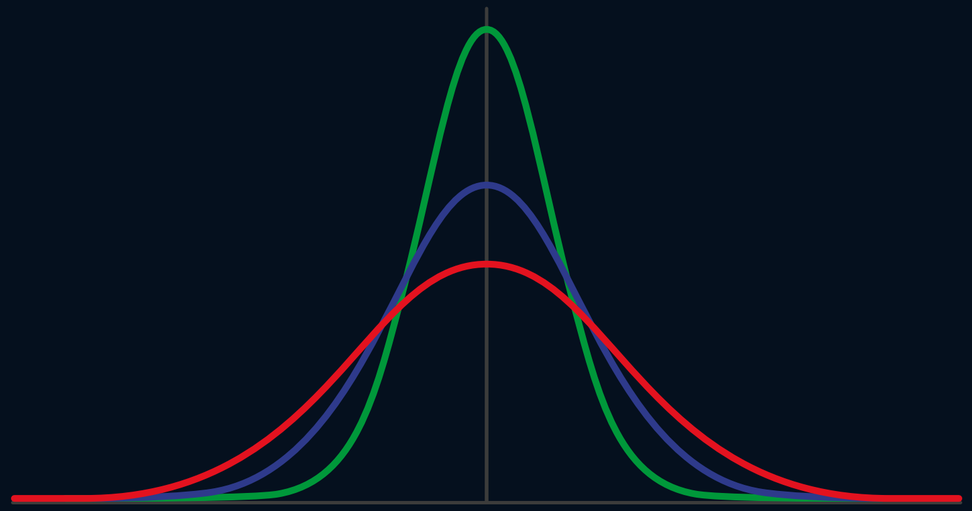
Types of Statistics

Descriptive



Organizing, summarizing, and presenting data in an informative way.

Inferential



Process to obtain conclusion based on sample that aims to generalize of the population.

Probability Concept

Probability is a measure of the likelihood of an event to occur. Many events cannot be predicted with total certainty. Probability can range from 0 to 1, where 0 means the event to be an impossible one and 1 indicates a certain event.

$$\text{Probability of an Event} = \frac{\text{Number of Favorable Outcomes}}{\text{Total Number of Possible Outcomes}}$$

$$P = \frac{n(A)}{n(S)}$$

Quiz 2

Seek 3 Number of Favorable Outcomes from Titanic.csv!

Number of male passengers survived

```
#male passanger survived
titan[(titan['Sex'] == 'male') & (titan['Survived'] == 1)]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
17	18	1	2	Williams, Mr. Charles Eugene	male	NaN	0	0	244373	13.0000	NaN	S
21	22	1	2	Beesley, Mr. Lawrence	male	34.0	0	0	248698	13.0000	D56	S
23	24	1	1	Sloper, Mr. William Thompson	male	28.0	0	0	113788	35.5000	A6	S
36	37	1	3	Mamee, Mr. Hanna	male	NaN	0	0	2677	7.2292	NaN	C
55	56	1	1	Woolner, Mr. Hugh	male	NaN	0	0	19947	35.5000	C52	S
...	...	...	...	...	...	...	...	...	...	...	...	...
838	839	1	3	Chip, Mr. Chang	male	32.0	0	0	1601	56.4958	NaN	S
839	840	1	1	Marechal, Mr. Pierre	male	NaN	0	0	11774	29.7000	C47	C
857	858	1	1	Daly, Mr. Peter Denis	male	51.0	0	0	113055	26.5500	E17	S
869	870	1	3	Johnson, Master. Harold Theodor	male	4.0	1	1	347742	11.1333	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C

109 rows × 12 columns

```
titan[(titan['Sex'] == 'male') & (titan['Survived'] == 1)].shape[0]
```

```
109
```

Number of female passengers survived and over 60 years old

```
#female passengers survived and over 60 years old
titan[(titan['Sex'] == 'female') & (titan['Age'] > 60) & (titan['Survived'] == 1)]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
275	276	1	1	Andrews, Miss. Kornelia Theodosia	female	63.0	1	0	13502	77.9583	D7	S
483	484	1	3	Turkula, Mrs. (Hedwig)	female	63.0	0	0	4134	9.5875	NaN	S
829	830	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0000	B28	NaN

Number of male passengers who are not from 1st class

```
#male passengers who are not from 1st class
titan[(titan['Sex'] == 'male') & (titan['Survived'] == 0) & (titan['Pclass'] != 1)]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
12	13	0	3	Saundercock, Mr. William Henry	male	20.0	0	0	A/5. 2151	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
881	882	0	3	Markun, Mr. Johann	male	33.0	0	0	349257	7.8958	NaN	S
883	884	0	2	Banfield, Mr. Frederick James	male	28.0	0	0	C.A./SOTON 34068	10.5000	NaN	S
884	885	0	3	Sutehall, Mr. Henry Jr	male	25.0	0	0	SOTON/OQ 392076	7.0500	NaN	S
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

391 rows × 12 columns

Group By in Python

Group By each class who are survived

```
#Group By in Python
#Group By each class who are survived
titan[(titan['Survived'] == 1)].groupby (titan['Pclass']).count()
```

Pclass	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	136	136	136	136	136	122	136	136	136	136	117	134
2	87	87	87	87	87	83	87	87	87	87	13	87
3	119	119	119	119	119	85	119	119	119	119	6	119

Quiz 3

Seek 2 Number of Favorable Outcomes from Titanic.csv using exception

In [13]:
```
#male passengers who are not from 1st class
titan[(titan['Sex'] == 'male') & (titan['Survived'] == 0) & (titan['Pclass'] != 1)]
```

Out[13]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
12	13	0	3	Saundercock, Mr. William Henry	male	20.0	0	0	A/5. 2151	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
881	882	0	3	Markun, Mr. Johann	male	33.0	0	0	349257	7.8958	NaN	S
883	884	0	2	Banfield, Mr. Frederick James	male	28.0	0	0	C.A./SOTON 34068	10.5000	NaN	S
884	885	0	3	Sutehall, Mr. Henry Jr	male	25.0	0	0	SOTON/OQ 392076	7.0500	NaN	S
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

391 rows × 12 columns

In [14]:
```
#female passenger aged 18 years not from class 2
titan[(titan['Sex'] == 'female') & (titan['Age'] == 18) & (~(titan['Pclass'] == 2))]
```

Out[14]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
38	39	0	3	Vander Planke, Miss. Augusta Maria	female	18.0	2	0	345764	18.0000	NaN	S
45	50	0	3	Arnold-Franchi, Mrs. Josef (Josefine Franchi)	female	18.0	1	0	349237	17.8000	NaN	S
311	312	1	1	Ryerson, Miss. Emily Borie	female	18.0	2	2	PC 17608	262.3750	B57 B59 B63 B66	C
585	586	1	1	Taussig, Miss. Ruth	female	18.0	0	2	110413	79.6500	E68	S
654	655	0	3	Hegarty, Miss. Hanora "Nora"	female	18.0	0	0	365226	6.7500	NaN	Q
677	678	1	3	Turja, Miss. Anna Sofia	female	18.0	0	0	4138	9.8417	NaN	S
700	701	1	1	Astor, Mrs. John Jacob (Madeleine Talmadge Force)	female	18.0	1	0	PC 17757	227.5250	C62 C64	C
702	703	0	3	Barbara, Miss. Saiide	female	18.0	0	1	2691	14.4542	NaN	C
786	787	1	3	Sjoblom, Miss. Anna Sofia	female	18.0	0	0	3101265	7.4958	NaN	S
807	808	0	3	Pettersson, Miss. Ellen Natalia	female	18.0	0	0	347087	7.7750	NaN	S
855	856	1	3	Aks, Mrs. Sam (Leah Rosen)	female	18.0	0	1	392091	9.3500	NaN	S

Quiz 4

Seek 3 probability from Titanic.csv

```
#probability of each class to survive
```

```
136
```

```
NA1 = titan[(titan['Pclass'] == 1) & (titan['Survived'] == 1)].shape[0]
```

```
NS = titan[(titan['Survived'] == 1)].shape[0]
```

```
NA2 = titan[(titan['Pclass'] == 2) & (titan['Survived'] == 1)].shape[0]
```

```
NA3 = titan[(titan['Pclass'] == 3) & (titan['Survived'] == 1)].shape[0]
```

Quiz 4

Seek 3 probability from Titanic.csv

```
#probability class 1
Pel1 = NA1/NS
print(Pel1)
```

```
0.3976608187134502
```

```
#probability class 2
Pel2 = NA2/NS
print(Pel2)
```

```
0.2543859649122807
```

```
#probability class 3
Pel3 = NA3/NS
print(Pel3)
```

```
0.347953216374269
```

What is inside the descriptive statistics

Descriptive Statistics

- Measure of central tendency
- Measure of spread / dispersion
- Skewness
- Correlation
- Kurtosis

Measure of Central Tendency

Mode

The most frequent number occurring in the data set is known as the mode.

Mode : 21

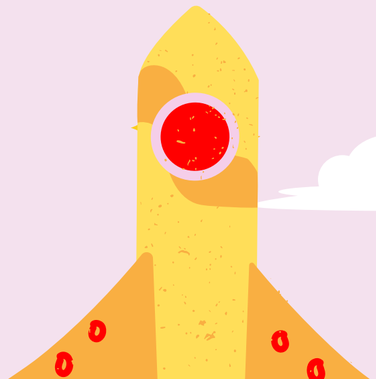
Name	Age
Jaza	22
Irvan	20
Risma	21
Rizal	21
Shafira	21

Mean

Mean represents the average of the given collection of data. It is applicable for both continuous and discrete data

Mean = (22+20+21+21+21) / 5
= 21

Name	Age
Jaza	22
Irvan	20
Risma	21
Rizal	21
Shafira	21



Mean

- All scores in distribution affected the mean
- Many samples from the same population will have similar means
- The mean will change if we add extreme value

Name	Age
Jaza	22
Irvan	20
Risma	21
Rizal	21
Shafira	21

Mean = 21

Name	Age
Jaza	22
Irvan	20
Risma	100
Rizal	21
Shafira	21

Mean = 36

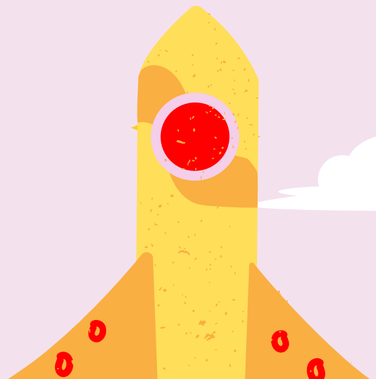
Median

Median represents the mid-value of the given set of data when arranged in a particular order

- Doesn't change much by extreme values (outliers)
- Median is a robust statistics

Name	Exam Score
Jaza	22
Irvan	20
Risma	21
Rizal	21
Shafira	21

Median : 21



Descriptive Statistics

Data Types	Best Measure of Central Tendency
Nominal	Mode
Ordinal	Median
Interval / Ratio (Skewed)	Median
Interval / Ratio (Non-Skewed)	Mean

Using a Python

```
# data titanic
print('Mean: ', df['Fare'].mean())
print('Median: ', df['Fare'].median())
print('Modus: ', df['Fare'].mode().values[0])
```

```
Mean:  32.2042079685746
Median:  14.4542
Modus:  8.05
```

By running the python code above, we can determine the mean, median, and mode value from a data

Case

Create a csv file that contains the personal data of the group members then determines the mean, median, and mode value!

```
df2 = pd.read_csv('kelompok2.csv')
df2.head()
```

	Nama	Usia	Domisili	Hobi	Cita-Cita	Hewan Peliharaan
0	Jaza	22	Jogja	Masak	Pengusaha	10
1	Irvan	20	Palembang	Main gitar	Pemusik	7
2	Risma	21	Semarang	Membaca	Data Scientist	8
3	Rizal	21	Bandung	Nonton	Freelancer	4
4	Shafira	21	Surabaya	Travelling	Data Analyst	2

```
# numpy
print('Mean: ', df2['Usia'].mean())
print('Median: ', df2['Usia'].median())
print('Modus: ', df2['Usia'].mode().values[0])
```

```
Mean:  21.0
Median:  21.0
Modus:  21
```

Measure of Spread / Dispersion

Range

It is simply the difference between the maximum value and the minimum value given in a data set

```
df_age_notnull = df['Age'][df['Age'].notnull()]
```

```
df_age_notnull
```

```
0      22.0
1      38.0
2      26.0
3      35.0
4      35.0
       ...
885    39.0
886    27.0
887    19.0
889    26.0
890    32.0
Name: Age, Length: 714, dtype: float64
```

```
range_age = np.ptp(df_age_notnull)
print(range_age)
```

```
79.58
```

```
df_age_notnull.max()
```

```
80.0
```

```
df_age_notnull.min()
```

```
0.42
```

```
df_age_notnull.max() - df_age_notnull.min()
```

```
79.58
```

Case

Determine the range from other variables on the titanic by using manual methods and using numpy!

```
#manual
print('Max: ', df_fare_notnull.max())
print('Min: ', df_fare_notnull.min())
print('Range: ', df_fare_notnull.max()-df_fare_notnull.min())
```

```
Max:  512.3292
Min:  0.0
Range:  512.3292
```

```
df_fare_notnull = df['Fare'][df['Fare'].notnull()]
df_fare_notnull
```

```
0       7.2500
1      71.2833
2       7.9250
3      53.1000
4       8.0500
        ...
886    13.0000
887    30.0000
888    23.4500
889    30.0000
890     7.7500
Name: Fare, Length: 891, dtype: float64
```

```
#numpy
range_fare = np.ptp(df_fare_notnull)
print(range_fare)
```

```
512.3292
```

Variance

Variance is the expected value of the squared variation of a random variable from its mean value, in probability and statistics. Informally, variance estimates how far a set of numbers (random) are spread out from their mean value.

$$\sigma^2 = \frac{\Sigma (x - \mu)^2}{N}$$ Population Variance

$$s^2 = \frac{\Sigma (x - \bar{x})^2}{n - 1}$$ Sample Variance

Standard Deviation

The standard deviation is a measure of the amount of variation or dispersion of a set of values. A low standard deviation indicates that the values tend to be close to the mean of the set, while a high standard deviation indicates that the values are spread out over a wider range.

The Standard Deviation of a Population:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

The Standard Deviation of a Sample:

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2}$$

Using a Python

```
import statistics
```

```
variance_age = statistics.variance(df_age_notnull)
print(variance_age)
```

```
211.01912474630805
```

```
stdev_age = statistics.stdev(df_age_notnull)
print(stdev_age)
```

```
14.526497332334042
```

By running the python code above, we can determine the variance and standard deviation from a data

Case

Determine the variance and standard deviation from the personal data of the group members using google sheet and python!

Google Sheet

=D7/4

	A	B	C	D
1	Nama	Usia	x-mean	(x-mean)^2
2	Jaza	22	1	1
3	Irvan	20	-1	1
4	Risma	21	0	0
5	Rizal	21	0	0
6	Shafira	21	0	0
7		Total		2
8				
9	Mean	21		
10	Variance	0.5		
11	Standard Deviasion	0.7071067812		

$$\sigma^2 = \frac{\Sigma (x - \mu)^2}{N}$$ Population Variance

$$s^2 = \frac{\Sigma (x - \bar{x})^2}{n - 1}$$ Sample Variance

The Standard Deviation of a Population: $\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$

The Standard Deviation of a Sample: $s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$

Case

Determine the variance and standard deviation from the personal data of the group members using google sheet and python!

Python

Note:

by default, the variantce and standard deviation in Python uses formulas for sample not population.

```
df2_usia_notnull = df2['Usia'][df2['Usia'].notnull()]
```

```
variance_usia = statistics.variance(df2_usia_notnull)
print(variance_usia)
```

```
0.5
```

```
stdev_usia = statistics.stdev(df2_usia_notnull)
print(stdev_usia)
```

```
0.7071067811865476
```

Quantile

Quantiles is a range from any value to any other value. Note that percentiles and quartiles are simply types of quantiles.

- Q0 is the smallest value in the data
- Q1 is the value separating the first quarter from the second quarter of the data
- Q2 is the middle value (median), separating the bottom from the top half
- Q3 is the value separating the third quarter from the fourth quarter
- Q4 is the largest value in the data

```
quantile_age = np.quantile(df_age_notnull, [0,0.25,0.5,0.75,1])
print(quantile_age)
```

```
[ 0.42  20.125 28.    38.    80.   ]
```

✦ Case

Determine the Q0,Q1,Q2,Q3,Q4 from the personal data of the group members using google sheet and python!

```
quantile_usia = np.quantile(df2_usia_notnull, [0,0.25,0.5,0.75,1])
print(quantile_usia)
```

```
[20. 21. 21. 21. 22.]
```

Python

Google Sheet

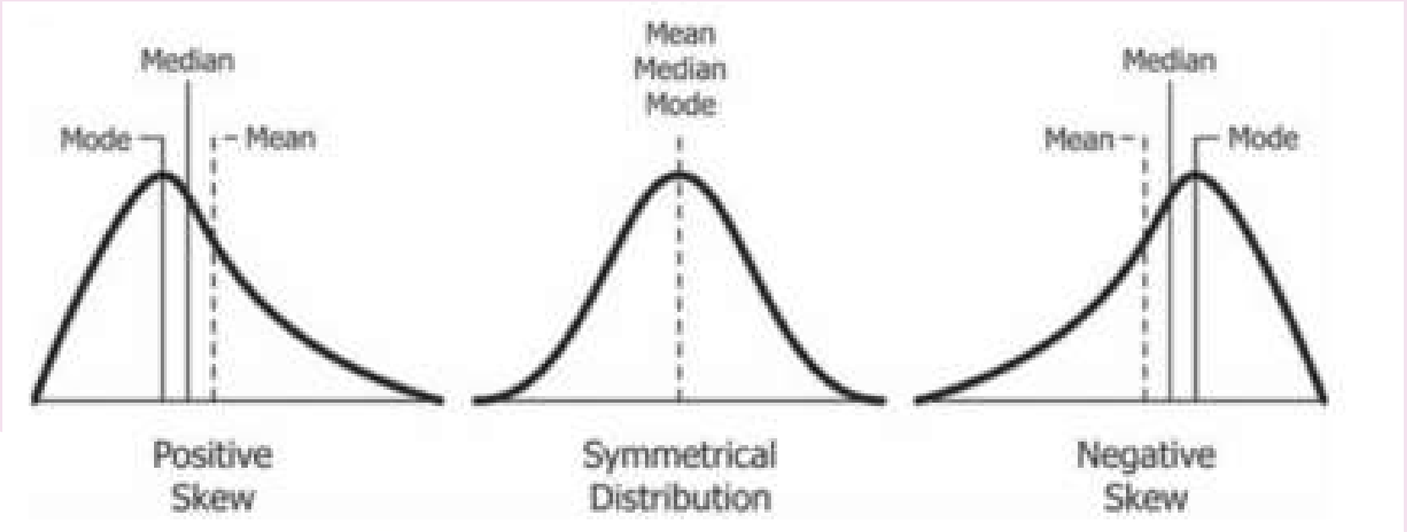
=PERCENTILE($B$2:$B$6;0,25)

A	B	C
Nama	Usia	
Jaza	22	
Irvan	20	
Risma	21	
Rizal	21	
Shafira	21	
Q0	20	
Q1	21	
Q2	21	
Q3	21	
Q4	22	

Skewness

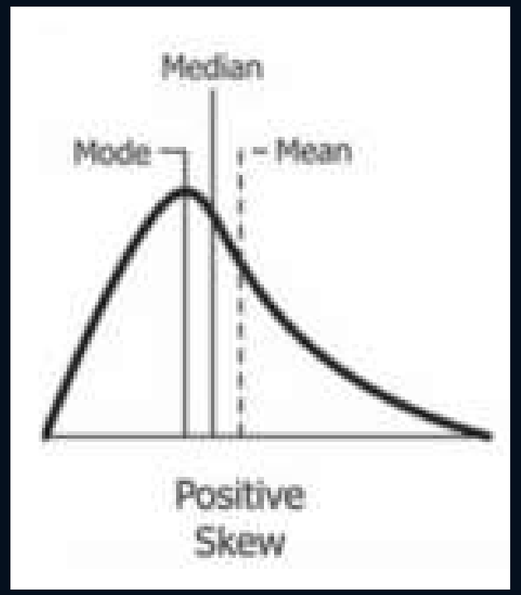
Skewness

Skewness is a measure of the asymmetry of a distribution. A distribution is asymmetrical when its left and right side are not mirror images.



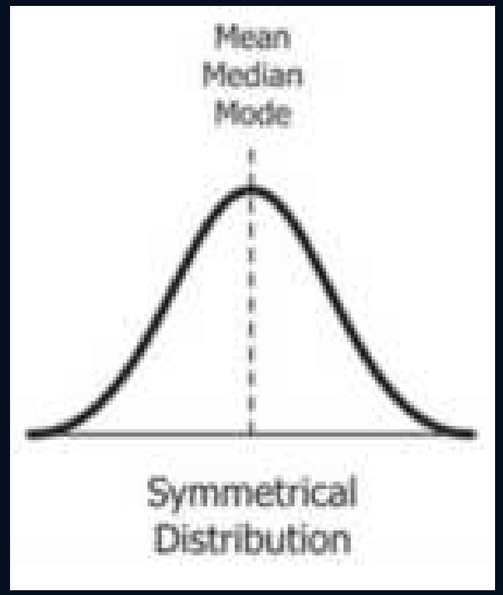
Types of Skewness

Skewness Positive



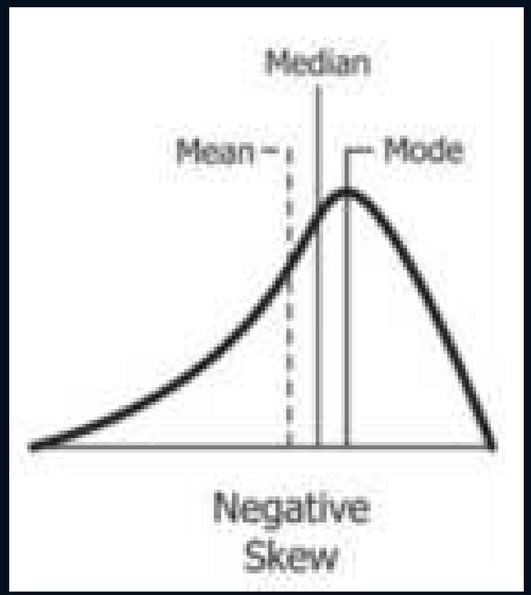
A positive skew distribution is longer on the right side of its peak than on its left. Positive skew is also referred to as right-skewed .

No Skew



When a distribution has no skew, it is symmetrical. Its left and right sides are mirror images.

Skewness Negative

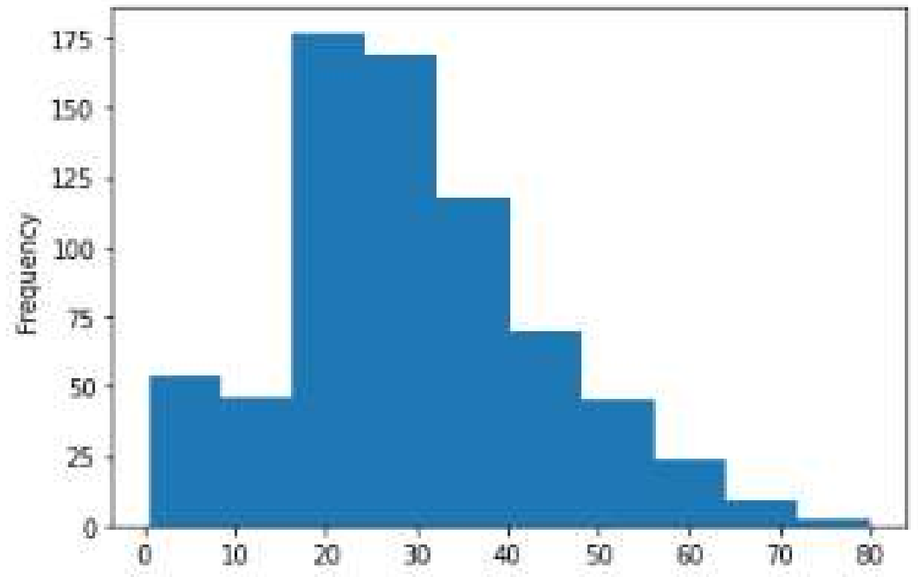


A negative skew distribution is longer on the left side of its peak than on its right. Negative skew is also referred to as left-skewed .

Using a Python

```
#skewness
df['Age'].plot(kind='hist')
```

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



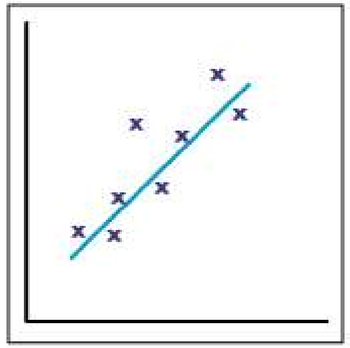
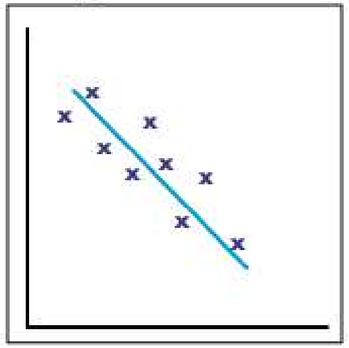
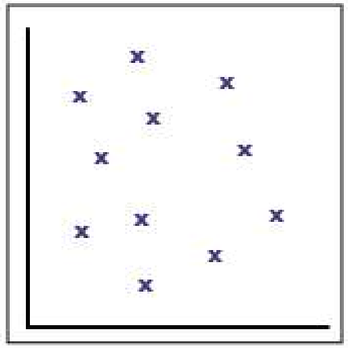
By running the python code above, we can find out the type of skewness from histogram.

In this case, the skewness is positive skewness.

CORRELATION

Correlation

Correlation is a statistical measure that expresses the extent to which two variables are linearly related (meaning they change together at a constant rate). It's a common tool for describing simple relationships without making a statement about cause and effect.

Positive correlation	Negative correlation	No correlation
		
The points lie close to a straight line, which has a positive gradient.	The points lie close to a straight line, which has a negative gradient.	There is no pattern to the points.
This shows that as one variable **increases** the other **increases**.	This shows that as one variable **increases**, the other **decreases**.	This shows that there is **no connection** between the two variables.

Using a Python

In Python, we can see correlation in data by using DataFrame.corr()

```
#Correlation
df[['Age','Survived']].corr()
```

	Age	Survived
Age	1.000000	-0.077221
Survived	-0.077221	1.000000

Code on above will return a output of correlation of age amd survived

Correlation

To get better understanding in correlation data, we can representing all data correlation with heatmap by using seaborn.heatmap

```
plt.figure(figsize=(10,7))
sns.heatmap(df.corr(),annot=True);
```

OUTLIER & MISSING VALUE

Outliers Detection with IQR

- IQR tells how spread the middle values are. It can be used to tell when a value is too far from the middle.
- An outlier is a point which falls more than 1.5 times the interquartile range above the third quartile or below the first quartile.

Step:

1: Arrange the data in incresing order

2: Calculate first (q1) and third quartile (q3)

3: find interquartile range (q3-q1)

4: Find lower bound (q1*1.5)

5: Find upper bound (q3*1.5)

Dataset:

	Nama	Usia	TB	BB
0	Jaza	22	170	75
1	Irvan	20	170	60
2	Risma	21	155	57
3	Rizal	21	176	51
4	Shafira	21	154	56

Anything that lies outside of lower and upper bound is an outlier. Outlier is different with anomalies. Anomalies is error from the data input so that the value does not make sense.

Outliers Detection with IQR

```
data_TB_notnull = data['TB'][data['TB'].notnull()]
qu1, qu3 = np.percentile(data_TB_notnull, [25,75])
print(qu1,qu3)
```

```
155.0 170.0
```

Value of quantile 1 is 155 and quantile 3 is 170

```
iqr = qu3-qu1
print(iqr)
```

```
15.0
```

Value of IQR is 15

```
lower_bound = qu1 - (1.5*iqr)
upper_bound = qu3 + (1.5*iqr)

print(lower_bound)
print(upper_bound)
```

```
132.5
192.5
```

Value of lower bound is 132.5 and upper bound is 192.5. So, anything that lies outside of lower and upper is an outlier

Anomalies Detection

- in Data Science Anomaly and Outlier terms are interchangeable.
- Anomaly detection itself is a technique that is used to identify unusual patterns (outliers) in the data that do not match the expected behavior.

```
#checking anomalies
df['Age'].describe()
```

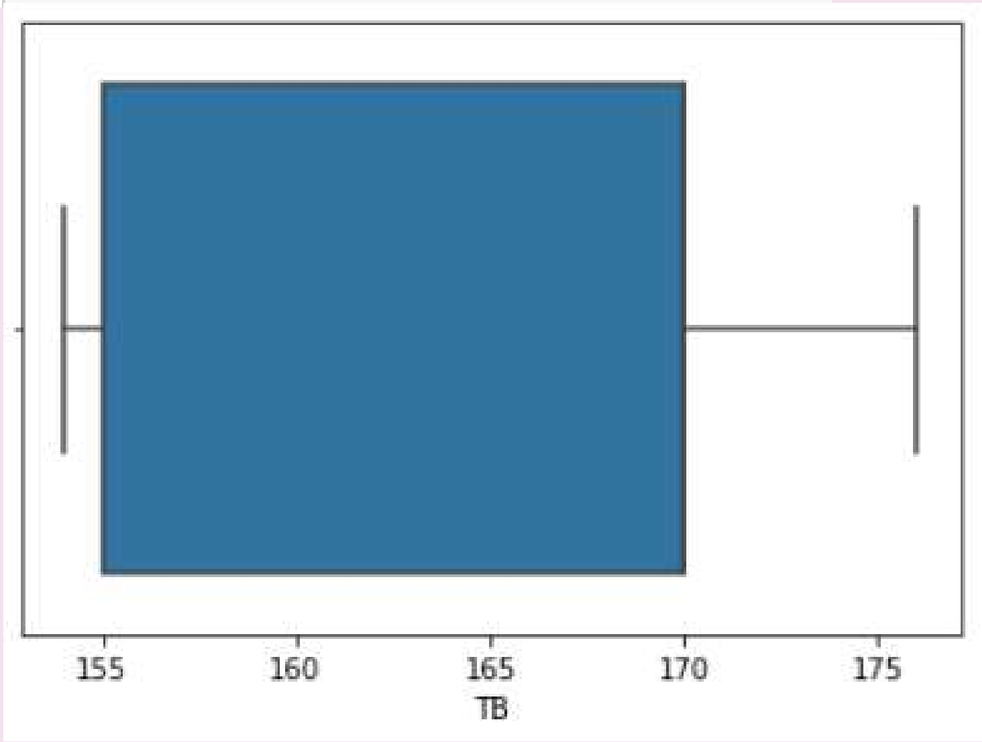
```
count    714.000000
mean      29.699118
std       14.526497
min        0.420000
25%       20.125000
50%       28.000000
75%       38.000000
max       80.000000
Name: Age, dtype: float64
```

In variable Age at titanic dataset, there is no anomalies because it makes sense if someone is 80 years old

Outliers Detection with BOXPLOT

- A box plot is a good way to show many important features of quantitative (numerical) data.
- It shows the median of the data. This is the middle value of the data and one type of an average value.
- It also shows the range and the quartiles of the data. This tells us something about how spread out the data is.

```
#boxplot
sns.boxplot(data['TB'])
```



from the boxplot of Tinggi Badan (kel 2 dataset)

it can be found that there is not an otlier

Handling Outlier

- One of the simplest way to handle outliers is to just remove them from the data. If you believe that the outliers in the dataset are because of errors during the data collection process then you should remove it or replace it with NaN.

```
df['Fare'].describe()
```

```
count    891.000000
mean      32.204208
std       49.693429
min        0.000000
25%        7.910400
50%       14.454200
75%       31.000000
max      512.329200
Name: Fare, dtype: float64
```

```
##handling outlier
df_fare_new = df['Fare'][(df['Fare'] > lower_bound) & (df['Fare'] < upper_bound)]
print(df_fare_new)
```

```
31     146.5208
195    146.5208
268    153.4625
269    135.6333
297    151.5500
305    151.5500
318    164.8667
319    134.5000
325    135.6333
332    153.4625
334    133.6500
337    134.5000
373    135.6333
498    151.5500
609    153.4625
660    133.6500
708    151.5500
856    164.8667
Name: Fare, dtype: float64
```

```
df_fare_new.describe()
```

```
count     18.000000
mean     146.253467
std       10.553834
min      133.650000
25%      135.633300
50%      149.035400
75%      152.984375
max      164.866700
Name: Fare, dtype: float64
```

Originally this data amounted to 891, once the outlier was overcome it turned out to be only left only 18 data.

✦ Handling Missing Value

Missing values are usually represented in the form of Nan or null or None in the dataset.

In this case, we will be filling the missing values with a certain number.

The possible ways to do this are:

- Filling the missing data with the mean or median value if it's a numerical variable.
- Filling the missing data with mode if it's a categorical value.
- Filling the numerical value with 0 or -999, or some other number that will not occur in the data. This can be done so that the machine can recognize that the data is not real or is different.
- Filling the categorical value with a new type for the missing values.

You can use the fillna() function to fill the null values in the dataset.

	Nama	TB	BB	Gender
0	Jaza	170.0	75.0	NaN
1	Irvan	NaN	60.0	Male
2	Risma	155.0	57.0	Female
3	Rizal	176.0	NaN	Male
4	Shafira	154.0	56.0	Female

Handling Missing Value (CATEGORICAL)

- There is 1 missing value at column Gender.
- Gender is a categorical variable, so it can be fill with mode

- After handling the missing value, it can be seen that there is 3 female and 2 male at the data

```
#categorical
dk2['Gender'].value_counts()
```

```
Male      2
Female    2
Name: Gender, dtype: int64
```

```
dk2.Gender.isna().sum()
```

```
1
```

```
val = dk2.Gender.mode().values[0]
dk2['Gender'] = dk2.Gender.fillna(val)
dk2['Gender'].isna().sum()
```

```
0
```

```
dk2['Gender'].value_counts()
```

```
Female    3
Male      2
Name: Gender, dtype: int64
```

Handling Missing Value (NUMERICAL)

- There is 1 missing value at column TB.
- TB is a numerical variable, so it can be fill with median or mean. Before that, we must check the distribution of the data. If it has normal distribution, fill with mean. But if it skewness, it can be fill with median.

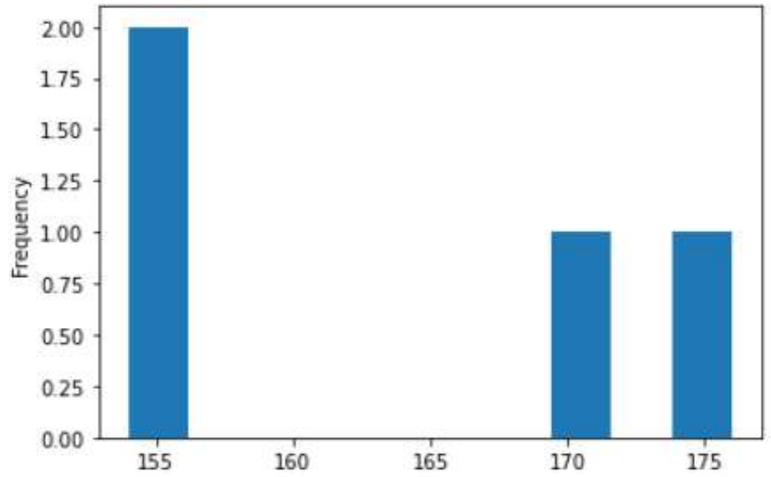
- From the histogram plot, we can see that the distribution is right skew, so it can be fill with median

- After handling the missing value, we can see that there is no missing value again.

```
#numerik
dk2['TB'].isna().sum()
```

```
1
```

```
dk2['TB'].plot(kind='hist');
```




```
val = dk2['TB'].median()
dk2['TB'] = dk2['TB'].fillna(val)
```

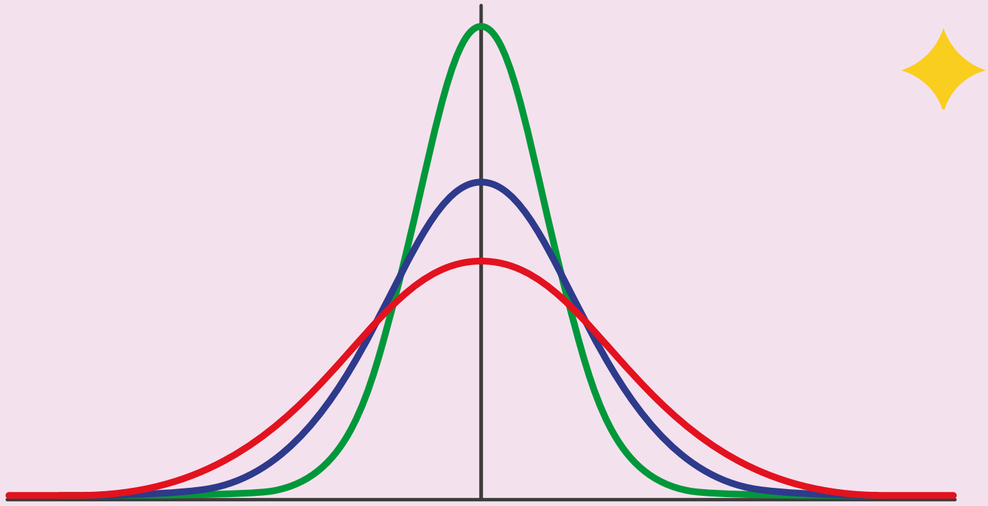
```
dk2['TB'].isna().sum()
```

```
0
```

Inferential Statistics

Inferential statistics provides a way to draw conclusions about broad groups or populations based on a set of sample data. In some instances, it's impossible to get data from an entire population or it's too expensive. Inferential statistics solves this problem.

Using Python to apply inferential statistics concepts including sampling distributions, confidence intervals, hypothesis testing, etc.



Confidence Intervals

A confidence interval, in statistics, refers to the probability that a population parameter will fall between a set of values for a certain proportion of times. Analysts often use confidence intervals than contain either 95% or 99% of expected observations. Thus, if a point estimate is generated from a statistical model of 10.00 with a 95% confidence interval of 9.50 - 10.50, it can be inferred that there is a 95% probability that the true value falls within that range.

First we will import some modules and start random seed 42 times

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


np.random.seed(42)
```

Confidence Intervals

We use dataset named "coffee_dataset" with 2974 rows and 4 columns

In [2]:
```
df = pd.read_csv('coffee_dataset.csv')
df.head()
```

Out[2]:

	user_id	age	drinks_coffee	height
0	4509	<21	False	64.538179
1	1864	>=21	True	65.824249
2	2060	<21	False	71.319854
3	7875	>=21	True	68.569404
4	6254	<21	True	64.020226

In [3]:
```
df.shape
```

Out[3]: (2974, 4)

Confidence Intervals

We will use 200 samples from the dataset

```
coffee_full = pd.read_csv('coffee_dataset.csv')
coffee_red = coffee_full.sample(200)
coffee_red
```

	user_id	age	drinks_coffee	height
2402	2874	<21	True	64.357154
2864	3670	>=21	True	66.859636
2167	7441	<21	False	66.659561
507	2781	>=21	True	70.166241
1817	2875	>=21	True	71.369120
...	...	...	...	...
1187	6237	<21	False	62.493744
463	1857	<21	False	66.476106
1195	6397	<21	False	64.555794
1080	4065	<21	False	66.842149
1422	3971	<21	False	61.891849

200 rows × 4 columns

Confidence Intervals

Explore the dataset

Explore Dataset

```
coffee_red.head()
coffee_red.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 200 entries, 2402 to 1422
Data columns (total 4 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   user_id         200 non-null    int64  
 1   age             200 non-null    object 
 2   drinks_coffee   200 non-null    bool   
 3   height          200 non-null    float64
dtypes: bool(1), float64(1), int64(1), object(1)
memory usage: 6.4+ KB
```

Confidence Intervals

Proportion of coffee drinkers in sample

```
mean_drinker = coffee_red['drinks_coffee'].mean()
mean_nondrinker = 1 - mean_drinker
print(mean_drinker)
print(mean_nondrinker)
```

```
0.595
0.405
```

Average height of coffee drinkers and non drinkers

Average height of coffee drinkers

```
In [7]: drinks_height = coffee_red[coffee_red['drinks_coffee']]['height'].mean()
        drinks_height
Out[7]: 68.11962990858618
```

Average height of non-coffee drinkers

```
In [8]: nondrinks_height = coffee_red[~coffee_red['drinks_coffee']]['height'].mean()
        nondrinks_height
Out[8]: 66.78492279927877
```

Confidence Intervals

Bootstrap sample of 200 draws

```
bootsamp = coffee_red.sample(200,replace=True)
bootsamp['drinks_coffee'].mean()
```

```
0.605
```

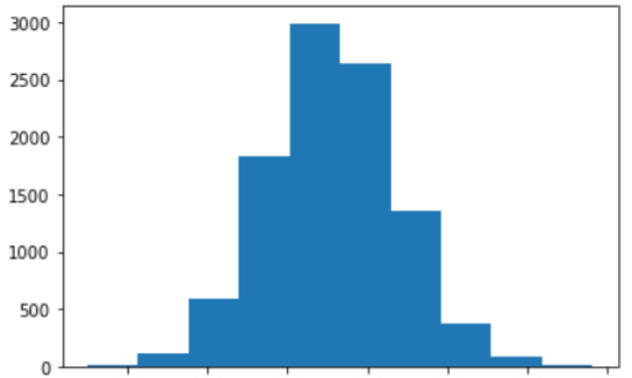
Confidence Intervals

Bootstrap sample 10,000 times

```
boot_means = []
for _ in range(10000):
    bootsamp = coffee_red.sample(200,replace=True)
    mean = bootsamp[bootsamp['drinks_coffee']==False]['height'].mean()
    boot_means.append(mean)

plt.hist(boot_means)
```

```
(array([  16.,  106.,  590., 1831., 2989., 2644., 1352.,  377.,   86.,
          9.]),
 array([65.24631713, 65.56293231, 65.8795475 , 66.19616268, 66.51277787,
        66.82939305, 67.14600823, 67.46262342, 67.7792386 , 68.09585379,
        68.41246897]),
 <BarContainer object of 10 artists>)
```



Confidence Intervals

Obtain 95% confidence interval

```
In [12]: np.percentile(boot_means,2.5),np.percentile(boot_means,97.5)
Out[12]: (65.99291328157521, 67.58402738281573)

In [14]: coffee_full[coffee_full['drinks_coffee']==False]['height'].mean()
Out[14]: 66.44340776214703
```

Confidence Intervals

Another example with dice, bootstrap 10 times

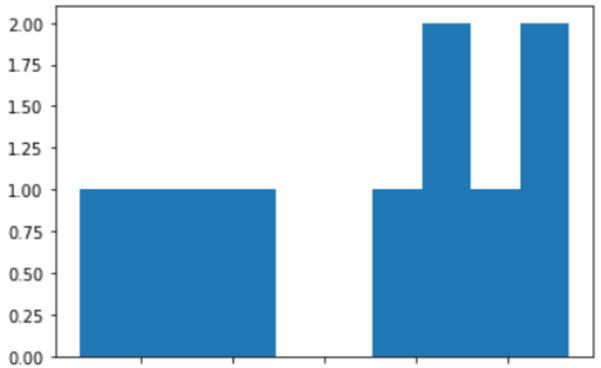
```
dice = [1,2,3,4,5,6]
dices = pd.DataFrame({'number' : dice})
bootsamp2 = dices.sample(6, replace=True)
dice_mean=[]
print(bootsamp2)

for i in range(10):
    bootsamp2 = dices.sample(6, replace=True)
    mean = bootsamp2['number'].mean()
    dice_mean.append(mean)

plt.hist(dice_mean)
```

```
   number
4       5
5       6
1       2
2       3
3       4
3       4

(array([1., 1., 1., 1., 0., 0., 1., 2., 1., 2.]),
 array([2.16666667, 2.43333333, 2.7       , 2.96666667, 3.23333333,
        3.5       , 3.76666667, 4.03333333, 4.3       , 4.56666667,
        4.83333333]),
 <BarContainer object of 10 artists>)
```



Hypothesis Testing

Hypothesis Testing is a type of statistical analysis in which you put your assumptions about a population parameter to the test. It is used to estimate the relationship between 2 statistical variables.

First we will import some modules and use dataset named "blood_pressure"

```
import pandas as pd
from scipy import stats
from statsmodels.stats import
weightstats as stests

df = pd.read_csv('blood_pressure.csv')
```

Hypothesis Testing

The dataset has 120 rows and 5 columns

In [3]:
```
df[['bp_before','bp_after']].describe()
df.head(5)
```

Out[3]:

	patient	sex	agegrp	bp_before	bp_after
0	1	Male	30-45	143	153
1	2	Male	30-45	163	170
2	3	Male	30-45	153	168
3	4	Male	30-45	153	142
4	5	Male	30-45	146	141

In [4]:
```
df.shape
```

Out[4]: (120, 5)

Hypothesis Testing

Paired sampled t-test: The paired sample t-test is also called dependent sample t-test. It's an uni variate test that tests for a significant difference between 2 related variables. An example of this is if you where to collect the blood pressure for an individual before and after some treatment, condition, or time point.

H0: means difference between two sample is 0

H1: mean difference between two sample is not 0

```
In [6]: ttest,pval = stats.ttest_rel(df['bp_before'],df['bp_after'])
        pval

Out[6]: 0.0011297914644840823

In [7]: if pval<0.05:
            print('Reject null hypothesis')
        else:
            print('Accept null hypothesis')

        Reject null hypothesis
```

Thank You

for your attention