This notebook shows the statistical concepts which is mostly used in Data Science.Lot more information can be included. Please let me know topics which can be included...!

Table of Contents

1. What is Statistics?

Table of Contents

Statistics is a form of mathematical analysis that uses quantified models and representations for a given set of experimental data or real-life studies.Statistics studies methodologies to gather, review, analyze and draw conclusions from data.It is the discipline that concerns the collection, organization, analysis, interpretation and presentation of data.

According to Vasily Klyuchevsky

> Statistics is the Science about how, not being able to think and understand make the figures do it yourself.

Statistics is the explanation of the insights (variance) in the light of what remains unexplained.

2.Why is Statistics important?

Table of Contents

In our day to day life , we are always confronted with the situations where the outcome is not certain and have to make decision based on incomplete data. For example

- Should I run for the bus ?
- Should I take the medications?
- Which stock should I buy?

With the help of statistics , we will be able to find out

- The clarification for our questions
- Identify the variable/feature that will answer our question
- Determine the required sample size.
- Describe the variation
- Make quantitative statements about the estimated parameters.
- Make predictions based onyour data.

- Also helps us in finding hidden patterns and buisness insights from the data.

Thus we can when we have huge amount of data , we can apply the statistical concepts so that right and meaningful conclusion can be drawn. Thus in this way we can convert our data into useful information.

3.Display of Statistical data

Table of Contents

The choice of statistical procedure depends on the data type. Data can be categorical or numerical. In addition to this , we distinguish data between univariate,bivariate and multivariate data. *Univariate* data are data of only one variable e.g the size of the person. *Bivariate* data have two parameters like income as a variation of experience. Multivariate data have three or more variables e.g the position of the particle in space etc.

- **Categorical**

  a) **boolean** :- data which can have possible only two values - male/female - smoker/non smoker - True/false

  b) **Nominal** :- Many classifcations reqires more than two categories.Such data are called nominal data. An example married/single/ divorced

- **Numerical**

  a) **Numerical Continuous** :- It is recommended to record the data in it's original continuous format with meaningful number of deciaml places. For the price of house , sales price etc.

  b) **Numerical Discrete** :- Some numerical data can take only integer values. These data are numerically discrete. For example no of children: 0,1,2

To understand how the data is plotted . Please go throught the following kernels :

https://www.kaggle.com/saurav9786/matplot-tutorial-for-everyone

https://www.kaggle.com/saurav9786/data-visualization-using-seaborn

https://www.kaggle.com/saurav9786/interactive-3-d-plots-for-data-visualization

4.Populations and Samples

Table of Contents

In the statistical analysis of data, we use data from a few selected samples to draw conclusions about the population from which these samples were taken.The main difference between a population and a sample has to do with how observations are assigned to the data set

Population : Includes all of the elements from a set of data. **Sample**: Consists of one or more observations from the population.

Please note that more than one sample can be derived from the population.

When estimating a parameter of a population, e.g., the weight of male Europeans,we cannot measure all subjects.

Parameter : Characteristic of a population, such as a mean or standard deviation. Often notated using Greek letters. **Statistic** : A measurable characteristic of a sample. Examples of statistics are: • the mean value of the sample data • the range of the sample data • deviation of the data from the sample mean

Sampling distribution : The probability distribution of a given statistic based on a random sample. **Statistical inference** : Enables you to make an educated guess about a population parameter based on a statistic computed from a sample randomly drawn from that population.

Population parameters are often indicated using Greek letters, while **sample statistics** typically use standard letters.

5.Distribution of one variable

Table of Contents

Distribution of one variable is termed as univariate distributions. They can be further divided into continuous distribution where the value can take float values (sales price of an item) and discreet distributions where the observations can take only the integer values (no of children)

Distribution Center

There are different parameters which charaterize the center of distribution with different parameters. The data can be evaluated by their rank or by their value. The most commonly used parameter are following :-

1. **Mean** :- BY talking about mean we are talking about the average or the arithmetic mean. It is calculated by the (sum of the observations)/no of observations.

$$\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n} = \frac{X_1 + X_2 + \lambda \; + X_n}{n}$$

2.**Median** :- The median value is the that is the half way when the data are ranked in order. In contrast to the mean , the median is not affected by the outlier.

$$Median = \left(\frac{n+1}{n}\right)^{th} term$$

2. **Mode** :- Mode refers to the most frequent occurring value in a distribution.

$$Mode = l + h\left(\frac{f_m - f_1}{2f_m - f_1 - f_2}\right)$$

Where,

l = Lower Boundary of modal class

$h = size\ of\ model\ class$

$f_m = Frequency\ corresponding\ to\ modal\ class$

$f_1 = Frequency\ preceding\ to\ modal\ class$

$f_2 = Frequency\ proceeding\ to\ modal\ class$

3. **Geometric Mean** :- Geometric mean can be used to describe the location the distribution. It can be calculated via by the arithmetic mean of the log of the values.

$$mean_{geometric} = \left(\prod_{i=1}^{N} x_i\right)^{1/N} = exp\left(\frac{\sum_i ln(x_i)}{n}\right)$$

Quantifying variability

1. **Range** :- Range is the diffrence between the highest and the lowest value of data. Range is affected by the outliers (data points are the value much higher or lower than the rest of the data). range = x(max) - x(min)
2. **Interquartile range** :- First percentile (Q1)divides the vakue between minumum and Q2 into two equal halfs.Q1 is that value which has 25% value below it and rest above. IQR= Q3-Q1

3. **Standard Deviation and variance** :- The Standard Deviation is a measure of how spread out numbers are.Its symbol is σ (the greek letter sigma).It is the square root of the Variance.

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

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

4. **Variance** :- It is defined by the average of the squared differences from the mean.Interpreting variance is not intuitive. Instead which we look for the standard deviation which has the same units as the variable.
5. **Coefficient of the variation** :- It is defined as the ratio of the standard deviation to mean. In symbolic form,

$$CV = \frac{S}{\overline{X}} \text{ for the sample data and } = \frac{\sigma}{\mu} \text{ for the population}$$

The Empirical rule and Chebyshev Rule

The Empirical rule

- Approximately 68% of the data lie within one standard deviation of the mean, that is, in the interval with endpoints $\bar{x}\pm s$ for samples and with endpoints $\mu\pm\sigma$ for populations; if a data set has an approximately bell-shaped relative frequency histogram.
- Approximately 95% of the data lie within two standard deviations of the mean, that is, in the interval with endpoints $\bar{x}\pm 2s$ for samples and with endpoints $\mu\pm 2\sigma$ for populations; and
- Approximately 99.7% of the data lies within three standard deviations of the mean, that is, in the interval with endpoints $\bar{x}\pm 3s$ for samples and with endpoints $\mu\pm 3\sigma$ for populations.

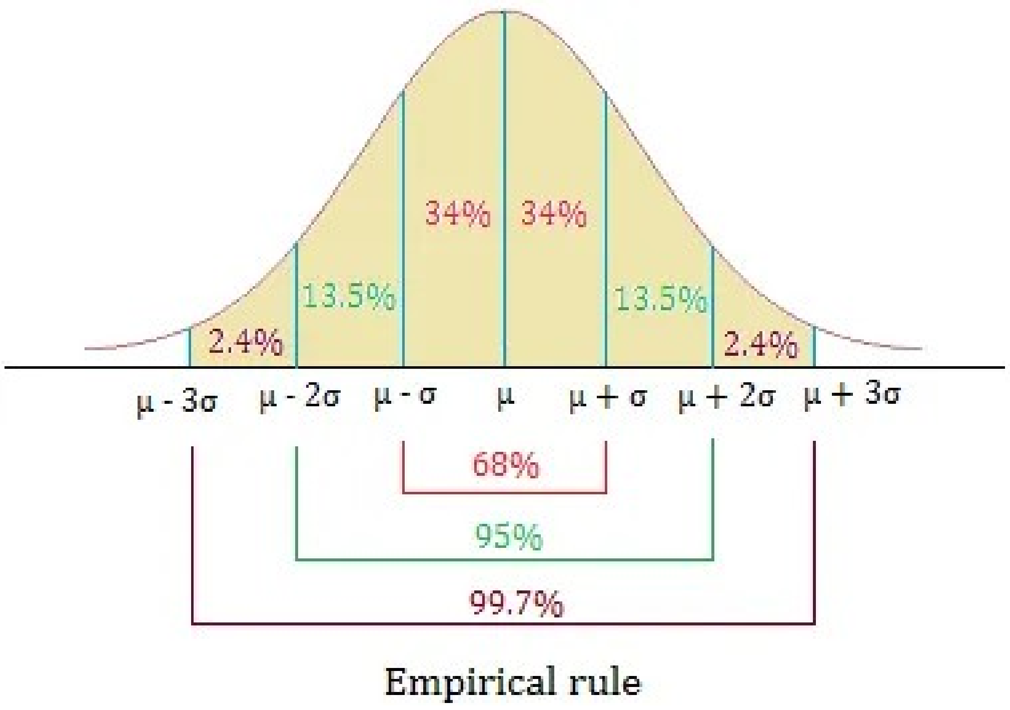


Image Src :- https://statistics-made-easy.com/empirical-rule/

Chebyshev Rule

The Empirical Rule does not apply to all data sets, only to those that are bell-shaped, and even then is stated in terms of approximations. A result that applies to every data set is known as Chebyshev's Theorem.For any numerical data set,

- At least 3/4 of the data lie within two standard deviations of the mean, that is, in the interval with endpoints x¯±2s for samples and with endpoints μ±2σ for populations;
- At least 8/9 of the data lie within three standard deviations of the mean, that is, in the interval with endpoints x¯±3s for samples and with endpoints μ±3σ for populations;
- At least 1−1/k2 of the data lie within k standard deviations of the mean, that is, in the interval with endpoints x¯±ks for samples and with endpoints μ±kσ for populations, where k is any positive whole number that is greater than 1 .

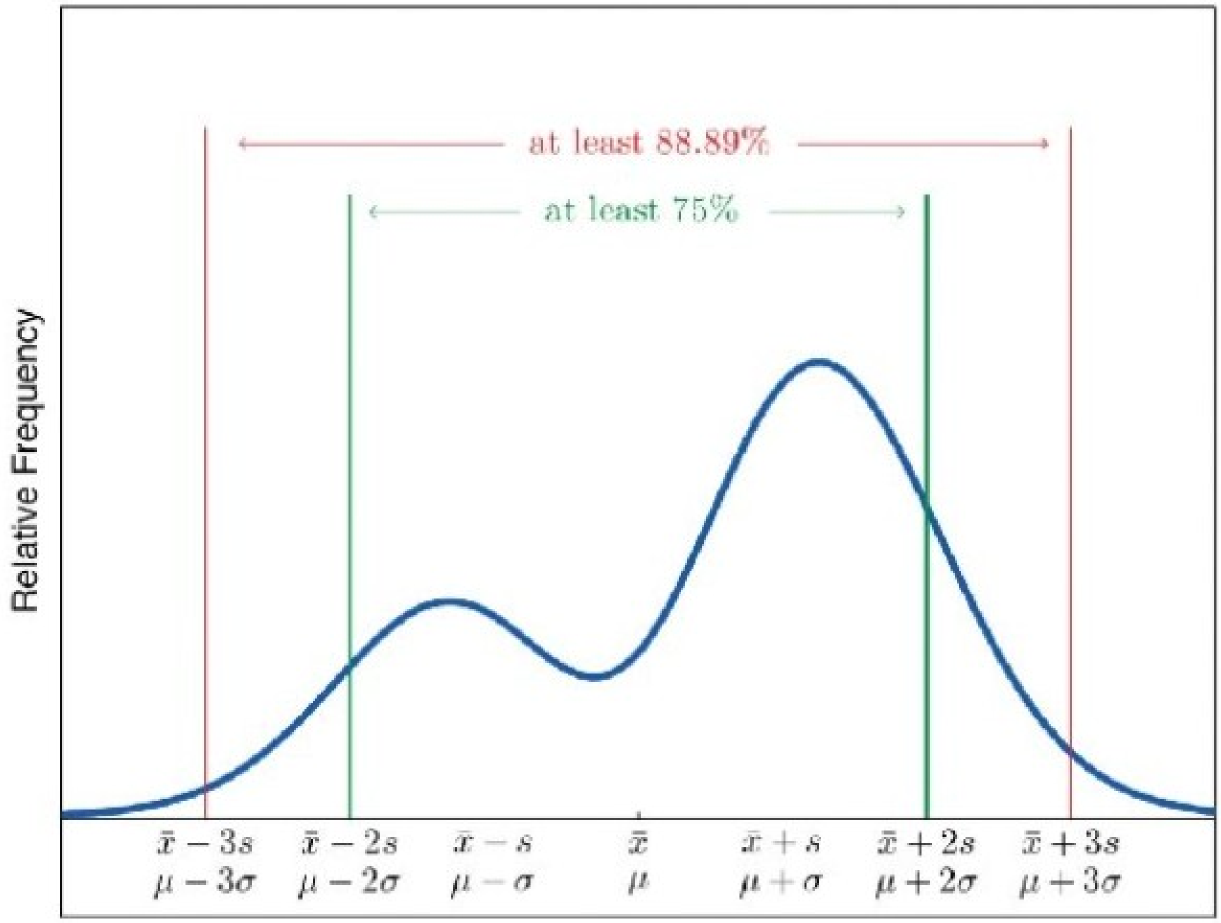


Image src :- https://slidetodoc.com/statistics-lecturer-dr-veronika-alhanaqtah-2-5-chebyshevs/

6.Probability Distributions

Table of Contents

The mathematical tools to describe the distribution of numerical data in populations and samples are probability distributions.It is basically the total listing of the various values the random variable can take along with the corresponding probability of each value. For example pattern of distribution of the machine breakdowns in a manufacturing unit.

- The random variable here is the values which the machine breakdown could assume.
- The probability corresponsding to each value of the breakdown is the relative frequency of occurence of the breakdown.
- The probability distribution for this example is constructed by the actual breakdown pattern observed over a period of time.

Probability distribution can be discreet , continuous and Boolean.

6.1.Binomial Distributions

Table of Contents

The binomial distribution is a widely used probability distribution of a discreet random variable. It plays a major role in quality control and quality assurance function. Manufacturing units use the binomial distribution for defective analysis. It is also used in service organizations like banks and insurance corporations to get an idea of satisfied customers with the service. The parameters of a binomial distribution are n and p where n is the total number of trials, and p is the probability of success in each trial. Its probability distribution function is given by :

$$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0,1\}$$

where $$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0,1\}$$

Uses of Binomial distribution in an industry :-

- Check whether the code written is defective or not.
- Check the hardware parts are defective or not.
- Check the no of defective pages in the report.
- Satisfied customers.

Conditions for applying Binomial Distributions (Bernoulli Process)

- Trials are independent and random.
- There are fixed number of trials (n trials).
- There can be only two possible outcomes of the trials(success or failure)
- The probability of success is uniform through out the trials.

If you want know more about this binomial distribution , please follow the following link

https://medium.com/datadriveninvestor/all-you-need-to-know-about-probability-distribution-ac5aa9c56222

Let's see how can implement this in python with the mentioned example

A washing machine assembly Unit manufactures a single type of washing machines product and sells it to its customers under multiple Brand names through its Channel Partners. From past audit and quality data, it knows that on an average, 10% of all manufactured units since inception have been defective. The manufacturer receives a wholesale order of 100 new washing machines that are to have highest form of quality and no defectie units. However, it is the liability of the Manufacturer to inform the Client about the probability of a few units being defective out of the whole lot. Construct the Binomial Probability Distribution of washing machines that are defective out of the 100.

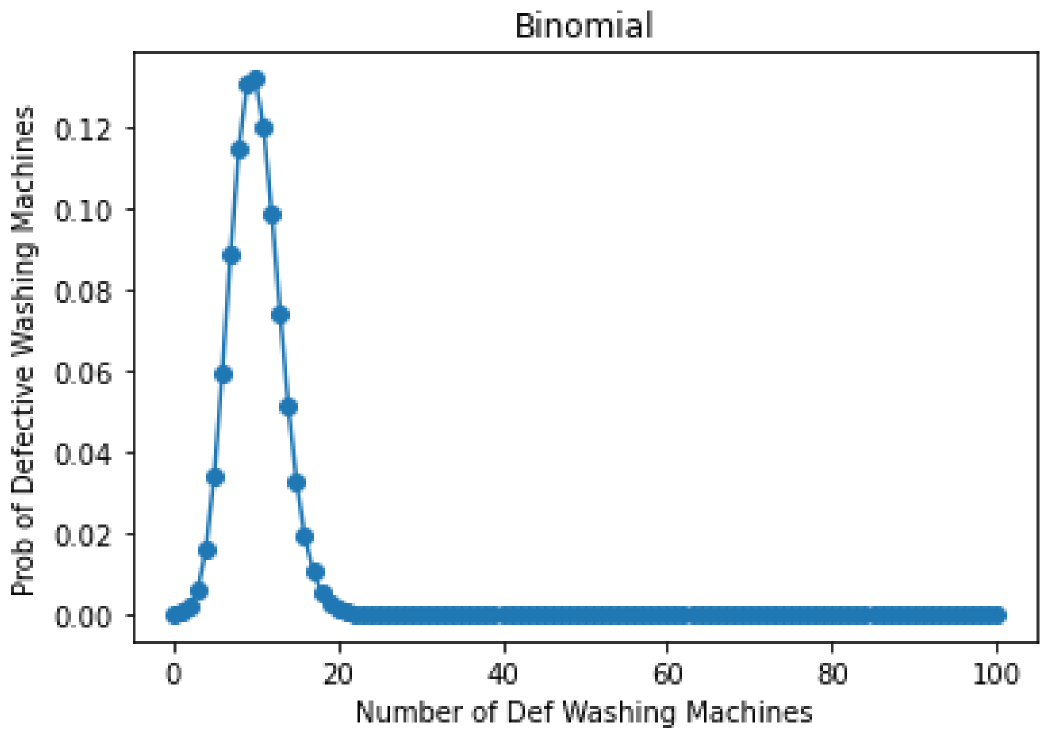
```python
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats

p=0.1
n=100
k=np.arange(0,101)
binomial = stats.binom.pmf(k,n,p)
binomial
```

```
array([2.65613989e-005, 2.95126654e-004, 1.62319660e-003, 5.89160247e-
003,
       1.58745955e-002, 3.38658038e-002, 5.95787289e-002, 8.88952464e-
002,
       1.14823027e-001, 1.30416277e-001, 1.31865347e-001, 1.19877588e-
001,
       9.87880124e-002, 7.43020948e-002, 5.13038273e-002, 3.26824382e-
002,
       1.92917170e-002, 1.05915309e-002, 5.42652508e-003, 2.60219331e-
003,
       1.17098699e-003, 4.95655869e-004, 1.97761685e-004, 7.45188959e-
005,
       2.65646064e-005, 8.97293372e-006, 2.87594029e-006, 8.75800748e-
007,
       2.53704185e-007, 6.99873614e-008, 1.84040839e-008, 4.61751209e-
009,
       1.10627894e-009, 2.53289454e-010, 5.54588020e-011, 1.16199395e-
011,
       2.33116070e-012, 4.48030885e-013, 8.25320051e-014, 1.45783029e-
014,
       2.47021243e-015, 4.01660558e-016, 6.26930501e-017, 9.39585764e-
018,
       1.35243405e-018, 1.87003227e-019, 2.48434239e-020, 3.17150093e-
021,
       3.89096178e-022, 4.58798215e-023, 5.19971310e-024, 5.66417549e-
025,
       5.93044015e-026, 5.96773852e-027, 5.77126976e-028, 5.36320018e-
029,
       4.78857159e-030, 4.10715692e-031, 3.38329018e-032, 2.67604873e-
033,
       2.03181478e-034, 1.48037507e-035, 1.03467074e-036, 6.93430129e-
038,
       4.45432548e-039, 2.74112337e-040, 1.61514003e-041, 9.10692557e-
043,
       4.91059712e-044, 2.53042042e-045, 1.24512751e-046, 5.84566904e-
048,
       2.61611732e-049, 1.11493584e-050, 4.52001018e-052, 1.74104096e-
053,
       6.36345379e-055, 2.20379352e-056, 7.22040612e-058, 2.23416223e-
059,
       6.51630651e-061, 1.78773841e-062, 4.60257857e-064, 1.10905508e-
065,
       2.49390692e-067, 5.21601447e-069, 1.01085552e-070, 1.80740450e-
072,
       2.96669931e-074, 4.44449334e-076, 6.03573170e-078, 7.36963577e-
080,
       8.01047367e-082, 7.65636671e-084, 6.33505520e-086, 4.44565277e-
088,
       2.57271572e-090, 1.17879300e-092, 4.00950000e-095, 9.00000000e-
098,
       1.00000000e-100])
```

```
plt.plot(k,binomial, 'o-')
plt.title('Binomial')
plt.xlabel('Number of Def Washing Machines')
plt.ylabel('Prob of Defective Washing Machines')
```

```
Text(0, 0.5, 'Prob of Defective Washing Machines')
```



6.2 Poisson Distribution

Table of Contents

The Poisson distribution can be used in quality control . The only difference is the binomial distribution deals with defective , but the Poisson distribution deals with the no of defects. Poisson distribution is described in terms of the rate (μ) at which the events happen. An event can occur 0, 1, 2, … times in an interval. The average number of events in an interval is designated λ (lambda). Lambda is the event rate, also called the rate parameter. The probability of observing k events in an interval is given by the equation:

$$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0,1\}$$

Note that the normal distribution is a limiting case of Poisson distribution with the parameter λ→∞. Also, if the times between random events follow an exponential distribution with rate λ, then the total number of events in a time period of length t follows the Poisson distribution with parameter λt.

Examples :-

- A computer is defective is binomial while the type of defects in each defective computer is Poisson.
- Number of defective pages in the Novel is binomial but number of defects per page is Poisson.
- Dissatisfied customers percentage is binomial while number of complaints is Poisson
- Other real life examples include a) no of cars arriving at a highway checkpoint per hour. b) The no of customers visiting the ATM machine per hour.

Conditions for Poisson distribution :-

- The probability of getting exactly one success in a continuous interval such as length, area , time and the like is constant.
- The probability of a success in any one interval is independent of the probability of success occurring in any other interval.
- The probability of getting more than one success in an interval is 0

Please read the following article which describe it in detail :

https://medium.com/datadriveninvestor/all-you-need-to-know-about-probability-distribution-ac5aa9c56222

Lets see the following example how we can implement this is python

If on an average, 6 customers arrive every one minute at a shop during the busy hours of working, a) what is the probability that exactly four customers arrive in a given minute? b) What is the probability that more than three customers will arrive in a given minute?

```
rate=6
n=np.arange(0,20)
poisson = stats.poisson.pmf(n,rate)
poisson

array([2.47875218e-03, 1.48725131e-02, 4.46175392e-02, 8.92350784e-02,
       1.33852618e-01, 1.60623141e-01, 1.60623141e-01, 1.37676978e-01,
       1.03257734e-01, 6.88384890e-02, 4.13030934e-02, 2.25289600e-02,
       1.12644800e-02, 5.19899078e-03, 2.22813891e-03, 8.91255562e-04,
       3.34220836e-04, 1.17960295e-04, 3.93200983e-05, 1.24168732e-
05])
```

```
poisson[4]

0.13385261753998332

poisson[4] + poisson[5] + poisson[6]

0.45509889963594335
```

6.3 Normal distribution

Table of Contents

Normal distribution also called as Gaussian distribution is the most common type of distribution. It is a kind of symmetric distribution where most of the observed values of the random variable is clustered around it's central peak and the other values are equally distributed in both the directions.A normal distribution has a bell-shaped density curve described by its mean μ and standard deviation σ. The density curve is symmetrical, centered about its mean, with its spread determined by its standard deviation showing that data near the mean are more frequent in occurrence than data far from the mean. The probability distribution function of a normal density curve with mean μ and standard deviation σ at a given point x is given by:

$$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0,1\}$$

$$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0,1\}$$

Normal Distribution has the following features :-

- It is a continuous distribution which looks like a bell . It is also termed as "Bell Shaped distribution".
- Mean , median and mode are all equal to each other,
- It is symmetrical about it's mean.
- If the tails of the normal distribution are extended , they will run parallel to the horizontal axis without actual touching it. (asymptotic to the x-axis)
- It has only two parameters namely it's mean and the standard deviation.
- Approximately 68% of the data in a bell shaped distribution is within the 1 standard deviation of the mean
- Approximately 95% of the data in a bell shaped distribution lies within two standard deviations of the mean.
- Approximately 99.7% of the data in a bell-shaped distribution lies within three standard deviations of the mean,

Please read the following article which describe it in detail :

https://medium.com/datadriveninvestor/all-you-need-to-know-about-probability-distribution-ac5aa9c56222

Lets see the following example how we can implement this is python

The mean salaries of Data Scientists working in Amsterdam, Netherlands is calculated to be 7,00,000 EUR with a standard deviation of 90,000 INR. The random variable salary of Data Scientists follows a normal distribution.

- What is the probability that a Data Scientist in Amsterdam has a salary more than 10,00,000 EUR?
- What is the probability that a Data Scientist in Amsterdam has a salary between 6,00,000 & 9,00,000 EUR?
- What is the probability that a Data Scientist in Amsterdam has a salary less than 4,00,000 EUR?

```
z=(1000000-700000)/90000
z

3.3333333333333335

1 - stats.norm.cdf(3.333)

0.00042957471189908336

z1= (600000-700000)/90000
z2= (900000-700000)/90000

stats.norm.cdf(z2) - stats.norm.cdf(z1)

0.8536055914064735

z = (400000-70000)/90000

z

3.6666666666666665

stats.norm.cdf(-0.333)

0.36956714157750203
```

6.4 Uniform distribution

Table of Contents

The probability distribution function of the continuous uniform distribution is:

$$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0,1\}$$

Since any interval of numbers of equal width has an equal probability of being observed, the curve describing the distribution is a rectangle, with constant height across the interval and 0 height elsewhere. Since the area under the curve must be equal to 1, the length of the interval determines the height of the curve.

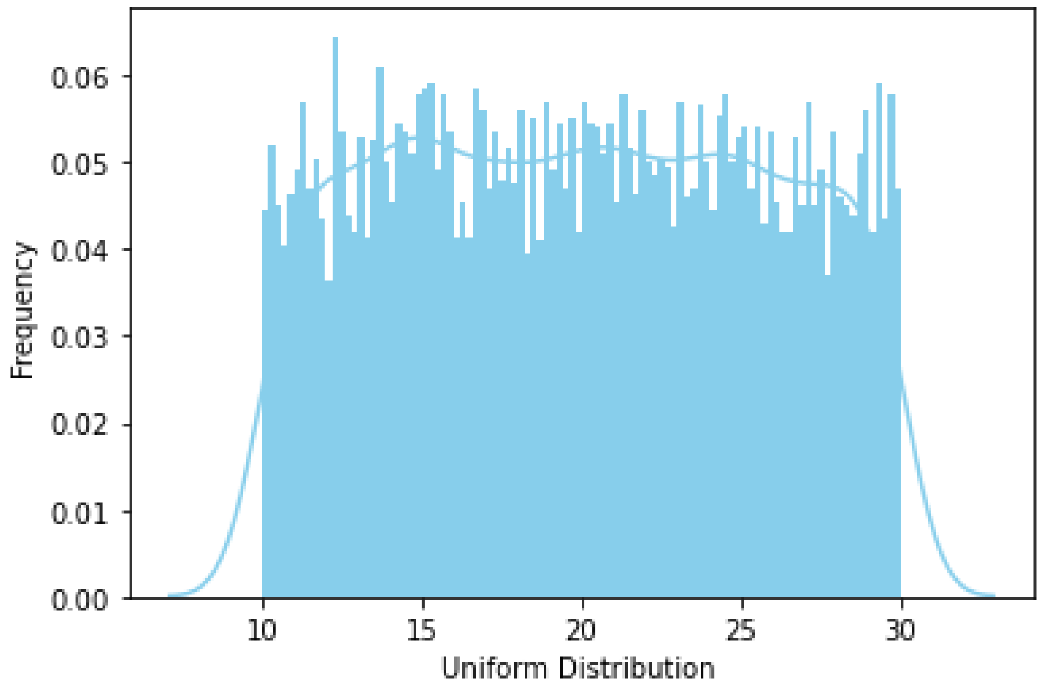
Let'see how we can implement this in python

```
# import uniform distribution
from scipy.stats import uniform

# random numbers from uniform distribution
n = 10000
start = 10
width = 20
data_uniform = uniform.rvs(size=n, loc = start, scale=width)

ax = sns.distplot(data_uniform,
                  bins=100,
                  kde=True,
                  color='skyblue',
                  hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Uniform Distribution ', ylabel='Frequency')
```

```
[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Uniform Distribution ')]
```



6.5 Gamma distribution

Table of Contents

The gamma distribution is a two-parameter family of continuous probability distributions. While it is used rarely in its raw form but other popularly used distributions like exponential, chi-squared, erlang distributions are special cases of the gamma distribution. The gamma distribution can be parameterized in terms of a shape parameter α=k and an inverse scale parameter β=1/θ, called a rate parameter., the symbol Γ(n) is the gamma function and is defined as (n−1)! :

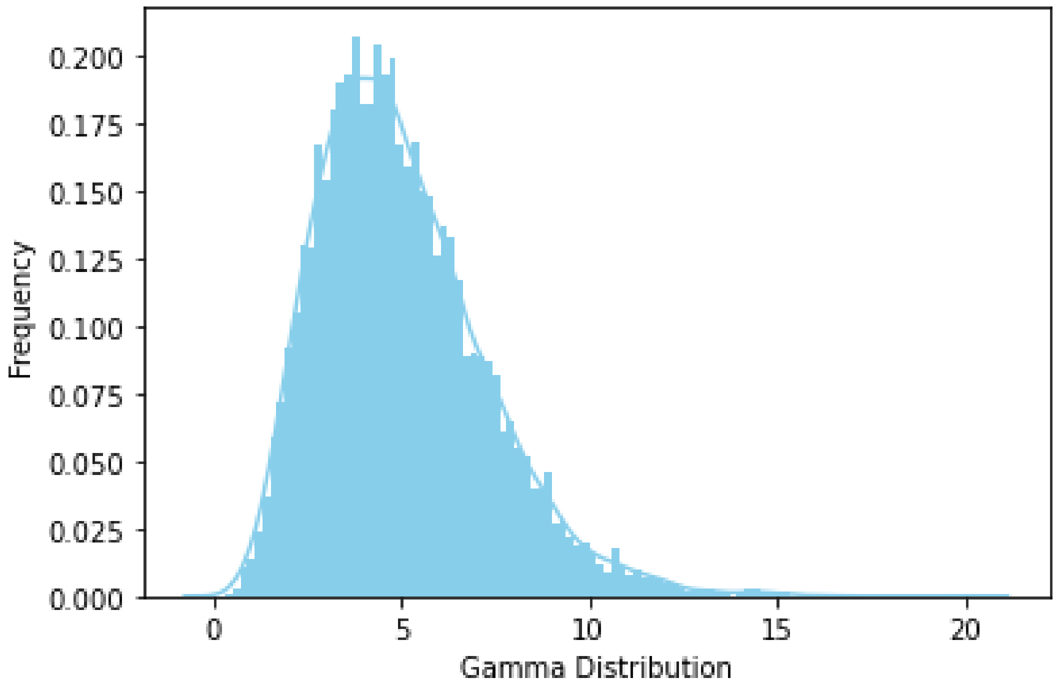
$$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0,1\}$$

We can generate a gamma distributed random variable using scipy.stats module's gamma.rvs() method which takes shape parameter a as its argument. When a is an integer, gamma reduces to the Erlang distribution, and when a=1 to the exponential distribution. To shift distribution use the loc argument, to scale use scale argument, size decides the number of random variates in the distribution. If we want to maintain reproducibility, include a random_state argument assigned to a number.

```
from scipy.stats import gamma
data_gamma = gamma.rvs(a=5, size=10000)

ax = sns.distplot(data_gamma,
                  kde=True,
                  bins=100,
                  color='skyblue',
                  hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Gamma Distribution', ylabel='Frequency')

[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Gamma Distribution')]
```



6.6 Exponential distribution

Table of Contents

The exponential distribution describes the time between events in a Poisson point process, i.e., a process in which events occur continuously and independently at a constant average rate. It has a parameter λ called rate parameter, and its equation is described as :

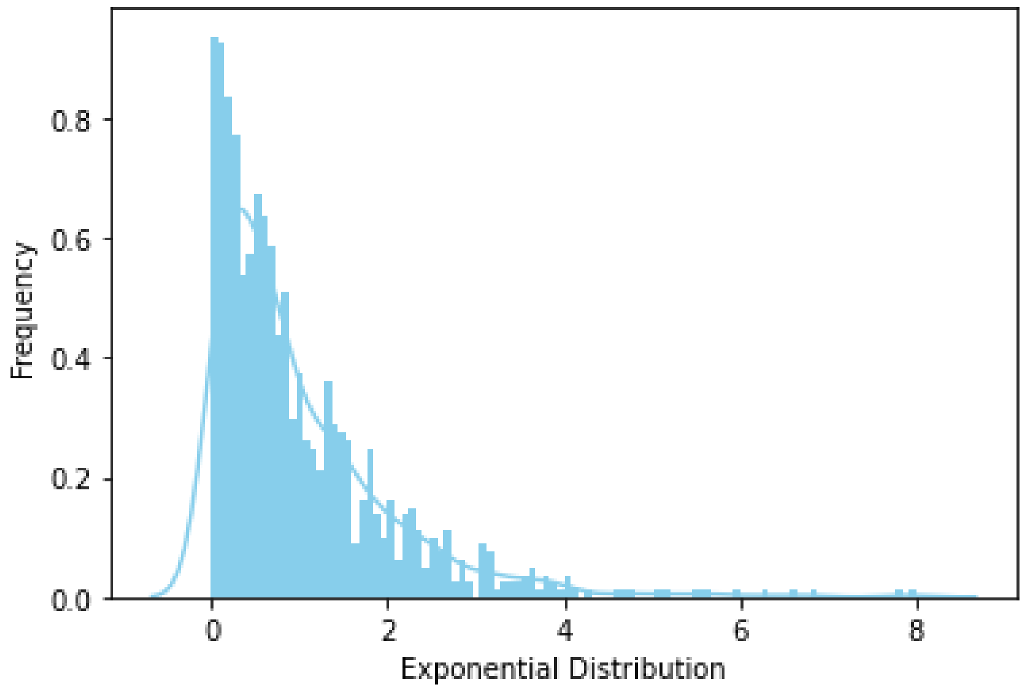
$$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0,1\}$$

We can generate an exponentially distributed random variable using scipy.stats module's expon.rvs() method which takes shape parameter scale as its argument which is nothing but 1/lambda in the equation. To shift distribution use the loc argument, size decides the number of random variates in the distribution. If we want to maintain reproducibility, include a random_state argument assigned to a number.

```
from scipy.stats import expon
data_expon = expon.rvs(scale=1,loc=0,size=1000)

ax = sns.distplot(data_expon,
                  kde=True,
                  bins=100,
                  color='skyblue',
                  hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Exponential Distribution', ylabel='Frequency')

[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Exponential Distribution')]
```



6.7 Bernoulli distribution

Table of Contents

A Bernoulli distribution has only two possible outcomes, namely 1 (success) and 0 (failure), and a single trial, for example, a coin toss. So the random variable X which has a Bernoulli distribution can take value 1 with the probability of success, p, and the value 0 with the probability of failure, q or 1−p. The probabilities of success and failure need not be equally likely. The Bernoulli distribution is a special case of the binomial distribution where a single trial is conducted (n=1). Its probability mass function is given by:

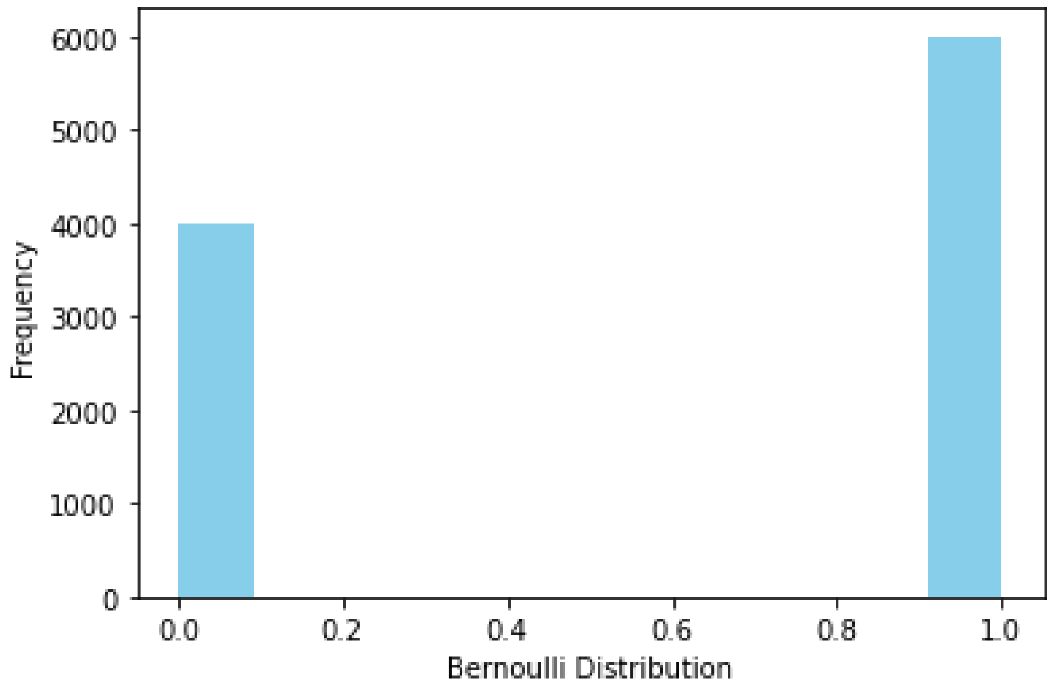
$$f(k;p) = p^k(1-p)^{1-k} \quad \text{for } k \in \{0, 1\}$$

We can generate a bernoulli distributed discrete random variable using scipy.stats module's bernoulli.rvs() method which takes p (probability of success) as a shape parameter. To shift distribution use the loc parameter. size decides the number of times to repeat the trials. If we want to maintain reproducibility, include a random_state argument assigned to a number.

```
from scipy.stats import bernoulli
data_bern = bernoulli.rvs(size=10000,p=0.6)

ax= sns.distplot(data_bern,
                 kde=False,
                 color="skyblue",
                 hist_kws={"linewidth": 15,'alpha':1})
ax.set(xlabel='Bernoulli Distribution', ylabel='Frequency')

[Text(0, 0.5, 'Frequency'), Text(0.5, 0, 'Bernoulli Distribution')]
```



7. Expected Value ,Variance and Covariance

Table of Contents

Expected Value

In probability, the average value of some random variable X is called the expected value or the expectation.The expected value uses the notation E with square brackets around the name of the variable; for example:

E[X]

It is calculated by the following formula

E[X] = sum(x1 * p1, x2 * p2, x3 * p3, ..., xn * pn)

The expected value can be calculated as the sum of all values multiplied by the reciprocal of the number of values.

E[X] = sum(x1, x2, x3, ..., xn) . 1/n

The expected value is a function of the probability distribution of the observed value in our population. The sample mean of our sample is the observed mean value of our data. If the experiment has been designed correctly, the sample mean should converge to the expected value as more and more samples are included in the analysis.It is confusing because mean, average, and expected value are used interchangeably.

Mean can be denoted in the following ways :

- Greek letter mu can be used for this

mu = sum(x1, x2, x3, ..., xn) . 1/n

- x is the vector of observations and P(x) is the calculated probability for each value.

mu = sum(x . P(x)

- When calculated for a specific variable, such as x, the mean is denoted as a lower case variable name with a line above, called x-bar.

_ x = sum from 1 to n (xi) . 1/n

```
from numpy import mean
example = np.linspace(0,8)
result = mean(example)
result
```

```
4.0
```

```
from numpy import array
M = array([[1,2,3,4,5,6],[1,2,3,4,5,6]])
print(M)
```

```
col_mean = mean(M, axis=0)
print(col_mean)
row_mean = mean(M, axis=1)
print(row_mean)
```

```
[[1 2 3 4 5 6]
 [1 2 3 4 5 6]]
[1. 2. 3. 4. 5. 6.]
[3.5 3.5]
```

Variance

The variance of some random variable X is a measure of how much values in the distribution vary on average with respect to the mean.The variance is denoted as the function Var() on the variable.

Var[X]

Variance is calculated as the average squared difference of each value in the distribution from the expected value. Or the expected squared difference from the expected value.

Var[X] = sum (p(x1) . (x1 - E[X])^2, p(x2) . (x2 - E[X])^2, ..., p(x1) . (xn - E[X])^2)

If the probability of each example in the distribution is equal, variance calculation can drop the individual probabilities and multiply the sum of squared differences by the reciprocal of the number of examples in the distribution.

Var[X] = sum ((x1 - E[X])^2, (x2 - E[X])^2, ...,(xn - E[X])^2) . 1/n

The sample variance is denoted by the lower case sigma with a 2 superscript indicating the units are squared,not that you must square the final value .The sum of the squared differences is multiplied by the reciprocal of the number of examples minus 1 to correct for a bias.

sigma^2 = sum from 1 to n ((xi - mu)^2) . 1 / (n - 1)

```
from numpy import var
v = array([1,2,3,4,5,6])
print(v)
result = var(v, ddof=1)
print(result)
```

```
[1 2 3 4 5 6]
3.5
```

```
M = array([[1,2,3,4,5,6],[1,2,3,4,5,6]])
print(M)
col_mean = var(M, ddof=1, axis=0)
print(col_mean)
row_mean = var(M, ddof=1, axis=1)
print(row_mean)
```

```
[[1 2 3 4 5 6]
 [1 2 3 4 5 6]]
[0. 0. 0. 0. 0. 0.]
[3.5 3.5]
```

The example below demonstrates how to calculate the sample standard deviation for the rows and columns of a matrix.

```
from numpy import array
from numpy import std
M = array([[1,2,3,4,5,6],[1,2,3,4,5,6]])
print(M)
col_mean = std(M, ddof=1, axis=0)
print(col_mean)
row_mean = std(M, ddof=1, axis=1)
print(row_mean)
```

```
[[1 2 3 4 5 6]
 [1 2 3 4 5 6]]
[0. 0. 0. 0. 0. 0.]
[1.87082869 1.87082869]
```

Covariance

In probability, covariance is the measure of the joint probability for two random variables. It describes how the two variables change together.It is denoted as the function cov(X, Y), where X and Y are the two random variables being considered.

cov(X,Y)

Covariance is calculated as expected value or average of the product of the differences of each random variable from their expected values, where E[X] is the expected value for X and E[Y] is the expected value of y.

cov(X, Y) = E[(X - E[X]) . (Y - E[Y])]

The covariance can be calculated as the sum of the difference of x values from their expected value multiplied by the difference of the y values from their expected values multiplied by the reciprocal of the number of examples in the population.

cov(X, Y) = sum (x - E[X]) * (y - E[Y]) * 1/n

In statistics, the sample covariance can be calculated in the same way, although with a bias correction, the same as with the variance.

cov(X, Y) = sum (x - E[X]) * (y - E[Y]) * 1/(n - 1)

```
from numpy import cov
x = array([1,2,3,4,5,6,7,8,9])
print(x)
```

```
y = array([9,8,7,6,5,4,3,2,1])
print(y)
Sigma = cov(x,y)[0,1]
print(Sigma)
```

```
[1 2 3 4 5 6 7 8 9]
[9 8 7 6 5 4 3 2 1]
-7.5
```

The covariance can be normalized to a score between -1 and 1 to make the magnitude interpretable by dividing it by the standard deviation of X and Y. The result is called the correlation of the variables, also called the Pearson correlation coefficient, named for the developer of the method.

r = cov(X, Y) / sX sY

```
from numpy import corrcoef
x = array([1,2,3,4,5,6,7,8,9])
print(x)
y = array([9,8,7,6,5,4,3,2,1])
print(y)
Sigma = corrcoef(x,y)
print(Sigma)
```

```
[1 2 3 4 5 6 7 8 9]
[9 8 7 6 5 4 3 2 1]
[[ 1. -1.]
 [-1.  1.]]
```

8.Degree of Freedom

Table of Contents

In mechanics, a particle which moves in a plane has “2 DOF”: at each point in time, two parameters (the x=y-coordinates) define the location of the particle. If the particle moves about in space, it has “3 DOF”: the x=y=z-coordinates.

In statistics, a group of n values has n DOF. If we look only at the shape of the distribution of the values, we can subtract from each value the sample mean. Then, the remaining data only have n -1 DOF. (This is clearest for n = 2: if we know the mean value and the value of sample1, then we can calculate the value of sample2 by val2 = 2 * mean - val1.)

The case becomes more complex when we have many groups. For example,there is an example with 22 patients, divided into 3 groups. In the analysis of variance (ANOVA), the DOFs in this example are divided as follows:

• 1 DOF for the total mean value.

• 2 DOF for the mean value of each of the three groups (remember, if we know the mean values of two groups and the total mean, we can calculate the mean value of the third group).

• 19 DOF (= 22-1-2) are left for the residual deviations from the group means.

Degrees of freedom: Roughly, the minimum amount of data needed to calculate a statistic. More practically, it is a number, or numbers, used to approximate the number of observations in the data set for the purpose of determining statistical significance.

9.Hypothesis Testing

Table of Contents

The most common use of statistical tests are to assess whether an observation for e.g finding the relationship between features and coming to a conclusion that the relationship is either a statistical chance or is it real. Some of the important points :-

- Null Hypothesis says that there is no relationship while alternate hypothesis says that there is a significant relationship
- The data showing the apparent relationship have certain characteristics such as central values, spread , shape of the curve , direction of the speed.
- Statistical techniques are used to asess the probability of collecting such data from real world data if there was no such relationship in the real world.
- The probability is expressed as p value. Usually p less than 0.05 % shows that there is the relationship between features.
- P value is linked with the confidence level which is 95 % i.e the confidence in rejecting the null hypothesis.

9.1 Statistical Hypothesis Testing

Table of Contents

Statistics helps us in the interpretation of data which makes the data really interesting. We start creating questions and interpret the results. The statistical methods are used that provide confidence or likelihood about the answers. In general this class of methods is called statistical hypothesis testing or significance tests. In statistics, a hypothesis test calculates some quantity under a given assumption. The result of the test allows us to interpret whether the assumption holds or whether the assumption has been violated. Two concrete examples that we will use a lot in machine learning are:

- A test that assumes that data has a normal distribution.
- A test that assumes that two samples were drawn from the same underlying population distribution.

The assumption of a statistical test is called the null hypothesis, or hypothesis zero (H0 for short). It is often called the default assumption, or the assumption that nothing has changed(there is no relationship).H1 is really a short hand for some other hypothesis.

- Hypothesis 0 (H0): Assumption of the test fails to be rejected.

- Hypothesis 1 (H1): Assumption of the test does not hold and is rejected at some level of significance.

9.2 Statistical Test Intepretation

Table of Contents

The results of a statistical hypothesis test must be interpreted for us to start making claims.There are two common forms that a result from a statistical hypothesis test may take,and they must be interpreted in different ways. They are the p-value and critical values.

Interpret the p-value

A statistical hypothesis test may return a value called p or the p-value. This is a quantity that we can use to interpret or quantify the result of the test and either reject or fail to reject the null hypothesis. This is done by comparing the p-value to a threshold value chosen beforehand called the significance level.

- p-value <= alpha: signicant result, reject null hypothesis (H1).
- p-value > alpha: not signicant result, fail to reject the null hypothesis (H0).

The significance level can be inverted by subtracting it from 1 to give a confidence level of the hypothesis given the observed sample data. confidence level = 1-significance level

The p-value is a measure of how likely the data sample would be observed if the null hypothesis were true. It is represented as

Pr(data|hypothesis)

9.3 Errors in Statistical Tests

Table of Contents

There are two types of errors in statistical tests

Type 1 Error : Reject H0 when it is true

- The incorrect rejection of a true null hypothesis, called a false positive.
- We think the new procedure introduced as part of process improvement has increased productivity but in long run the productivity remains same as it was before the procedure was introduced
- While preparing to model loan status, you find strong link between income and loan status in the sample data and believe it to be true in population while it is not so
- Significance level (α) or Type 1 error rate: is the probability of making this type of error I (P value)
- This P value is usually set to 0.05 as a standard. This translates to 5% chance of incorrectly rejecting H0

Type 2 error:Failing to reject H0 when it is false

- The incorrect failure of rejecting a false null hypothesis, called a false negative.

- We think the new procedure introduced as part of process improvement has not had any effect but if we had used it long enough we would have noticed the productivity actually increased compare to before the procedure was introduced
- While preparing to model loan status, we do not find strong link between income and loan status in the sample data and believe there is no relation between them in population while a larger sample would have shown the relation
- The value β is the probability of a type 2 error or type 2 error rate

Lets summarize our understanding in the following table where column represents the decision on the null hypothesis and row represents the state of null hypothesis

Confusion Matrix	Predicted Positive	Predicted Negative
Actual PositiveH0 is false	True Positive(Correctly rejected H0)Prob = 1-β	False Negative(failed to reject H0)β error / Type II error
Actual Negative H0 is True	False Positive(incorrectly rejected H0)α error / Type I error	True Negative(correctly accepted H0)Prob = 1-α

9.4 Power of Hypothesis Tests

Table of Contents

The probability a test will reject H0 when it is supposed to. The estimated probability (power of test) is a function of sample size, variability, level of significance, and the difference between the null and alternative hypotheses.

Power: 1−β: probability of correctly rejecting H0 when it is fails

The power of a statistical test is function of

- It increases as |μ 0- μ 1 | a.k.a. effect (magnitude of the differences in means) increases
- It increases as the sample size n goes up
- It goes up as standard deviation of the sample goes down
- It goes up as α value increases (probability of committing type 1 error increases)

To keep the chances of making a correct decision high

- the probability of a Type I error (α, the level of significance of a hypothesis test) is kept low, usually 0.05 or less,
- the power of the test (1 β, the probability of rejecting H0 when H1 is true) is kept high, usually 0.8 or more.
- In order to achieve a desirable power for a fixed level of significance, the sample size will generally need to increase
- Increase in sample size will increasingly reflect the characteristics of the population and thus help in addressing both Type I and Type II error simultaneously

Power analysis is the process of estimating one of the four parameters that the power of a test is dependent on, given values for three other parameters. It helps in design of experiment (sample size, power of test likely) and in analysis of the predictions of a model using hypothesis testing

Effect size, a parameter we need to know for power analysis

1.Measure of the minimum magnitude of the signal (difference between an observation and the Null distribution) that can be dete cte d by the test. For e.g.difference between the sample mean and population mean

2.Effect size is calculated using a specific statistical measure, such as Pearson's correlation coefficient for the relationshi p b etween variables or Cohen's d for the difference between groups. It describes the difference in means in terms of the number of standard deviations

9.5 Sensitivity and Specificity

Table of Contents

One of the most commonly used terms in statistical analysis are sensitivity and specificity. Please have a look at some of the definitions

- Sensitivity :- It is called as power. Proportion of the positives that are correctly identified by a test (= probability of a positive test, given the patient is ill). It is calculated by TP/(TP+FN)
- Specificity :- Proportions of negative which are correctly identified by a test(=probability of a negative test , given that the patient is well). It is calualted by TN/(FP+TN)
- Positive Prediction Value (PPV):- Proportion of patients with positive test results which are correctly diagonised.
- Negative Prediction Value (NPV):- Proportion of patients with negative test results which are correctly diagonised.

10. Central Limit Theorem

Table of Contents

Central Limit theorem states that the sampling distribution of the sample means approaches a normal distribution as the sample size gets larger — no matter what the shape of the population distribution.

In other words we can also say that if we sample batches of data from any distribution and take the mean of each batch. Then the distribution of the means is going to resemble a Gaussian distribution. (Same goes for taking the sum).

It can be used in the scenario where we have to find following :-

- We want to find the probability that the mean is greater than a certain number
- We want to find the probability that the mean is less than a certain number
- We want to find the probability that the mean is between a certain set of numbers either side of the mean

Let's look into the details with the following example:-

```
pima_df =
pd.read_csv("/kaggle/input/pima-indians-diabetes-database/diabetes.csv
")
pima_df.head()

   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin
BMI  \
0            6      148             72             35        0  33.6

1            1       85             66             29        0  26.6

2            8      183             64              0        0  23.3

3            1       89             66             23       94  28.1

4            0      137             40             35      168  43.1


   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1

#There are 0 values in the dataset in the
Glucose,BloodPressure,SkinThickness, Insulin and BMI, we need to
replace them with the NAN

pima_df[["Glucose","BloodPressure","SkinThickness","Insulin","BMI"]]=p
ima_df[["Glucose","BloodPressure","SkinThickness","Insulin","BMI"]].re
place(0, np.NaN)

pima_df.isnull().any()

Pregnancies                 False
Glucose                      True
BloodPressure                True
SkinThickness                True
Insulin                      True
BMI                          True
DiabetesPedigreeFunction    False
Age                         False
Outcome                     False
dtype: bool

#Checking for the missing values in the dataset

pima_df.isna().sum()

Pregnancies                  0
Glucose                      5
BloodPressure               35
```

```
SkinThickness               227
Insulin                     374
BMI                          11
DiabetesPedigreeFunction      0
Age                           0
Outcome                       0
dtype: int64
```

```
pima_df.shape
```

```
(768, 9)
```

```
#Replacing the null values with the mean and median respectively

pima_df['Glucose'].fillna(pima_df['Glucose'].mean(), inplace = True)
pima_df['BloodPressure'].fillna(pima_df['BloodPressure'].mean(),inplac
e=True)
pima_df['SkinThickness'].fillna(pima_df['SkinThickness'].median(),inpl
ace=True)
pima_df['Insulin'].fillna(pima_df['Insulin'].median(),inplace=True)
pima_df['BMI'].fillna(pima_df['BMI'].median(),inplace=True)
```

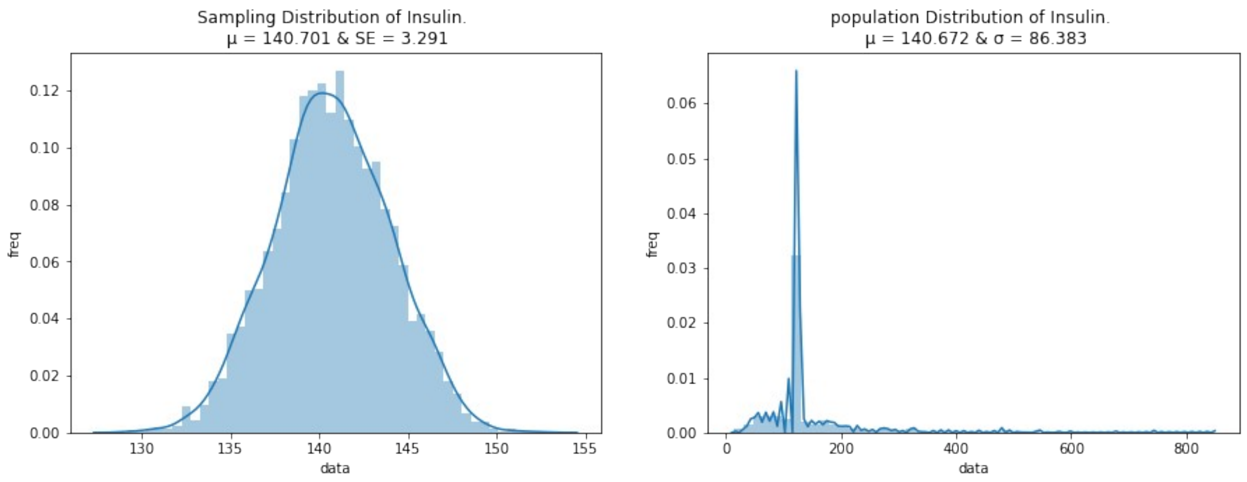
```
series1 = pima_df.Insulin
series1.dtype
```

```
dtype('float64')
```

```
def central_limit_theorem(data,n_samples = 1000, sample_size = 500,
min_value = 0, max_value = 768):
    b = {}
    for i in range(n_samples):
        x = np.unique(np.random.randint(min_value, max_value, size =
sample_size)) # set of random numbers with a specific size
        b[i] = data[x].mean()   # Mean of each sample
    c = pd.DataFrame()
    c['sample'] = b.keys()  # Sample number
    c['Mean'] = b.values()  # mean of that particular sample
    plt.figure(figsize= (15,5))

    plt.subplot(1,2,1)
    sns.distplot(c.Mean)
    plt.title(f"Sampling Distribution of Insulin. \n \u03bc =
{round(c.Mean.mean(), 3)} & SE = {round(c.Mean.std(),3)}")
    plt.xlabel('data')
    plt.ylabel('freq')

    plt.subplot(1,2,2)
    sns.distplot(data)
    plt.title(f"population Distribution of Insulin. \n \u03bc =
{round(data.mean(), 3)} & \u03C3 = {round(data.std(),3)}")
    plt.xlabel('data')
    plt.ylabel('freq')
    plt.show()
central_limit_theorem(series1,n_samples = 5000, sample_size = 500)
```



11. Tests of mean of numerical data

Table of Contents

In this section , we are going to talk about the hypothesis tests for the mean value of groups, and how to implement them :

- Comparison of one group with a fixed value
- Comparison of two groups with respect to each other.
- Comparison of three or more groups with each other

In all the scenarios mentioned above we distinguish between the two cases. Two things we have to keep in mind :-

- If the data is normally distributed then we can use the parametric tests like paired t -test , unpaired t -test and pearson correlation etc.
- If the data is not normally distributed then we can use the non parametric tests like spearman correlation , wilcoxon rank sum test etc.

11.1 One Sample T Test

Table of Contents

If we have to check the mean value of the normally distributed data against a reference value, we use the one sample t test which is based on the t- distribution.

If the mean and the standard deviation of a normally distributed population is known, we could calculate the corresponding standard error, and use values from the normal distribution to determine how likely it is to find a certain value.

Formulate the null hypothesis

- H0 - The difference in mean between sample BP column and population mean for BP is a statistical fluctuation. The given data represents the population distribution on the BP column
- H1 - The difference in mean between sample BP column and population mean is significant. The difference is too high to be result of statistical fluctuation
- If statistical tests result in rejecting H0, then building a model on the given sample data and expecting it to generalize may be a mistake

```
import scipy.stats as st
Mu = 72.4
# Std = ?  Population standard deviatin is unknown

x = pima_df['BloodPressure']  # Storing values in a list to avoid long
names
est_pop_std = np.sqrt(np.sum(abs(x - x.mean())**2) / (pima_df.size -
1))     #  sqrt(sum(xi - Xbar)^2 / (n -1))

sample_avg_bp =(pima_df['BloodPressure']).mean()

std_error_bp = est_pop_std / np.sqrt(pima_df.size) # Standard dev of
the sampling mean distribution... estimated from population

T_Statistic = (( sample_avg_bp - Mu) / std_error_bp)

pvalue = st.t.sf(np.abs(T_Statistic), pima_df.size-1)*2
print("Estimated Pop Stand Dev" , est_pop_std)
print("Sample Avg BP : " , sample_avg_bp)
print("Standard Error: " , std_error_bp)
print("T Statistic" , T_Statistic)
print("Pval" , pvalue)

if pvalue > 0.05:
     print('Samples are likely drawn from the same distributions (fail
to reject H0)')
else:
     print('Samples are likely drawn from different distributions
(reject H0)')

Estimated Pop Stand Dev 4.029780981170755
Sample Avg BP :  72.40518417462486
Standard Error:  0.048470731963628534
T Statistic 0.10695474185841736
Pval 0.9148279889570086
```

```
Samples are likely drawn from the same distributions (fail to reject
H0)
```

11.2 Wilcoxon Signed Rank Sum Test

Table of Contents

If the data is not normally distributed , then one sample T test will not work. Instead we perform the non paremetric test on the mean value . We can do this by performing a Wilcoxon signed rank sum test. This Method has three steps

- a. Calculate the difference between each observation and the value of interest.
- a. Ignoring the signs of the differences, rank them in order of magnitude.
- a. Calculate the sum of the ranks of all the negative (or positive) ranks, corresponding to the observations below (or above) the chosen hypothetical value.

The default assumption for the test, the null hypothesis, is that the two samples have the same distribution. In order for the Wilcoxon signed-rank test results to be trusted, the following assumptions need to be met:

- The dependent variable (DV) must be continuous which is measured on an ordinal or continuous scale
- The paired observations are randomly and independently drawn
- The paired observations come from the same population
- If any of these assumptions are violated, a different test should be used.

- Fail to Reject H0: Sample distributions are equal.

- Reject H0: Sample distributions are not equal.

```
# Wilcoxon signed-rank test
from numpy.random import seed
from numpy.random import randn
from scipy.stats import wilcoxon
# seed the random number generator
seed(1)
# generate two independent samples
data1 = 5 * randn(100) + 50
data2 = 5 * randn(100) + 51
# compare samples
stat, p = wilcoxon(data1, data2)
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
alpha = 0.05
if p > alpha:
    print('Same distribution (fail to reject H0)')
else:
    print('Different distribution (reject H0)')

Statistics=1886.000, p=0.028
Different distribution (reject H0)
```

11.3 Paired T Test

Table of Contents

In the comparison of two groups with each other, two cases have to be distinguished.In the first case, two values recorded from the same subject at different times are compared to each other.If the groups come from a single population (e.g. measuring before and after an experimental treatment), perform a paired t-test.

PAIRED SAMPLE T-TEST ASSUMPTIONS

In order for the paired sample t-test results to be trusted, the following assumptions need to be met:

- The dependent variable (DV) must be continuous which is measured on an interval or ratio scale
- The observations are independent
- The DV should be approximately normally distributed
- The paired sample t-test is robust to this violation. If there is a violation of normality, as long as it’s not in a major violation the test results can be considered valid
- The DV should not contain any significant outliers

For example, the size of students when they enter primary school and after their first year, to check if they have grown. Since we are only interested in the difference in each subject between the first and the second measurement, this test is called paired t-test, and is essentially equivalent to a onesample t-test for the mean difference

```
np.random.seed(1234)
data = np.random.randn(10)+0.1
data1 = np.random.randn(10)*5 # dummy data
data2 = data1 + data # same group-difference as "data"
stats.ttest_1samp(data, 0)
stats.ttest_rel(data2, data1)

Ttest_relResult(statistic=-0.1245849229873135,
pvalue=0.9035904508547089)
```

11.4 Two Sample T Test

Table of Contents

If the groups come from two different populations (e.g. two different species, or people from two separate cities), perform a two-sample t-test (a.k.a. independent t-test).An unpaired t-test, or t-test for two independent groups, compares two groups. An example would be the comparison of the effect of two medications given to two different groups of patients.

```
# Tests whether the means of two independent samples are significantly
different.

# Pima Indians Dataset has many missing values in multiple columns.
Let us replace the missing values with median. Does this
# step of handling missing values modify the distribution so much that
statistically it is no more equivalent of original data?

pima_df_mod = pima_df.copy()


pima_df_mod['BloodPressure'] =
pima_df_mod['BloodPressure'].mask(pima_df['BloodPressure'] ==
0,pima_df['BloodPressure'].median())

from scipy.stats import ttest_ind

stat, pvalue = ttest_ind(pima_df_mod['BloodPressure'] ,
pima_df['BloodPressure'])
print("compare means", pima_df_mod['BloodPressure'].mean() ,
pima_df['BloodPressure'].mean())
print("Tstatistic , Pvalue", stat, pvalue)

if pvalue > 0.05:
    print('Samples are likely drawn from the same distributions (fail
to reject H0)')
else:
    print('Samples are likely drawn from different distributions
(reject H0)')
```

```
compare means 72.40518417462486 72.40518417462486
Tstatistic , Pvalue 0.0 1.0
Samples are likely drawn from the same distributions (fail to reject
H0)
```

11.5 Mann WHitney Test

Table of Contents

When the two groups are not normally distributed we have to resort to a nonparametric test. The most common nonparametric test for the comparison of two independent groups is the Mann–Whitney(–Wilcoxon) test.

The following are the assumptions of the Mann Whiteney Test :-

- All the observations from both the group are independant of each other.

- The values present in the dependent variable should be in the ordinal form.
- The independent variable should be two independent, categorical groups.
- The null hypothesis always remains the same i.e. there is no significant difference between the two samples
- It is applied when the two groups are not normally distributed but should have the same curve shape which means if one is right skewed then the other should also be right skewed.

This Test is not affected by the outliers as it uses the median but not mean for the test.

Steps to perform the Mann Whitney Test

1. Get two samples of data ,let'say sample 1 and sample 2
2. Obtain the first observation from sample 1 and compare it with observations in sample 2. Count the number of observations in Sample 2 that are smaller than that and equal to it. For, example, 8 observations in sample 2 are smaller than the first observation in sample 1 and 2 equal then out U statistics for this sample: 8 + 2(1/2) = 9
3. Repeat Step 2 for all observations in sample 1
4. Add up all of your totals from Steps 2 and 3. This is our rank sum.
5. U statistics is calculated by

$$U_1 = n_1 n_2 + \frac{n_1(n_1+1)}{2} - R_1 \quad U_1 = n_1 n_2 + \frac{n_1(n_1+1)}{2} - R_1$$

$$U_2 = n_1 n_2 + \frac{n_2(n_2+1)}{2} - R_2 \quad U_2 = n_1 n_2 + \frac{n_2(n_2+1)}{2} - R_2$$

where:

- n1: number of samples in sample 1
- n2: number of samples in sample 2
- R1: Rank sum of sample 1
- R2: Rank sum of sample 2

1. Now, our test statistic (U) will be smaller of U1 and U2.
2. Now, we look to the critical values in the table with respect to n1 and n2 (take it U0).
   - if U <= U0 : we reject the null hypothesis.
   - else, we do not reject the null hypothesis.

In python ,it can be implmented by the following

u_statistic, pVal = stats.mannwhitneyu(group1, group2)

```
#import necessary libraries

from numpy.random import seed
from numpy.random import randn
from scipy.stats import mannwhitneyu
from numpy.random import rand

# Generating  two independent samples
```

```
data1 = 50 + (rand(100) * 20)
data2 = 51 + (rand(100) * 20)

# Samples comparison
stat, p = mannwhitneyu(data1, data2)
print('Statistics = %.3f, p = %.3f' % (stat, p))

# Interpretation
alpha = 0.05
if p > alpha:
    print('Same distribution (fail to reject H0)')
else:
    print('Different distribution (reject H0)')
```

```
Statistics = 4412.000, p = 0.076
Same distribution (fail to reject H0)
```

11.6 Classical T Test

Table of Contents

Let's take the performance score of students in two subjects (Maths and Physics). During maths tests , the students scored [79, 100,93, 75, 84, 107, 66, 86, 103, 81, 83, 89, 105, 84, 86, 86, 112, 112, 100, 94],and during Physics test the score of [92, 100, 76, 97, 72, 79, 94, 71, 84, 76, 82, 57, 67, 78, 94,83, 85, 92, 76, 88].

It can be implemented following way :-

```
import numpy as np
from scipy import stats
# Generate the data
np.random.seed(123)
maths = np.round(np.random.randn(10)*10+90)
physics = np.round(np.random.randn(10)*10+85)
# t-test
(t, pVal) = stats.ttest_rel (maths, physics)
# Show the result
print('The probability that the two distributions '
'are equal is {0:5.3f} .'.format(pVal))
```

```
The probability that the two distributions are equal is 0.661 .
```

11.7 Statistical Modelling

Table of Contents

It can be also expressed as statistical model , we assume the difference between both the subjects as a constant value.(The null hypothesis value is equal to zero).This model has one parameter: the constant value. We can find this parameter, as well as its confidence interval and a lot of additional information, with the following Python code:

```
import pandas as pd
import statsmodels.formula.api as sm
np.random.seed(123)
df = pd.DataFrame({'Maths': maths, 'Physics':physics})
result = sm.ols(formula='I(Physics-Maths) ~ 1', data=df).fit()
print(result.summary())
```

```
                            OLS Regression Results

======================================================================
========
Dep. Variable:      I(Physics - Maths)   R-squared:
0.000
Model:                             OLS   Adj. R-squared:
0.000
Method:                  Least Squares   F-statistic:
nan
Date:                 Sat, 17 Aug 2024   Prob (F-statistic):
nan
Time:                         08:13:20   Log-Likelihood:
-43.006
No. Observations:                   10   AIC:
88.01
Df Residuals:                        9   BIC:
88.31
Df Model:                            0

Covariance Type:             nonrobust

======================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
----------------------------------------------------------------------
--------
Intercept      2.7000      5.948      0.454      0.661     -10.755
16.155
======================================================================
========
Omnibus:                         1.755   Durbin-Watson:
2.301
Prob(Omnibus):                   0.416   Jarque-Bera (JB):
0.494
Skew:                            0.544   Prob(JB):
0.781
Kurtosis:                        3.038   Cond. No.
1.00
======================================================================
========
Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.

/opt/conda/lib/python3.7/site-packages/scipy/stats/stats.py:1535:
UserWarning: kurtosistest only valid for n>=20 ... continuing anyway,
n=10
  "anyway, n=%i" % int(n))
```

11.8 One way ANOVA

Table of Contents

ANOVA (Analysis of Variance) provides a statistical test of whether two or more population means are equal. Assume we want to determine whether multiple groups differ from one another in a measurement.

Types of ANOVA Tests:

- **One-way ANOVA**: When you want to test two or more groups to see if there's a statistical difference between them. Both the t-test and one-way ANOVA test can compare the means for two groups, but only the one-way ANOVA test can compare the means of multiple groups at once,when we want to test two or more groups to see if there's a statistical difference between them. Both the t-test and one-way ANOVA test can compare the means for two groups, but only the one-way ANOVA test can compare the means of multiple groups at once.

- **Two-way ANOVA**:It is an extension of the one-way ANOVA test. A two-way ANOVA test allows us to test the effect of two independent variables at the same time. For example, if we want to compare the strength of athletes by country and by gender, we could use a two-way ANOVA test to accomplish this.

- **Three-way ANOVA**: an extension of the one-way ANOVA test and two-way ANOVA test that allows you to test the effect of three independent variables at the same time. The three-way ANOVA test is also referred to as a three-factor ANOVA test.

Calculating ANOVA

For ANOVA tests, we would set up a null and alternative hypothesis like so:

Hnull → μ1 = μ2 = μ3 = μ4

Halternative → Hnull is not true.

To calculate ANOVA, we would use the following formulas:

$$SS_b = n\sum(\bar{X} - \bar{X}_i)^2$$
$$SS_w = \sum(n_i - 1)s_i^2$$
$$SS_t = \sum(X_i j - \bar{X})^2$$
$$MS_b = \frac{SS_b}{DF_b}$$
$$MS_w = \frac{SS_w}{DF_w}$$
$$F = \frac{MS_b}{MS_w}$$

$$SS_b = n\sum(\bar{X} - \bar{X}_i)^2$$
$$SS_w = \sum(n_i - 1)s_i^2$$
$$SS_t = \sum(X_i j - \bar{X})^2$$
$$MS_b = \frac{SS_b}{DF_b}$$
$$MS_w = \frac{SS_w}{DF_w}$$
$$F = \frac{MS_b}{MS_w}$$

Degrees of Freedom for ANOVA:

DF between = k -1 DF within = N-k DF total = N-1

Key:

k = number of groups

N = total number of observations

n = number of observations for each group

Limitations of the one-way ANOVA:

A one-way ANOVA helps us to determine whether the two or more groups are difference from each other, but it never tells you which specific groups were different from one another.

```
insurance_df = pd.read_csv("/kaggle/input/insurance/insurance.csv")
insurance_df.head()
```

```
   age     sex     bmi  children smoker     region      charges
0   19  female  27.900         0    yes  southwest  16884.92400
1   18    male  33.770         1     no  southeast   1725.55230
2   28    male  33.000         3     no  southeast   4449.46200
3   33    male  22.705         0     no  northwest  21984.47061
4   32    male  28.880         0     no  northwest   3866.85520
```

```
# Test to see if the distributions of bmi values for females having
different number of children, are significantly different
import copy
Ho = "No. of children has no effect on bmi"   # Stating the Null
Hypothesis
Ha = "No. of children has an effect on bmi"   # Stating the Alternate
Hypothesis

female_df = copy.deepcopy(insurance_df[insurance_df['sex'] ==
'female'])

zero = female_df[female_df.children == 0]['bmi']
one = female_df[female_df.children == 1]['bmi']
two = female_df[female_df.children == 2]['bmi']


f_stat, p_value = stats.f_oneway(zero,one,two)


if p_value < 0.05:  # Setting our significance level at 5%
    print(f'{Ha} as the p_value ({p_value.round(3)}) < 0.05')
else:
    print(f'{Ho} as the p_value ({p_value.round(3)}) > 0.05')
```

```
No. of children has no effect on bmi as the p_value (0.716) > 0.05
```

11.9 Tukey Test

Table of Contents

One of the most commonly used post hoc tests is Tukey's Test, which allows us to make pairwise comparisons between the means of each group while controlling for the family-wise error rate

What is family-wise error rate

As we have seen earlier that during hypothesis testing there is always a type I error rate that tells us the probability of rejecting a null hypothesis that is actually true. It's the probability of getting a "false positive". While performing the hypothesis testing the type I error rate is equal to the significance level (α), which is commonly chosen to be 0.01, 0.05, or 0.10. When we conduct multiple hypothesis tests at once, the probability of getting a false positive increases.

The formula to estimate the family-wise error rate is as follows:

Family-wise error rate = 1 – (1-α)^n where:

α: The significance level for a single hypothesis test

n: The total number of tests

For example, imagine that we roll a 6-sided dice. Then the probability that the dice lands on a "1" is just 16.6%.But if we increase the no of tests then then the chances of getting the 1 increases.

```
from scipy.stats import f_oneway
from statsmodels.stats.multicomp import pairwise_tukeyhsd

Ho = "No affect of group on the values "   # Stating the Null
Hypothesis
Ha = "Group has affect on the values"   # Stating the Alternate
Hypothesis

#enter data for three groups
a = [85, 86, 88, 75, 78, 94, 98, 79, 71, 80]
b = [91, 92, 93, 90, 97, 94, 82, 88, 95, 96]
c = [79, 78, 88, 94, 92, 85, 83, 85, 82, 81]

#perform one-way ANOVA

f_stat, p_value = stats.f_oneway(a,b,c)


if p_value < 0.05:  # Setting our significance level at 5%
    print(f'{Ha} as the p_value ({p_value.round(3)}) < 0.05')
else:
    print(f'{Ho} as the p_value ({p_value.round(3)}) > 0.05')

Group has affect on the values as the p_value (0.013) < 0.05

#create DataFrame to hold data
df = pd.DataFrame({'score': [85, 86, 88, 75, 78, 94, 98, 79, 71, 80,
                             91, 92, 93, 90, 97, 94, 82, 88, 95, 96,
                             79, 78, 88, 94, 92, 85, 83, 85, 82, 81],
                   'group': np.repeat(['a', 'b', 'c'], repeats=10)})

# perform Tukey's test
tukey_test = pairwise_tukeyhsd(endog=df['score'],
                          groups=df['group'],
                          alpha=0.05)
#display results
print(tukey_test)

 Multiple Comparison of Means - Tukey HSD, FWER=0.05
=====================================================
group1 group2 meandiff p-adj   lower    upper  reject
```

```
-----------------------------------------------------
     a      b      8.4 0.0158   1.4272 15.3728   True
     a      c      1.3 0.8864  -5.6728  8.2728  False
     b      c     -7.1 0.0453 -14.0728 -0.1272   True
-----------------------------------------------------
```

11.10 Bonferroni Correction

Table of Contents

The Bonferroni correction is the simplest and most conservative approach, which sets the α value for the entire set of comparisons equal to the division of the alpha value of an individual test by the number of tests performed.

$$Bonferrini : \alpha' = \frac{\alpha}{m}$$

The α′ is the new threshold that needs to be reached for a single test to be classified as significant.

```
rand1 = stats.norm.rvs(loc=5, scale=10, size=1000, random_state=0)
rand2 = stats.norm.rvs(loc=6.5, scale=8, size=1000, random_state=0)

#Bonferroni correction function

def bonferroni_correction_function(rvs, alpha, no_test):
    alpha_bonferroni = alpha/no_test

    counter = 0
    for i in range(no_test):
        rvs_random = stats.norm.rvs(loc=5, scale=10, size=1000,
random_state=i+1)

        statistic, pvalue = stats.ttest_ind(rvs, rvs_random,
equal_var=False)

        if pvalue <= alpha_bonferroni:
            counter = counter + 1

    print(counter)

bonferroni_correction_function(rand1, alpha=0.05, no_test=100)
bonferroni_correction_function(rand2, alpha=0.05, no_test=100)
```

```
0
18
```

This shows how rand2 is the only distribution that is significant.

11.11 Holm Correction

Table of Contents

The Holm adjustment, sometimes also referred to as Holm–Bonferroni method, sequentially compares the lowest p-value with a Type I error rate that is reduced for each consecutive test. For example, if you have three groups (and thus three comparisons), this means that the first p-value is tested at the 0.05/3 level (0.017), the second at the 0.05/2 level (0.025), and third at the 0.05/1 level (0.05)

The cost of the Bonferroni method is that by protecting against false-positive errors, the risk of failing to reject one or more false null hypotheses increases. Therefore, this other method improves the method above by sorting the obtained p-values from lowest to highest and comparing them to nominal alpha levels of α/m to α

lowest p_value <α/m, 2nd_lowest p_value <α/(m-1), …, highest p_value <α

```
def bonferroni_holm_correction_function(rvs, alpha, no_test):
    pvalue_test = []
    for i in range(no_test):
        rvs_random = stats.norm.rvs(loc=5, scale=10, size=1000,
random_state=i+1)

        statistic, pvalue = stats.ttest_ind(rvs, rvs_random,
equal_var=False)
        pvalue_test.append(pvalue)

    pvalue_test_sorted = sorted(pvalue_test, key=float)

    counter = 0
    for i in range(no_test):
        if pvalue_test_sorted[i] <= alpha/(no_test-i):
            counter = counter + 1

    print(counter)

bonferroni_holm_correction_function(rand1, alpha=0.05, no_test=100)
bonferroni_holm_correction_function(rand2, alpha=0.05, no_test=100)

0
19
```

This shows how rand2 is the only distribution that is significant.

11.12 Kruskal–Wallis Test

Table of Contents

When we compare two groups to each other, we use the t-test when the data are normally distributed, and the nonparametricMann–Whitney test otherwise. For three or more groups, the test for normally distributed data is the ANOVA-test; for notnormally distributed data, the corresponding test is the Kruskal–Wallis test. When the null hypothesis is true the test statistic for the Kruskal–Wallis test follows the chi-square distribution

```
# Kruskal-Wallis H-test
from numpy.random import seed
from numpy.random import randn
from scipy.stats import kruskal
# seed the random number generator
seed(1)
# generate three independent samples
rand1 =50 + 10 * randn(100)
rand2 =50 + 10 * randn(100)
rand3 =52+ 10 * randn(100)
# compare samples
stat, p = kruskal(rand1, rand2, rand3)
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
alpha = 0.05
if p > alpha:
    print('Same distributions (fail to reject H0)')
else:
    print('Different distributions (reject H0)')
```

```
Statistics=1.202, p=0.548
Same distributions (fail to reject H0)
```

11.13 Two way ANOVA Test

Table of Contents

A two way ANOVA is similar to one way ANOVA test . In the two way ANOVA test each sample is defined in two ways, and resultingly put into two categorical groups.The two-way ANOVA therefore examines the effect of two factors on a dependent variable. A two-way ANOVA is used to estimate how the mean of a quantitative variable changes according to the levels of two categorical variables. Use a two-way ANOVA when you want to know how two independent variables, in combination, affect a dependent variable.

When to use the Two Way ANOVA Test

- When the Dependant variable is quantitative or numerical in nature
- And the two independant variables are categorical in nature. when we have collected data on a quantitative dependent variable at multiple levels of two categorical independent variables.

How does the ANOVA test work? A two-way ANOVA with interaction tests three null hypotheses at the same time:

- There is no difference in group means at any level of the first independent variable.
- There is no difference in group means at any level of the second independent variable.

- The effect of one independent variable does not depend on the effect of the other independent variable (a.k.a. no interaction effect).

Assumptions of the two-way ANOVA

- Homogeneity of variance (a.k.a. homoscedasticity) The variation around the mean for each group being compared should be similar among all groups.
- Independence of observations Our independent variables should not be dependent on one another (i.e. one should not cause the other). This is impossible to test with categorical variables – it can only be ensured by good experimental design.
- Normally-distributed dependent variable The values of the dependent variable should follow a bell curve. If your data don't meet this assumption, you can try a data transformation.

```
#create data
df = pd.DataFrame({'water': np.repeat(['daily', 'weekly'], 15),
                   'sun': np.tile(np.repeat(['low', 'med', 'high'], 
5), 2),
                   'height': [6, 6, 6, 5, 6, 5, 5, 6, 4, 5,
                              6, 6, 7, 8, 7, 3, 4, 4, 4, 5,
                              4, 4, 4, 4, 4, 5, 6, 6, 7, 8]})

#view first ten rows of data
df[:10]

   water  sun  height
0  daily  low       6
1  daily  low       6
2  daily  low       6
3  daily  low       5
4  daily  low       6
5  daily  med       5
6  daily  med       5
7  daily  med       6
8  daily  med       4
9  daily  med       5

import statsmodels.api as sm
from statsmodels.formula.api import ols

#perform two-way ANOVA
model = ols('height ~ C(water) + C(sun) + C(water):C(sun)', 
data=df).fit() # https://www.statology.org/two-way-anova-python/
sm.stats.anova_lm(model, typ=2)

                    sum_sq    df        F    PR(>F)
C(water)          8.533333   1.0  16.0000  0.000527
C(sun)           24.866667   2.0  23.3125  0.000002
C(water):C(sun)   2.466667   2.0   2.3125  0.120667
Residual         12.800000  24.0      NaN       NaN
```

Result Interpretation

Since the p-values for water and sun are both less than .05, this means that both factors have a statistically significant effect on plant height.And since the p-value for the interaction effect (.120667) is not less than .05, this tells us that there is no significant interaction effect between sunlight exposure and watering frequency.

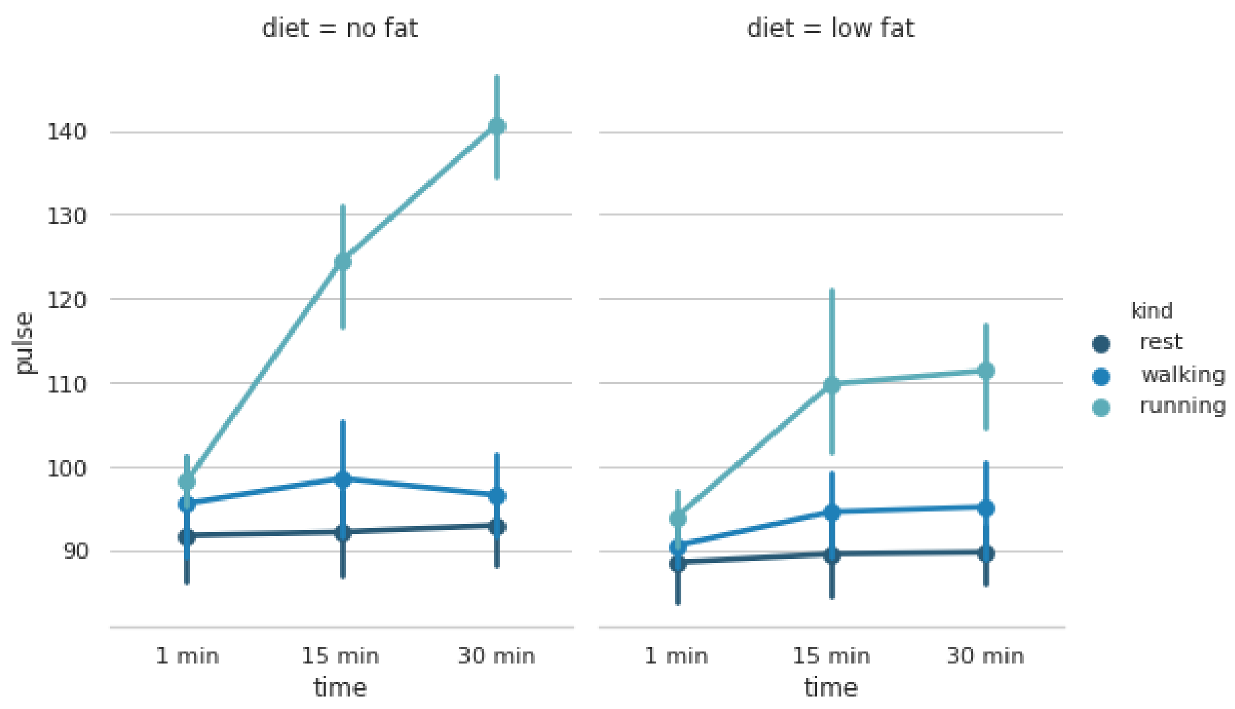
11.14 Three way ANOVA Test

Table of Contents

When we have more than two factors , there we can use the three way ANOVA test (but statistical modelling is preferred).However, as always with the analysis of statistical data, we should first inspect the data visually. seaborn makes this quite simple.For example the pulse-rate after (1/15/30) minutes of (resting/walking/running), in two groups who are eating different diets.

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="whitegrid")
df = sns.load_dataset("exercise")
sns.factorplot("time", "pulse", hue="kind", col="diet", data=df,
hue_order=["rest", "walking", "running"],
palette="YlGnBu_d", aspect=.75).despine(left=True)
plt.show()
```

```
/opt/conda/lib/python3.7/site-packages/seaborn/categorical.py:3669:
UserWarning: The `factorplot` function has been renamed to `catplot`.
The original name will be removed in a future release. Please update
your code. Note that the default `kind` in `factorplot` (`'point'`)
has changed `'strip'` in `catplot`.
  warnings.warn(msg)
```



11.15 Selecting the right test

Table of Contents

Below mentioned points is briefly describe which test to use depending on the nature of the data

- Group of nominal data

a) When the no of group is one and samples are independant :- One Sample T test or wilcoxon signed rank sum test

b) When the no of group is two or more and samples are independant :- Fisher exact test or chi-square test

c) When the no of group is two or more and samples are paired :- McNemar Test

- Group of ordinal data

a) When the no of group is two and samples are independant :- Mann-Whitney U test

b) When the no of group is two and samples are paired :- Wilcoxon signed rank test

c) When the no of group is three or more and samples are independent :- Kruskal–Wallis test

d) When the no of group is three or more and samples are paired :- Friedman test

- Group of continuous data

a) When the no of group is two and samples are independant :-Student’s t-test or Mann–Whitney test

b) When the no of group is two and samples are paired :- Paired t-test or Wilcoxon signedrank sum test

c) When the no of group is three or more and samples are independent :-ANOVA or Kruskal–Wallis test

d) When the no of group is three or more and samples are paired :- Repeated measures ANOVA or Friedman test

12. Tests on categorical data

Table of Contents

In a data sample the number of data falling into a particular group is called the frequency, so the analysis of categorical data is the analysis of frequencies. When two or more groups are compared the data are often shown in the form of a frequency table, sometimes also called contingency table.If we have only one factor (i.e., a table with only one row), the analysis options are somewhat limited. Some of the most common statistical methods used for analysis of the frequency:-

- Chi Square Test :- This statistical tests checks that the entries in the individual cells in a frequency table all come from the same distribution.it checks the null hypothesis H0 that the results are independent of the row or column in which they appear. The alternative hypothesis Ha does not specify the type of association, so close attention to the data is required to interpret the information provided by the test.

- Fisher's Exact Test :- The chi-square test is approximate, the Fisher’s Exact Test is an exact test. It is computationally more expensive and intricate than the chi-square test, and was originally used only for small sample numbers. It is more advisable to use.

-McNemar's Test :- This is a matched pair test for 2 * 2 Tables. If we want to compare the results of doctor when checking the same patients, we would use this test.

-Cochran's Q Test :- It is an extension of the MeNemar's tests for related samples that provides a method for testing for differences between three or more matched/Paired sets of frequencies or proportion

12.1 Chi Sqaare Test

Table of Contents

A Chi-Square test is a test of statistical significance for categorical variables.Chi-square test in hypothesis testing is used to test the hypothesis about the distribution of observations/frequencies in different categories.

- Assumptions of the Chi-Square Test

a) The χ2 assumes that the data for the study is obtained through random selection, i.e. they are randomly picked from the population

b) The categories are mutually exclusive i.e. each subject fits in only one category. For e.g.- from our above example – the number of people who lunched in your restaurant on Monday can't be filled in the Tuesday category

c) The data should be in the form of frequencies or counts of a particular category and not in percentages

d) The data should not consist of paired samples or groups or we can say the observations should be independent of each other

e) When more than 20% of the expected frequencies have a value of less than 5 then Chi-square cannot be used. To tackle this problem: Either one should combine the categories only if it is relevant or obtain more data

```
#Example from :- https://analyticsindiamag.com/a-beginners-guide-to-
chi-square-test-in-python-from-scratch/
np.random.seed(10)
# Sample data randomly at fixed probabilities
type_bottle = np.random.choice(a=
["paper","cans","glass","others","plastic"],
                              p = [0.05, 0.15 ,0.25, 0.05, 0.5],
                              size=1000)

# Sample data randomly at fixed probabilities
month = np.random.choice(a= ["January","February","March"],
                              p = [0.4, 0.2, 0.4],
                              size=1000)

bottles = pd.DataFrame({"types":type_bottle,
                       "months":month})

bottles_tab = pd.crosstab(bottles.types, bottles.months, margins =
True)

bottles_tab.columns = ["January","February","March","row_totals"]

bottles_tab.index =
["paper","cans","glass","others","plastic","col_totals"]

observed = bottles_tab.iloc[0:5,0:3]   # Get table without totals for
later use
bottles_tab
```

```
            January  February  March  row_totals
paper            25        65     64         154
cans             50       107     94         251
glass             8        15     15          38
others            7        21     32          60
plastic          96       189    212         497
col_totals      186       397    417        1000
```

```
#Calculating the expected values

expected =  np.outer(bottles_tab["row_totals"][0:5],
                     bottles_tab.loc["col_totals"][0:3]) / 1000

expected = pd.DataFrame(expected)

expected.columns = ["Janurary","Feburary","March"]
expected.index = ["paper","cans","glass","others","plastic"]

expected
```

```
         Janurary  Feburary    March
paper      28.644    61.138   64.218
cans       46.686    99.647  104.667
glass       7.068    15.086   15.846
others     11.160    23.820   25.020
plastic    92.442   197.309  207.249
```

```
chi_squared_stat = (((observed-expected)**2)/expected).sum().sum()
print(chi_squared_stat)
```

```
3.1891910015593856
```

```
from scipy.stats import chi2

critical_value= chi2.ppf(q = 0.95, # Find the critical value for 95%
confidence*
                      df = 8)   # df= degree of freedom

print("Critical value:",critical_value)

p_value = 1 - chi2.cdf(x=chi_squared_stat,  # Find the p-value
                             df=8)
print("P value:",p_value)
```

```
Critical value: 15.50731305586545
P value: 0.9219296414720469
```

```
import scipy
scipy.stats.chi2_contingency(observed= observed)
```

```
(7.169321280162059,
 0.518479392948842,
 8,
 array([[ 28.644,  61.138,  64.218],
        [ 46.686,  99.647, 104.667],
        [  7.068,  15.086,  15.846],
        [ 11.16 ,  23.82 ,  25.02 ],
        [ 92.442, 197.309, 207.249]]))
```

Finally, we get a p-value of 0.51847 which is greater than 0.5. Therefore, we will accept the null hypothesis that says there is no relationship between the features. The test result does not detect a significant relationship between the variables.

12.2 Fisher Exact Test

Table of Contents

Fisher's exact test is a statistical test used for testing the association between the two independent categorical variables. It is a non-parametric test and compares the proportion of categories in categorical variables.

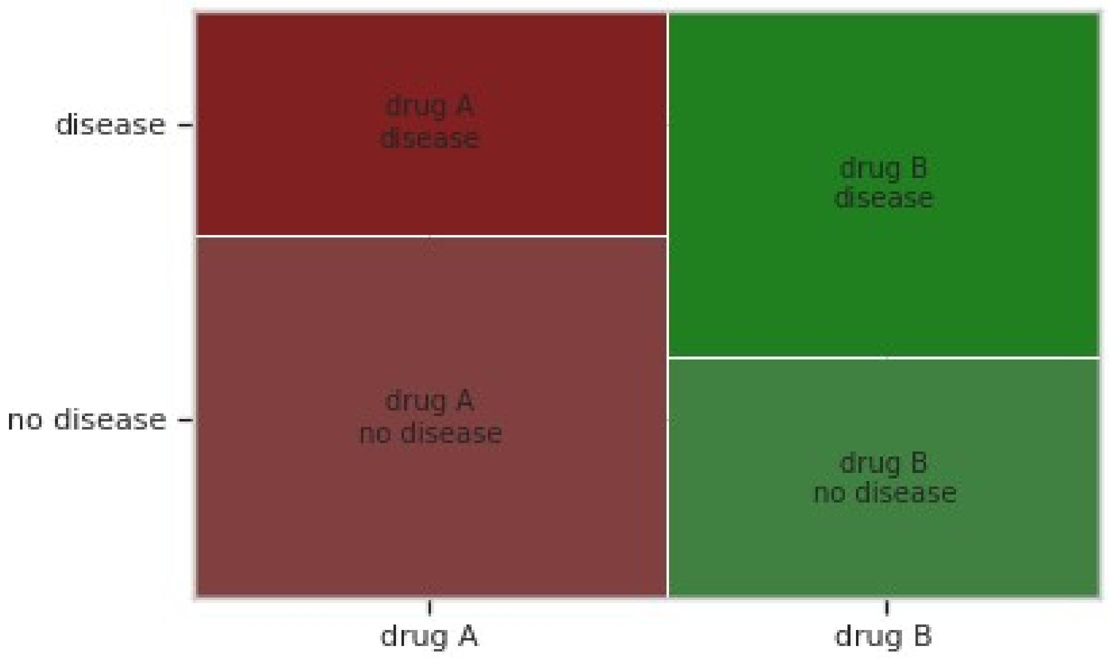
```
# Ref :- https://www.reneshbedre.com/blog/fisher-exact-test-
python.html

# create a pandas dataframe with row and column names
import pandas as pd
df = pd.DataFrame({'drug A':[80, 50], 'drug B':[48, 70]},
index=pd.Index(['no disease', 'disease']))
df

            drug A  drug B
no disease      80      48
disease         50      70

from statsmodels.graphics.mosaicplot import mosaic

df_dict = {('drug A', 'no disease'): 80, ('drug A', 'disease'): 50,
('drug B', 'no disease'): 48,
           ('drug B', 'disease'): 70}
mosaic(df_dict)
plt.show()
```



```
from scipy.stats import fisher_exact
oddsr, p = fisher_exact(table=df.to_numpy(), alternative='two-sided')
oddsr, p

(2.3333333333333335, 0.001425903669576289)
```

The p value (two-tailed) obtained from Fisher's exact test is significant [p = 0.00142, Odds ratio = 2.33] is statistically significant ($p < 0.05$) and therefore, we reject the null hypothesis. Thus, we can conclude that there is a significant association between drug treatment and disease status.

12.3 McNemar Test

Table of Contents

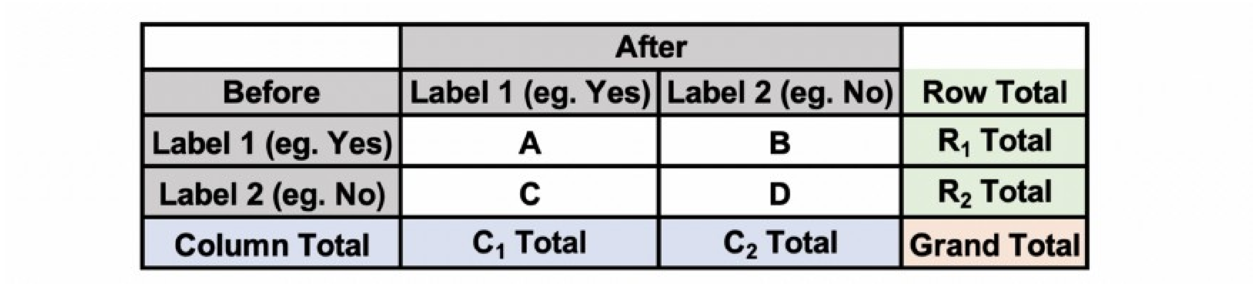
It is a statistical method for comparing proportions from two dependent populations.It is a type of paired chi-square test ($\chi 2$), but this time, both populations are dependent.Only used for paired nominal data.

1. Before-after comparison data, to uncover any changes in perception, attitude, behavior in marketing campaigns, drug testing, etc.
2. Matched pair case-control:

i. Each case has a matching control, i.e., matched on age, gender, race, etc.

ii. Twins studies, i.e., the matched pairs are twins.

Similar to the $\chi 2$ test, data need to be arranged in a 2×2 contingency table before calculating McNemar's statistic



	After		
Before	Label 1 (eg. Yes)	Label 2 (eg. No)	Row Total
Label 1 (eg. Yes)	A	B	R_1 Total
Label 2 (eg. No)	C	D	R_2 Total
Column Total	C_1 Total	C_2 Total	Grand Total

The null hypothesis (H_0) in McNemar's test is marginal homogeneity, i.e., two marginal probabilities for each outcome are the same (R_1 Total = C_1 Total and similarly R_2 Total = C_2 Total), if we expand the equation → A + B = A + C → B = C[1]

The test statistic has an approximately χ2 distribution with 1 degree of freedom:

$$\chi^2 = \frac{(|B - C| - 1)^2}{(B + C)} \sim \chi_1^2$$

```
# Reference :- https://towardsdatascience.com/mcnemars-test-with-
python-e1bab328d15c

# create sample data according to survey
data = [['Toyota', 'Toyota'] for i in range(55)] + \
       [['Toyota', 'Mitsubishi'] for i in range(5)] + \
       [['Mitsubishi', 'Toyota'] for i in range(15)] + \
       [['Mitsubishi', 'Mitsubishi'] for i in range(25)]
df = pd.DataFrame(data, columns = ['Before Ad Screening', 'After Ad
Screening'])
df

   Before Ad Screening After Ad Screening
0               Toyota             Toyota
1               Toyota             Toyota
2               Toyota             Toyota
3               Toyota             Toyota
4               Toyota             Toyota
..                 ...                ...
95          Mitsubishi         Mitsubishi
96          Mitsubishi         Mitsubishi
97          Mitsubishi         Mitsubishi
98          Mitsubishi         Mitsubishi
99          Mitsubishi         Mitsubishi

[100 rows x 2 columns]

# create contingency table
data_crosstab = pd.crosstab(df['Before Ad Screening'],
                            df['After Ad Screening'],
                            margins=True, margins_name="Total")
data_crosstab
```

```
After Ad Screening   Mitsubishi  Toyota  Total
Before Ad Screening
Mitsubishi                   25      15     40
Toyota                        5      55     60
Total                        30      70    100
```

```
# significance level
alpha = 0.01

# Calcualtion of McNemar's statistic
rows = df['Before Ad Screening'].unique()
columns = df['After Ad Screening'].unique()
mcnemar = (abs(data_crosstab['Toyota']['Mitsubishi'] -
data_crosstab['Mitsubishi']['Toyota']) - 1)**2 /
(data_crosstab['Toyota']['Mitsubishi'] + data_crosstab['Mitsubishi']
['Toyota'])

# The p-value approach
print("Approach 1: The p-value approach to hypothesis testing in the
decision rule")
p_value = 1 - stats.chi2.cdf(mcnemar, (len(rows)-1)*(len(columns)-1))
conclusion = "Failed to reject the null hypothesis."
if p_value <= alpha:
    conclusion = "Null Hypothesis is rejected."

print("McNemar's statistic is:", mcnemar, " and p value is:", p_value)
print(conclusion)
```

```
Approach 1: The p-value approach to hypothesis testing in the decision
rule
McNemar's statistic is: 4.05  and p value is: 0.044171344908442656
Failed to reject the null hypothesis.
```

```
# The critical value approach
print("\
n----------------------------------------------------------------------
------------------")
print("Approach 2: The critical value approach to hypothesis testing
in the decision rule")
critical_value = stats.chi2.ppf(1-alpha, (len(rows)-1)*(len(columns)-
1))
conclusion = "Failed to reject the null hypothesis."
if mcnemar > critical_value:
    conclusion = "Null Hypothesis is rejected."

print("McNemar's statistic is:", mcnemar, " and critical value is:",
critical_value)
print(conclusion)


--------------------------------------------------------------------------
----------------
Approach 2: The critical value approach to hypothesis testing in the
decision rule
McNemar's statistic is: 4.05  and critical value is:
6.6348966010212145
Failed to reject the null hypothesis.

# McNemar’s Test on matched pair case-control data

# create sample data according to survey
data = [['Yes', 'Yes'] for i in range(71)] + \
       [['Yes', 'No'] for i in range(50)] + \
       [['No', 'Yes'] for i in range(24)] + \
       [['No', 'No'] for i in range(55)]
df = pd.DataFrame(data, columns = ['Treatment B', 'Treatment A'])
df.columns = pd.MultiIndex.from_product([['any_emergency_room_visit'],
df.columns])

# create contingency table
data_crosstab = pd.crosstab(df['any_emergency_room_visit']['Treatment
B'],
                            df['any_emergency_room_visit']['Treatment
A'],
                            margins=True, margins_name="Total")
data_crosstab.columns =
pd.MultiIndex.from_product([['any_emergency_room_visit'],
data_crosstab.columns])

# significance level
alpha = 0.01

# Calcualtion of McNemar's statistic
rows = df['any_emergency_room_visit']['Treatment B'].unique()
columns = df['any_emergency_room_visit']['Treatment A'].unique()
mcnemar = (abs(data_crosstab['any_emergency_room_visit']['Yes']['No']
- data_crosstab['any_emergency_room_visit']['No']['Yes']) - 1)**2 /
(data_crosstab['any_emergency_room_visit']['Yes']['No'] +
data_crosstab['any_emergency_room_visit']['No']['Yes'])

# The p-value approach
print("Approach 1: The p-value approach to hypothesis testing in the
decision rule")
p_value = 1 - stats.chi2.cdf(mcnemar, (len(rows)-1)*(len(columns)-1))
conclusion = "Failed to reject the null hypothesis."
if p_value <= alpha:
    conclusion = "Null Hypothesis is rejected."

print("McNemar's statistic is:", mcnemar, " and p value is:", p_value)
print(conclusion)

# The critical value approach
print("\
n------------------------------------------------------------------------
-----------------")
print("Approach 2: The critical value approach to hypothesis testing
in the decision rule")
critical_value = stats.chi2.ppf(1-alpha, (len(rows)-1)*(len(columns)-
1))
conclusion = "Failed to reject the null hypothesis."
if mcnemar > critical_value:
    conclusion = "Null Hypothesis is rejected."

print("McNemar's statistic is:", mcnemar, " and critical value is:",
critical_value)
print(conclusion)
```

```
Approach 1: The p-value approach to hypothesis testing in the decision
rule
McNemar's statistic is: 8.445945945945946  and p value is:
0.003658580873555639
Null Hypothesis is rejected.

------------------------------------------------------------------------
----------------
Approach 2: The critical value approach to hypothesis testing in the
decision rule
McNemar's statistic is: 8.445945945945946  and critical value is:
6.6348966010212145
Null Hypothesis is rejected.
```

12.4 Cochran Q Test

Table of Contents

Cochran’s Q test is a statistical test that is used to determine whether the proportion of “successes” is equal across three or more groups in which the same individuals appear in each group.For example, we may use Cochran’s Q test to determine if the proportion of students who pass a test is equal when using three different studying techniques.

```
from mlxtend.evaluate import cochrans_q
from mlxtend.evaluate import mcnemar_table
from mlxtend.evaluate import mcnemar
```

```
#Ref:- http://rasbt.github.io/mlxtend/user_guide/evaluate/cochrans_q/
## Dataset:

# ground truth labels of the test dataset:

y_true = np.array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0,
                   0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0,
                   0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0,
                   0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0,
                   0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0,
                   0, 0, 0, 0, 0])


# predictions by 3 classifiers (`y_model_1`, `y_model_2`, and
`y_model_3`):

y_model_1 = np.array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0,
                      0, 0])

y_model_2 = np.array([1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                      1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0,
                      0, 0])

y_model_3 = np.array([1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                      1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0,
                              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0,
                              1, 1])

q, p_value = cochrans_q(y_true,
                        y_model_1,
                        y_model_2,
                        y_model_3)

print('Q: %.3f' % q)
print('p-value: %.3f' % p_value)

Q: 7.529
p-value: 0.023
```

Since the p-value is smaller than α, we can reject the null hypothesis and conclude that there is a difference between the classification accuracies.Lastly, let's illustrate that Cochran's Q test is indeed just a generalized version of McNemar's test:

```
chi2, p_value = cochrans_q(y_true,
                           y_model_1,
                           y_model_2)

print('Cochran\'s Q Chi^2: %.3f' % chi2)
print('Cochran\'s Q p-value: %.3f' % p_value)

Cochran's Q Chi^2: 5.333
Cochran's Q p-value: 0.021

chi2, p_value = mcnemar(mcnemar_table(y_true,
                                      y_model_1,
                                      y_model_2),
                        corrected=False)

print('McNemar\'s Chi^2: %.3f' % chi2)
print('McNemar\'s p-value: %.3f' % p_value)

McNemar's Chi^2: 5.333
McNemar's p-value: 0.021
```