

DATA & SCIENCE MACHINE LEARNING ALGORITHMS

A CONCISE GUIDE

(including R & Python functions, Datasets, and Free Text Books)

ANANTHU
CHAKRAVARTHI

To
my
parents,
&
children

A. S. Chakravarthi

*M.S.Eng. (Princeton, *94)*

Contents

Introduction to Data Science

What is Data Science?

Steps in Data Science Process

Data Science Analytics

Applications of Data Science

What is required to be a good Data Scientist?

How is this material organized?

Statistics

What is Statistics?

Measures of Central Tendency

Mean

Median

Mode

Measures of Dispersion (or Variability)

Range

Interquartile Range (IQR)

Variance or Standard Deviation

Sum of Squared Deviation

Coefficient of Variation

Measures of Shape

Skewness

Kurtosis

Statistical Distributions

What is a Random Variable?

Binomial Distribution

[Normal Distribution](#)

[Central Limit Theorem](#)

[Poisson Distribution](#)

[Chi-squared Distribution](#)

[Student's t-distribution](#)

[Machine Learning Algorithms](#)

[Classification of Variables](#)

[Functions](#)

[Classification of Data](#)

[Levels of Measurement](#)

[Machine Learning](#)

[Supervised Learning](#)

[Unsupervised Learning](#)

[Types of Algorithms](#)

[Regression Algorithms](#)

[Classification Algorithms](#)

[Clustering Algorithms](#)

[Programming Languages for Machine Learning Algorithms](#)

[R](#)

[Python](#)

[Regression Algorithms](#)

[Simple Linear Regression](#)

[Multiple Linear Regression](#)

[Simple Polynomial Regression](#)

[Multiple Polynomial Regression](#)

[Nonlinear Regression](#)

[Logistic Regression](#)

[Classification Algorithms](#)

[Naive Bayes Classifier](#)

[K-Nearest Neighbors \(KNN\)](#)

[Support Vector Machines \(SVM\)](#)

[Decision Trees](#)

[Random Forests](#)

[Artificial Neural Networks \(ANN\)](#)

[Clustering Algorithms](#)

[K-Means Clustering](#)

[Hierarchical Clustering](#)

[RESOURCES](#)

[E-BOOKS \(FREE\)](#)

[ONLINE, OPEN-ACCESS TEXT BOOKS](#)

[KNOWLEDGE RESOURCES \(Indispensable for a Data Scientist\)](#)

[REFERENCES](#)

[DATASETS](#)

Introduction to Data Science

What is Data Science?

Data Science is an inter-disciplinary field to extract useful insights or knowledge regarding data through collection, management, analysis, visualization and reporting.

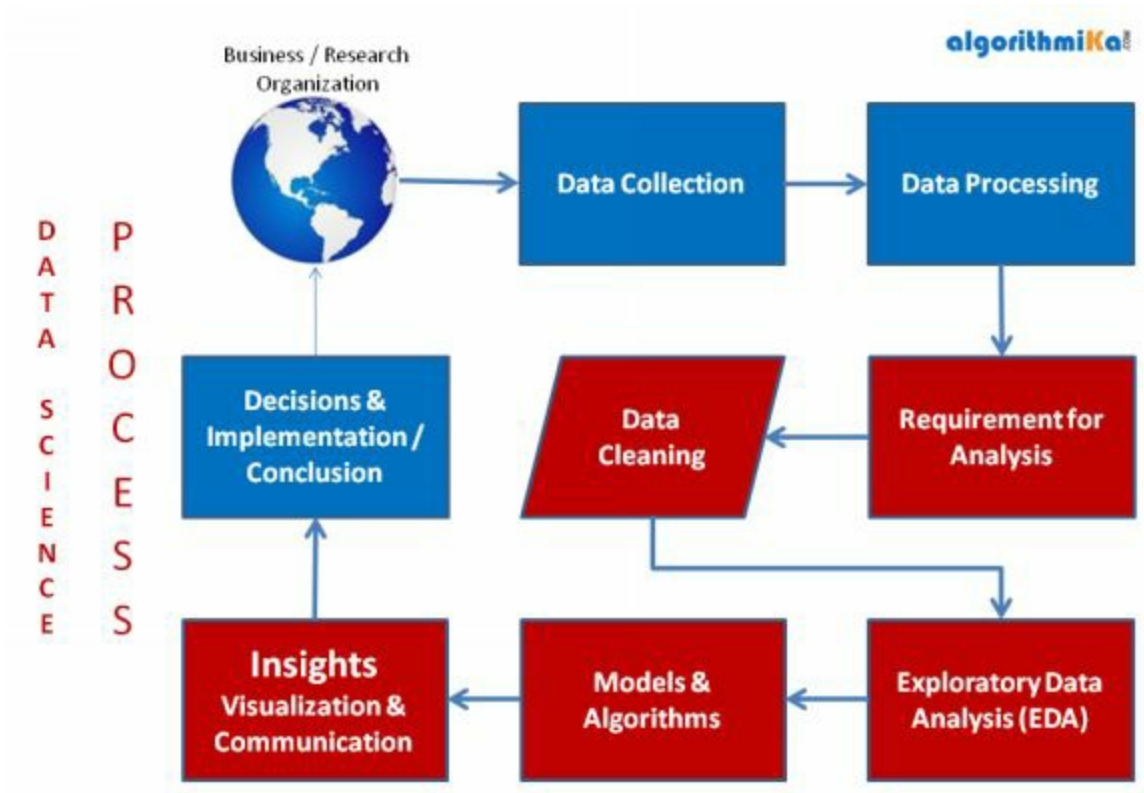
Data Science is relatively new branch of computational sciences, which is gaining importance with the explosion of the data and the advent of web-commerce which needs the analysis of customer behavior.

Data Scientist is a person who has the knowledge and skills to conduct systematic and sophisticated analysis of data to extract useful insights. Data Scientist is defined as a person who is better at statistics than a programmer and better at programming than a statistician.

Steps in Data Science Process

The following steps need to be followed for the disciplined Data Science process:

1. Understanding the Business Problem
2. Cleaning of Data
3. Exploratory Data Analysis
 1. Descriptive Analytics
 2. Visualization (Graphs)
 3. Trend Analysis
4. Insights Report



Data Science Analytics

Data Science Analytics are classified as:

1. Descriptive Analytics: This is the part of data analytics which use data aggregation to provide insight into the past and answers: “What has happened?”
2. Predictive Analytics: This is the part of data analytics which uses statistical models and forecasting techniques to estimate the future and answer: “What could happen?”
3. Prescriptive Analytics: This is the part of data analytics which uses simulation and optimization algorithms to advice on possible outcomes and answer: “What should we do?”

Applications of Data Science

Data Science has the applications in the following Domains, including:

- Hitech: Self-driving cars, Artificial Intelligence, Spam filtering, Voice recognition, Character recognition, No-Q Checkouts
- Finance: Credit Risk Modelling, Credit Scoring, Financial Analysis, Wealth Management, Portfolio Risk Management, Currency & Stock Trading
- Retail & Marketing: Customer Behavior Analysis, Market Basket Analysis
- Aerospace: Avionics
- Pharma: Clinical Trials, Molecular Modeling
- Life Sciences: Genomics
- Insurance: Actuary
- Human Resources: Salary Estimation, Employee Attraction & Attrition

What is required to be a good Data Scientist?

Data Science analytics process needs to have the knowledge of the following:

- EXCEL
- SQL
- R and/or Python
- Statistics & Probability
- Visualization tools: TABLEAU / QlikView ...
- Big Data: Hadoop, Spark, Scala
- an attitude to learn and enjoy the new scientific developments

How is this material organized?

The author has attempted to provide concise information with as less Mathematics as possible in this booklet. More detailed theory and the Mathematical Analyses will be added in the text book form by the same author, which will be published in the future.

The author has given crucial information of the Package, Function, usage, etc. of R and Python to be used in the respective programs. In the book form, detailed analysis of the datasets using both R and Python in the Disciplined Data Science Process will be included. Program code and visuals of data and analysis will also be included.

Please refer to www.algorithmiKa.com, for the updates of this booklet.

Statistics

What is Statistics?

Statistics is the science of collection, organization, analysis, interpretation, and presentation of data, from the whole data or from a representative sample.

Statistics has three types of measures:

1. Measures of Central Tendency
2. Measures of Dispersion (or Variation)
3. Measures of Skewness

Measures of Central Tendency

Measures of Central Tendency are the numerical descriptive measures which indicate the center of the distribution.

There are three common measures of central tendency:

1. Mean
2. Median
3. Mode

Mean

This is commonly represented by the arithmetic mean which is the arithmetic average of the probability distribution. Arithmetic Mean is commonly denoted by μ or \bar{x} .

$$\begin{aligned} \text{Arithmetic Mean or Sample Mean, } \bar{x} &= \frac{\sum_{i=1}^n x_i}{n} = \frac{x_1 + x_2 + \dots + x_n}{n} \quad \text{for equal weights} \end{aligned}$$

$$\text{Weighted Arithmetic Mean, } \bar{x} = \frac{\sum_{i=1}^n x_i w_i}{\sum_{i=1}^n w_i} = \frac{x_1 w_1 + x_2 w_2 + \dots + x_n w_n}{w_1 + w_2 + \dots + w_n} \quad \text{for weights } w_i.$$

For discrete probability distribution, if each possible value x_i has its probability $P(x_i)$:

$$\text{Distribution Mean, } \mu = \sum_{i=1}^n x_i p(x_i)$$

Example:

If the elements of the sample are:

1, 2, 3, 3, 3, 4, 4, 5, 5, 6

$$\text{Arithmetic Mean, } \mu \text{ or } \bar{x} = \frac{1 + 2 + 3 + 3 + 3 + 4 + 4 + 5 + 5 + 6}{10} = \frac{36}{10} = 3.6$$

Geometric Mean (GM)

The Geometric Mean is an average represented by the product of the elements (e.g., Rate of Growth).

$$\text{Geometric Mean, } \bar{x} = \left(\sum_{i=1}^n x_i \right)^{1/n}$$

Example:

Geometric Mean of the above sample,

$$\bar{x} = (1.2.3.3.3.4.4.5.5.6)^{1/10} = \sqrt[10]{259,200} = 3.47$$

Harmonic Mean (HM)

The Harmonic Mean is an average represented by the inverse of the elements (e.g., distance per unit of time)

$$\text{Harmonic Mean, } \bar{x} = n \cdot \left(\sum_{i=1}^n \frac{1}{x_i} \right)^{-1}$$

Example:

Harmonic Mean of the above sample,

$$\bar{x} = 10 \cdot \frac{1}{\left(\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{3} + \frac{1}{3} + \frac{1}{4} + \frac{1}{4} + \frac{1}{5} + \frac{1}{5} + \frac{1}{6} \right)} = 10 \cdot \frac{1}{\left(\frac{214}{60} \right)} = 2.8$$

Properties of Mean

1. Mean can be calculated on any set of numerical data.
2. A set of numerical data has only one mean.
3. Since it takes into account all the items of the numerical data, it is most reliable measure of Central Tendency.
4. It is affected by the outliers (extreme or deviant values).

Median

The median is the value separating the higher half of a data sample, a population, or a probability distribution, from the lower half.

In the above example of the elements of the sample:

1, 2, 3, 3, 3, 4, 4, 5, 5, 6

Median = Average of 5th & 6th elements = $(3 + 4) / 2 = 3.5$

Mode

The mode is the value that appears most often in a set of data.

In the above example of the elements of the sample: 1, 2, 3, 3, 3, 4, 4, 5, 5, 6

Mode = Value appeared most often = 3

Measures of Dispersion (or Variability)

In statistics, dispersion (also called variability, scatter, or spread) denotes how stretched or squeezed a distribution (theoretical or that underlying a statistical sample) is.

Common examples of measures of statistical dispersion are:

1. Range
2. Inter-quartile range (IQR)
3. Variance or Standard Deviation
4. Sum of Squares (SS)
5. Coefficient of Variation

Range

Range is the difference between the highest value and the lowest value.

In the above example of the elements of the sample: 1, 2, 3, 3, 3, 4, 4, 5, 5, 6

$$\text{Range} = 6 - 1 = 5$$

Interquartile Range (IQR)

Interquartile Range is the difference between the upper and lower quartiles (75th and 25th percentiles). It is also called the midspread, H-spread, or middle 50%.

Variance or Standard Deviation

Variance is the mean square deviation from the mean of the distribution.

$$\text{Population Variance, } \sigma^2 = \frac{\sum_{i=1}^N (x_i - \mu)^2}{N}$$

$$\text{Sample Variance, } s^2 = \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n-1}$$

Standard Deviation is the Root Mean Square deviation from the mean of the distribution. Hence, it is the Square Root of the variance, and denoted by:

1. σ for Population Standard Deviation and
2. s for Sample Standard Deviation.

Sum of Squared Deviation

Sum of Squares is the square deviation from the mean of the distribution.

$$\text{Sum of Squares, } SS(x) = \sum_{i=1}^n (x_i - \bar{x})^2$$

Coefficient of Variation

Coefficient of Variation is defined as the ratio of Standard Deviation σ to the mean μ . It shows the extent of variability in relation to the mean of the population.

$$\text{Coefficient of Variation, } C_v = \frac{\sigma}{\mu}$$

Measures of Shape

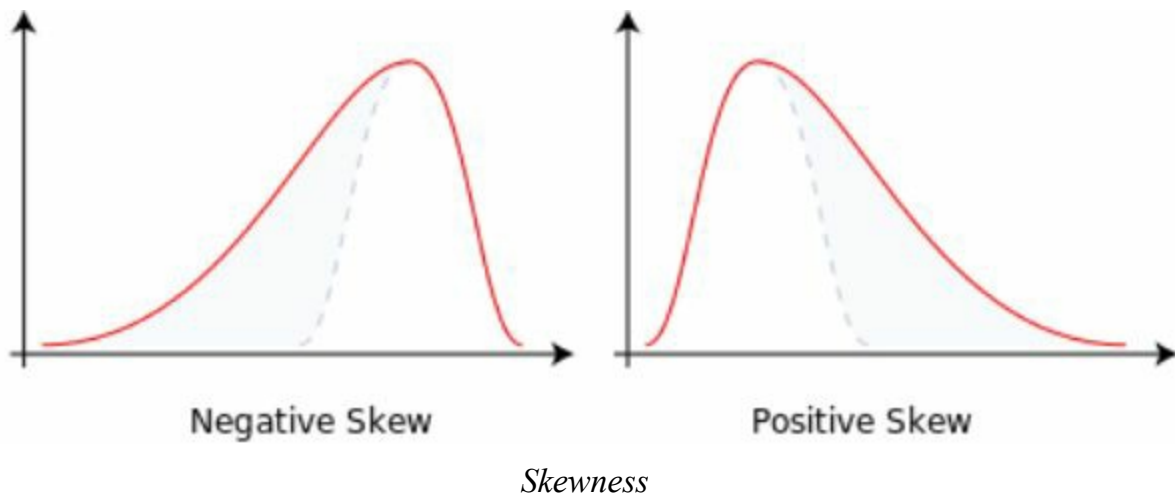
Measures of shape in a statistical distribution is measured by:

1. Skewness
2. Kurtosis

Skewness

Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean. The skewness value can be positive or negative, or even undefined.

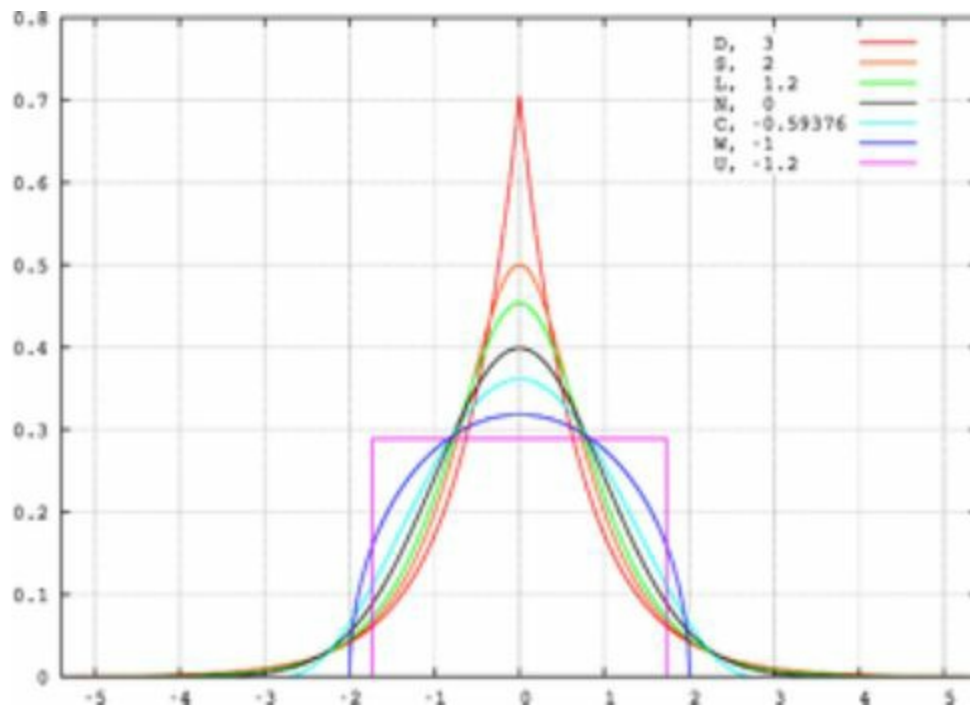
$$\text{Skewness, } \sigma_1 = E\left[\left(\frac{X - \mu}{\sigma}\right)^3\right]$$



Kurtosis

Kurtosis is a measure of the "tailedness" of a distribution.

$$\text{Kurtosis, } \text{Kurt}(x) = \frac{\mu_4}{\sigma^4} = \frac{E[(x - \mu)^4]}{(E[(x - \mu)^2])^2}$$



Kurtosis

Statistical Distributions

What is a Random Variable?

A Random Variable is a variable whose values are numerical outcomes of a random phenomenon.

A Random Variable is of two types:

1. Discrete Random Variable: Random variable that takes only a finite number of distinct values. e.g., Dice, Number of Students, Children in a family, etc.
2. Continuous Random Variable: Random variable that takes infinite number of possible values. e.g., Height, Weight, Length, etc.

Binomial Distribution

Binomial distribution is the discrete probability distribution of the number of successes in a sequence of n independent yes/no experiments, each of which yields success with probability p .

The Probability Mass Function of the probability of getting exactly k successes in n trials is given by:

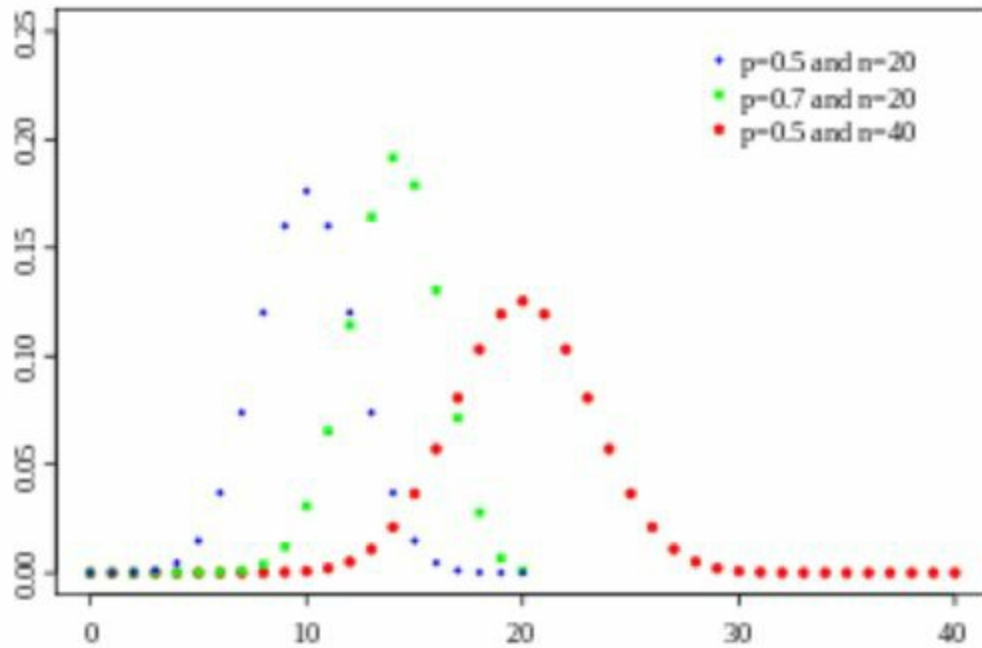
$$f(k, n, p) = {}^n C_k p^k (1 - p)^{n - k}, \quad \text{where } {}^n C_k = \frac{n!}{k!(n - k)!}$$

The Cumulative Distribution Function can be expressed as:

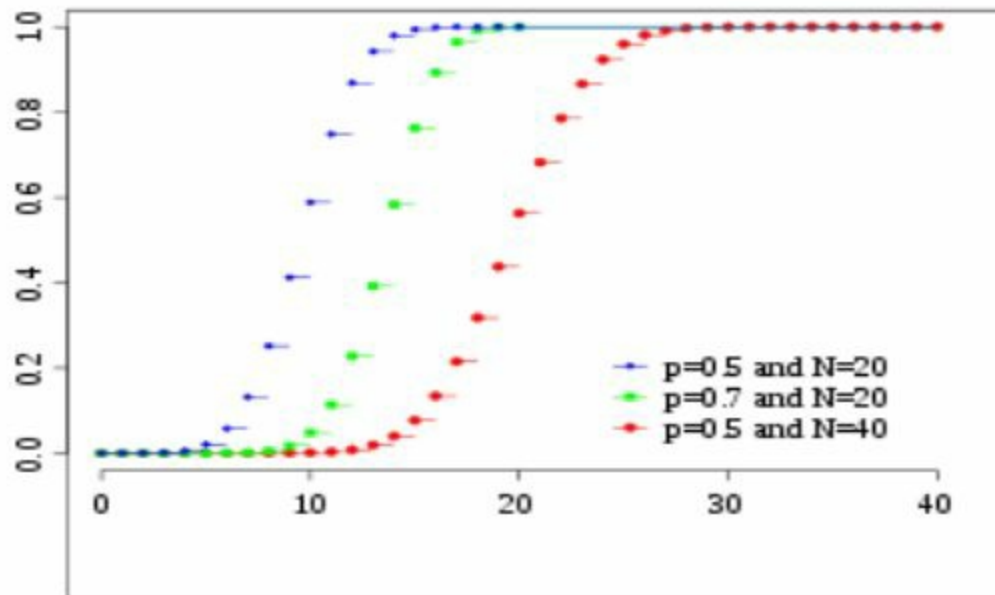
$$F(k, n, p) = \Pr(X \leq k) = \sum_{i=0}^k {}^n C_i p^i (1 - p)^{n - i}$$

The mean of the function is: $\mu = E(X) = np$

The Variance of the function is: $\sigma^2 = \text{Var}(X) = np(1 - p) = npq$



Probability Density Function for Binomial Distribution



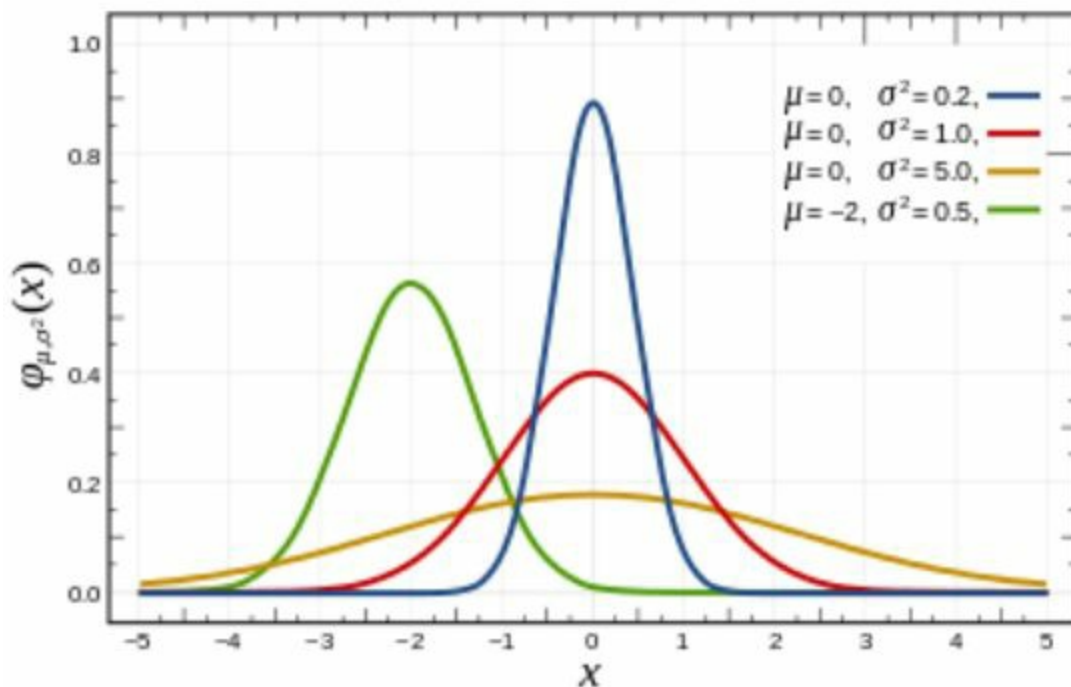
Cumulative Distribution Function for Binomial Distribution

Normal Distribution

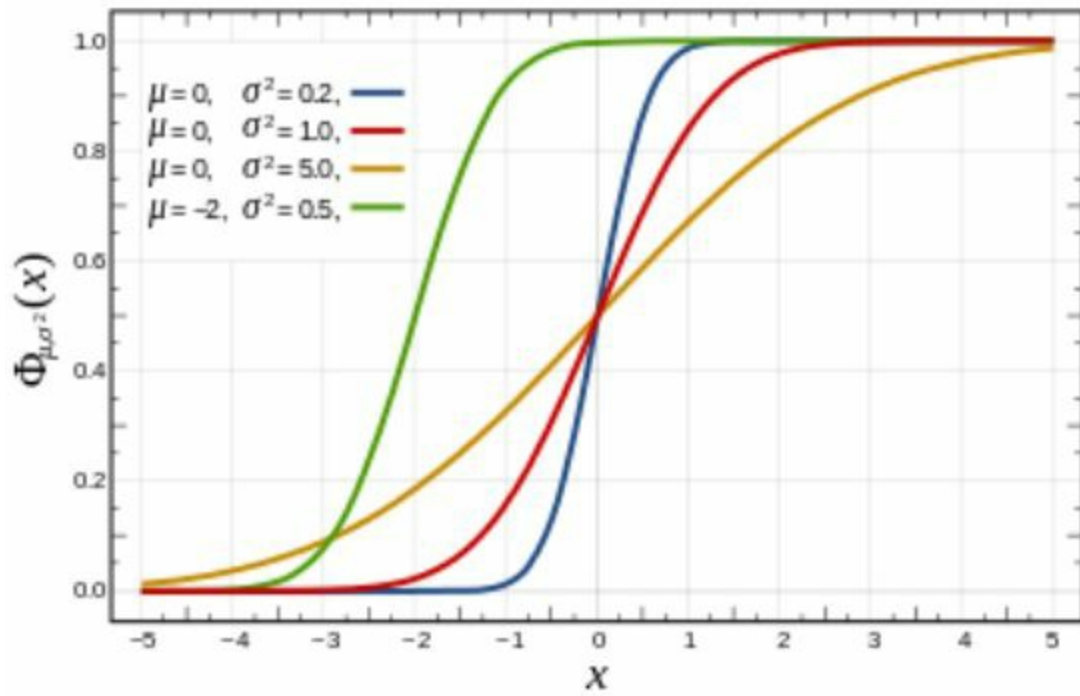
The probability density function of the normal distribution is:

$$f(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x - \mu)^2}{2\sigma^2}}$$

where μ is the mean and σ is the standard deviation.



Probability Density Function for Normal Distribution



Cumulative Distribution Function for Normal Distribution

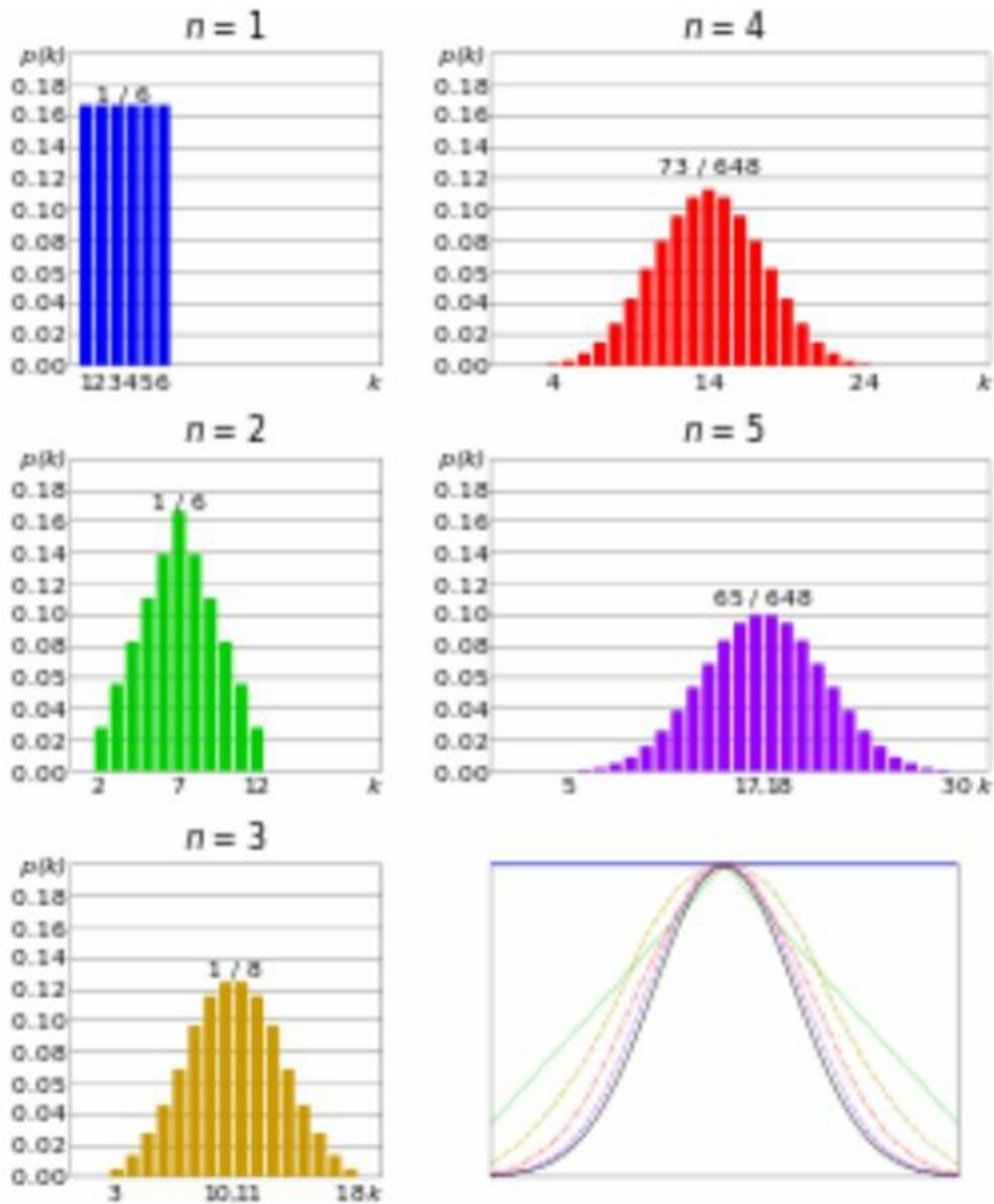
** Red curve is the standard normal distribution*

Central Limit Theorem

The central limit theorem states that under certain (fairly common) conditions, the sum of many random variables will have an approximately normal distribution.

The central limit theorem also implies that certain distributions can be approximated by the normal distribution, for example:

- The binomial distribution $B(n, p)$ is approximately normal with mean np and variance $np(1-p)$ for large n and for p not too close to zero or one.
- The Poisson distribution with parameter λ is approximately normal with mean λ and variance λ , for large values of λ .
- The chi-squared distribution $\chi^2(k)$ is approximately normal with mean k and variance $2k$, for large k .
- The Student's t -distribution $t(v)$ is approximately normal with mean 0 and variance 1 when v is large.



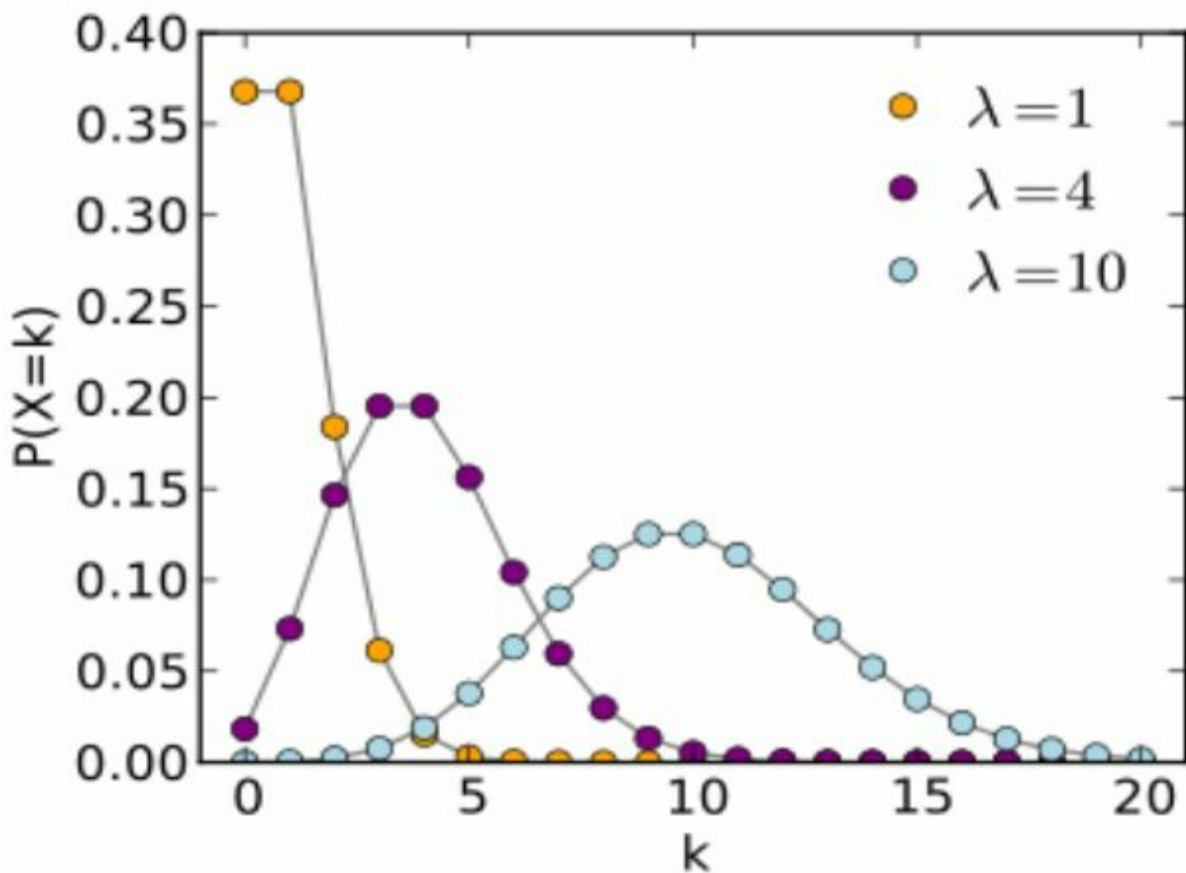
Comparison of probability density functions, $p(k)$ for the sum of n fair 6-sided dice to show their convergence to a normal distribution with increasing n , in accordance to the central limit theorem. In the bottom-right graph, smoothed profiles of the previous graphs are rescaled, superimposed and compared with a normal distribution (black curve).

Poisson Distribution

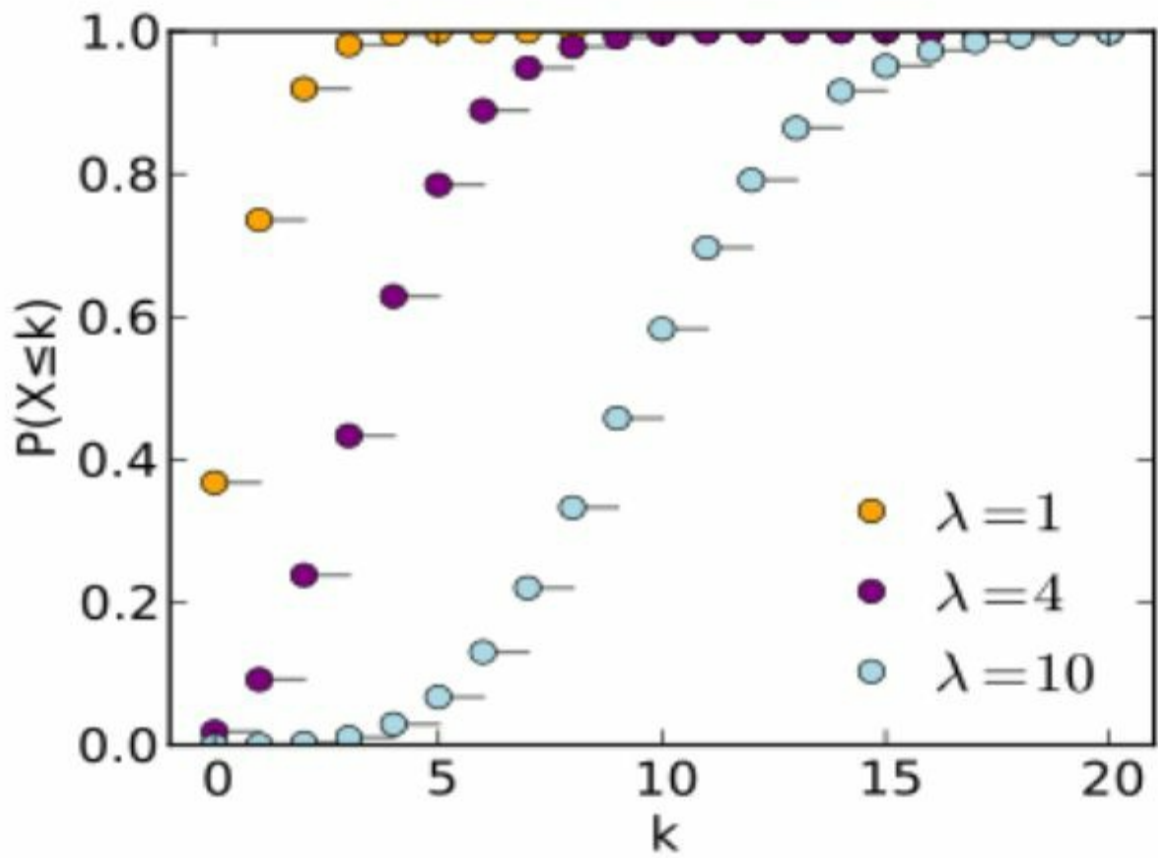
The Poisson distribution is a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed interval of time and/or space if these events occur with a known average rate and independently of the time since the last event.

Probability density function for Poisson Distribution is:

$$P(k) = \frac{\lambda^k e^{-\lambda}}{k!}$$



Probability Density Function for Poisson Distribution



Cumulative Distribution Function for Poisson Distribution

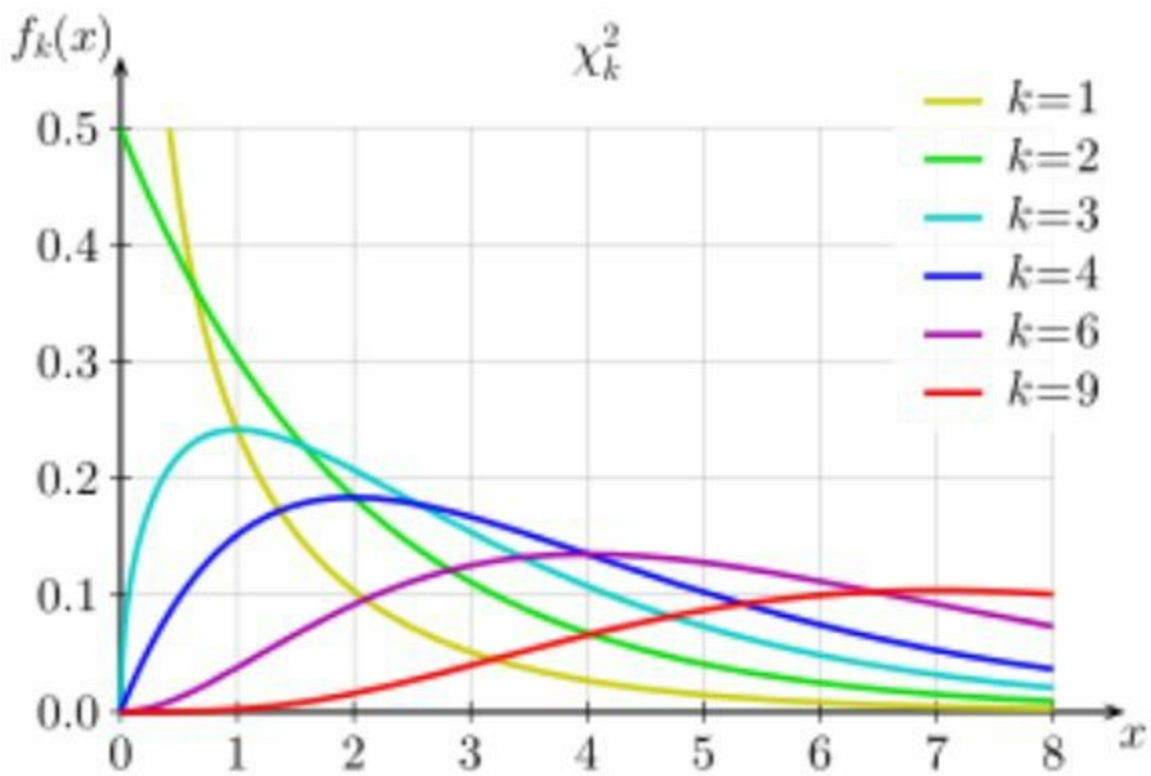
Chi-squared Distribution

The chi-squared distribution (also chi-square or χ^2 -distribution) with k degrees of freedom is the distribution of a sum of the squares of k independent standard normal random variables. It is one of the most widely used probability distributions in inferential statistics, e.g., in hypothesis testing or in construction of confidence intervals.

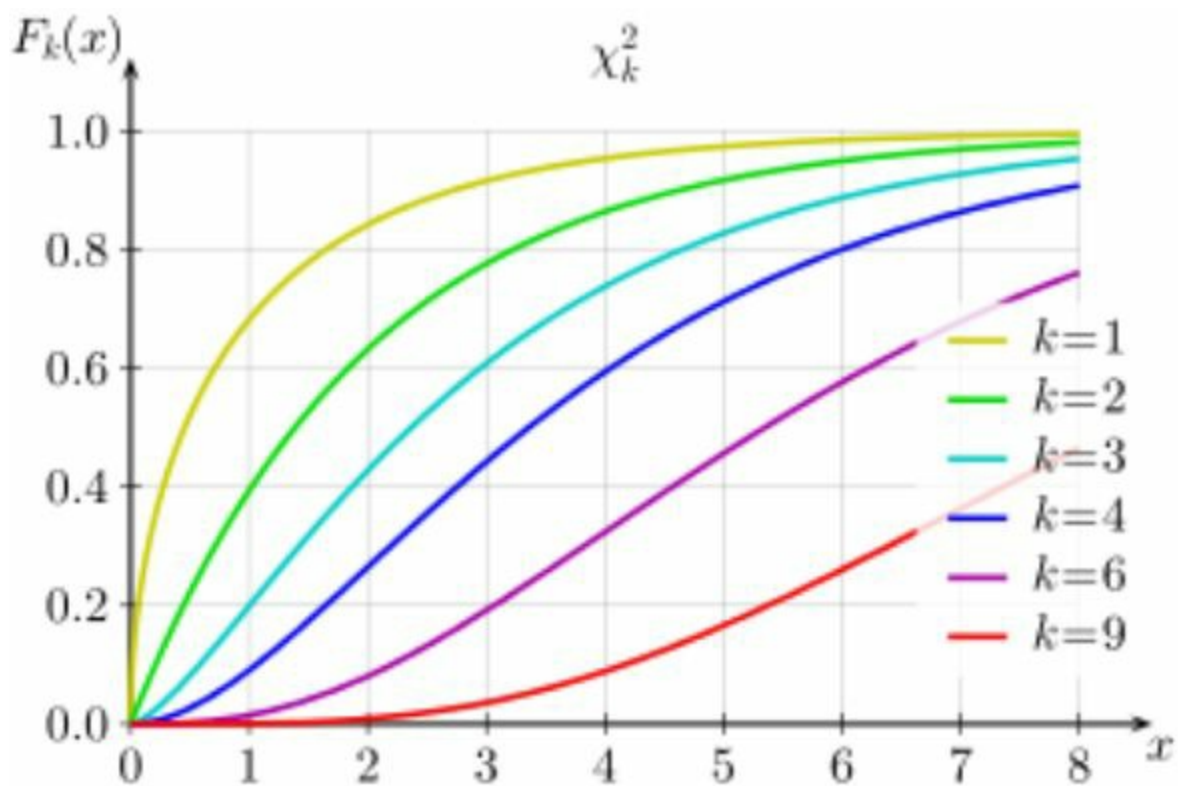
The Probability density function for Chi-squared Distribution is:

$$f(x, k) = \frac{x^{\frac{k}{2} - 1} e^{-x/2}}{2^{k/2} \Gamma\left(\frac{k}{2}\right)}, \quad x > 0;$$
$$= 0, \quad \text{Otherwise}$$

$\Gamma(k/2)$ denotes the Gamma function, which has closed-form values for integer k .



Probability Density Function for Chi-squared Distribution



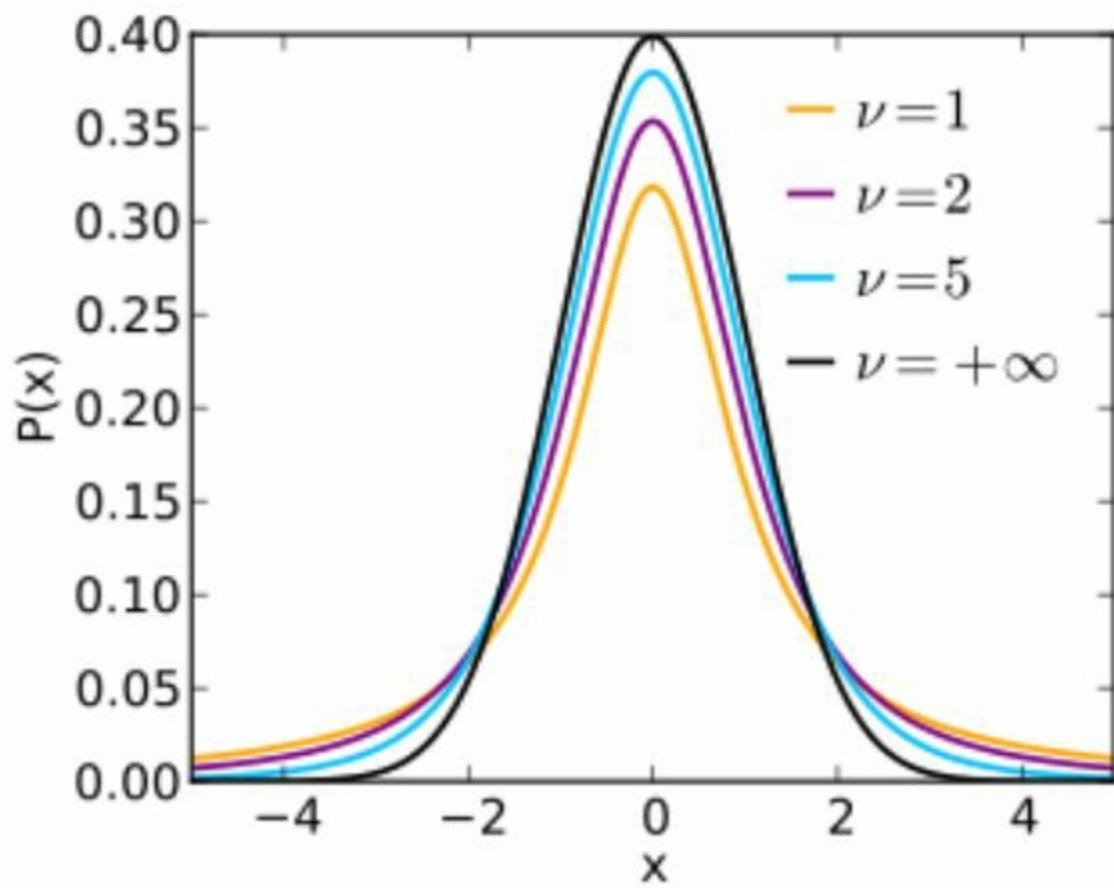
Cumulative Distribution Function for Chi-squared Distribution

Student's t-distribution

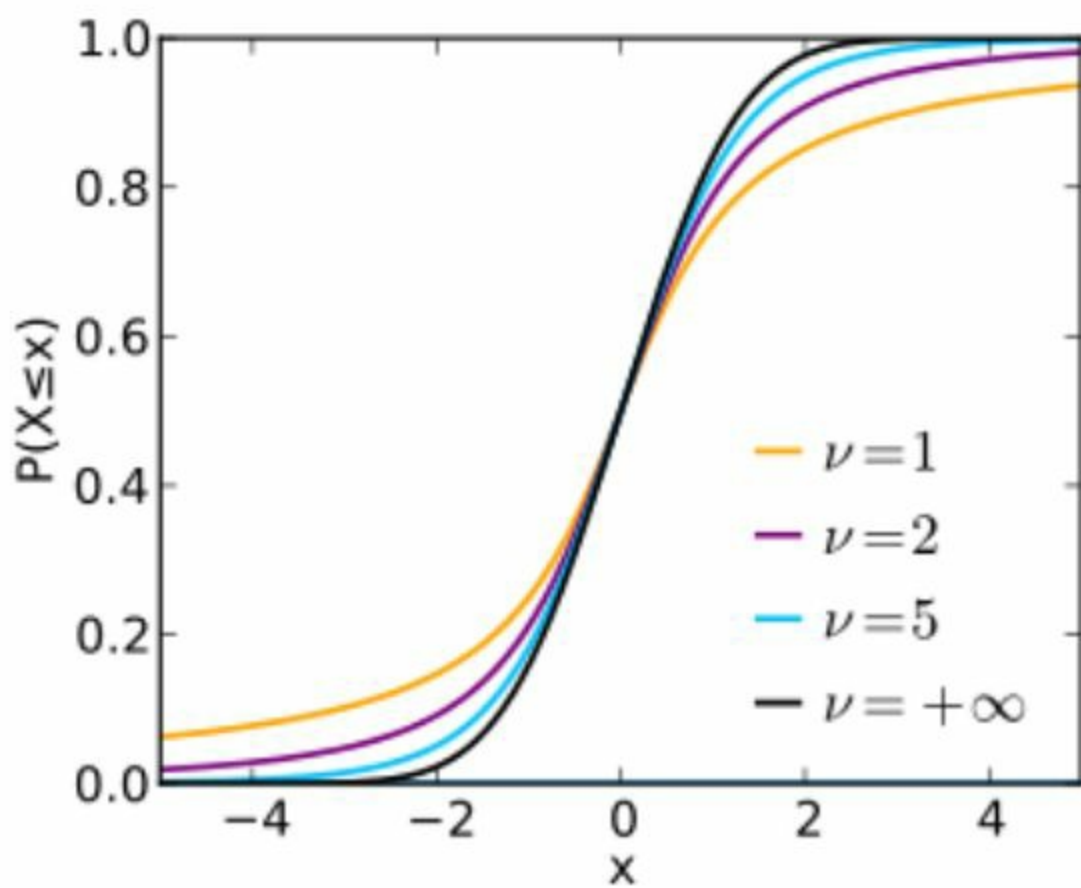
Student's t-distribution (or simply the t-distribution) is any member of a family of continuous probability distributions that arises when estimating the mean of a normally distributed population in situations where the sample size is small and population standard deviation is unknown.

Student's t-distribution has the probability density function given by:

$$f(t) = \frac{\Gamma\left(\frac{v+1}{2}\right)}{\sqrt{v\pi} \Gamma\left(\frac{v}{2}\right)} \left(1 + \frac{t^2}{v}\right)^{-\frac{v+1}{2}}$$



Probability Density Function for Student's t -distribution



Cumulative Distribution Function for Student's t -distribution

Machine Learning Algorithms

Classification of Variables

In Mathematical or Statistical modelling, there are two types of variables:

1. **Dependent variable:** The dependent variable represents the outcome whose variation is being studied. It is also called as:
 - Response variable
 - Predicted variable
 - Explained variable
 - Derived variable
 - Label

A dependent variable is usually denoted by y .

2. **Independent variables:** The independent variables represent the inputs or causes for the variation in the dependent variable. They are also called as:
 - Exploratory variables
 - Predictors
 - Features

An independent variable is usually denoted by x .

Functions

A function is a rule which takes the values of independent variables and provides the value of dependent variable.

The relationship between a dependent variable and an independent variable in a function is denoted by:

$$y = f(x) \quad \text{e.g., } y = mx + c \quad \text{or } y = ax^2 + bx + c$$

Univariate Function: If the function takes only one independent variable, it is called a *univariate* function. The above example represents a univariate function.

Linear Function: If the dependent variable is linearly correlated (means, with power of 1) with the independent variable, it is called *linear function*.

$$y = m x + c$$

Non-linear Function: If the dependant variable is non-linearly correlated (means, with the power of not equal to 1 or with any other mathematical transformations) with the independent variable, it is called *non-linear function*.

$$y = a x^2 + b x + c \quad \text{or } y = \log(x) \quad \text{or } y = e^x$$

Quadratic Function: The function in which the maximum power of the independent variable is 2 is called a quadratic function.

$$f(x) = a x^2 + b x + c$$

Multivariate Function: If the function takes More than one independent variable, it is called a *multivariate function*.

$$y = f(x_1, x_2, \dots, x_n) = a_1 x_1 + a_2 x_1^2 + b_1 x_2 + \dots + z x_n$$

Multiple Linear Function: If the dependent variable is linearly correlated (means, with power of 1) with multiple independent variables, it is called *multiple linear function*.

$$y = a x_1 + b x_2 + \dots + z x_n$$

Classification of Data

Data is classified into two types:

1. Continuous data: Continuous data can take any value in the defined range. Between any two continuous data points, there will be an infinite other possible data points. Continuous data is always numeric in nature.
e.g., Height, Weight, Length
2. Discrete data: Discrete data can only take particular values. Potential values it can take can be finite or infinite. Discrete data can be:
 - Numeric: Population, Number of Students, etc.
 - Categorical: Yes-No, Male-Female, Red-Blue-Green, etc.

Levels of Measurement

The level of measurement refers to the relationship among the values that are assigned to the attributes of a variable. It helps in deciding how to interpret the data from that variable.

Variable	Quality of Service		
Attributes	Bad	Neutral	Good
Values	1	2	3
Relationship	----->		

There are four levels of measurement:

- **Nominal:** Attributes are just named uniquely, without any relationship.
e.g., Serial Numbers of the sportsmen
- **Ordinal:** Attributes can be ordered by relationship.
e.g., Movie ratings
- **Interval:** It measures the distance between the attributes.
e.g., Difference between 10°C and 20°C is the same as the difference between 30°C and 40°C.
- **Ratio:** It measures the relationship between the attributes.
e.g., 10% more students in college this year

Machine Learning

Machine learning is the subfield of computer science that gives computers the ability to learn without being explicitly programmed (Arthur Samuel, 1959). Machine learning involves the design of algorithms which can learn from and make predictions on the data.

Machine learning is primarily of two types:

1. Supervised Learning
2. Unsupervised Learning

Supervised Learning

Supervised Learning is a Machine Learning task of inferring a function from analyzing labelled training data, which can be used for correctly predict for the new (test) data.

Regression algorithms and Classification algorithms fall under this category. They will be explained in the next section.

Unsupervised Learning

Unsupervised machine learning is the machine learning task of inferring a function to describe hidden structure from "unlabeled" data (a classification or categorization is not included in the observations).

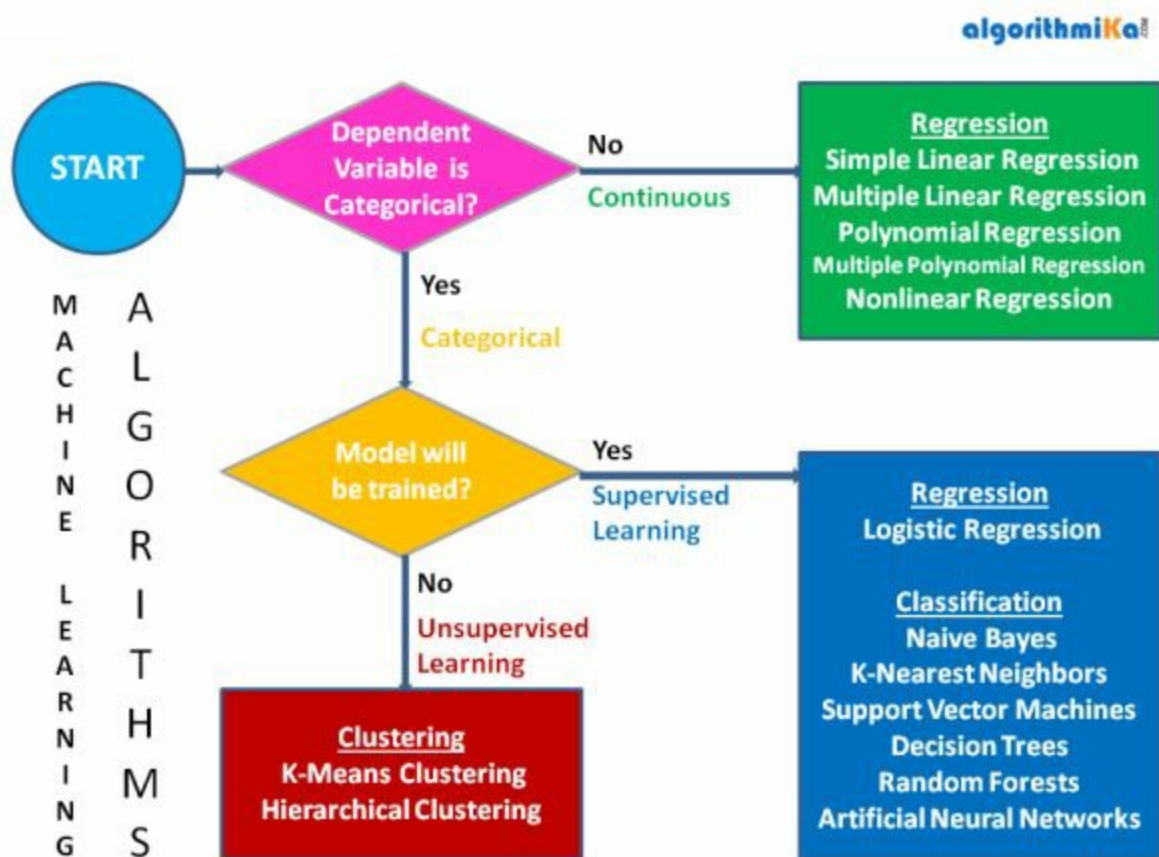
Clustering algorithms fall under this category, which will be explained in the next section.

Types of Algorithms

There are several types of algorithms that were evolved in Machine Learning, which will be used based on whether the predicted variable is continuous or categorical, and whether the learning is supervised or unsupervised.

The below chart shows the classification of the algorithms. If the dependent (or predicted) variable is continuous in nature, regression techniques are used (except, logistic). If it is categorical and if the model will be trained, then logistic regression or classification techniques are used. If the model can't be trained, clustering techniques are used.

There are many algorithms available, and the field is still evolving rapidly. We discuss here the most common and distinct algorithms.



Regression Algorithms

Regression is a form of predictive modelling technique that estimates the relationship between the dependent variable y and independent variable(s) x_i .

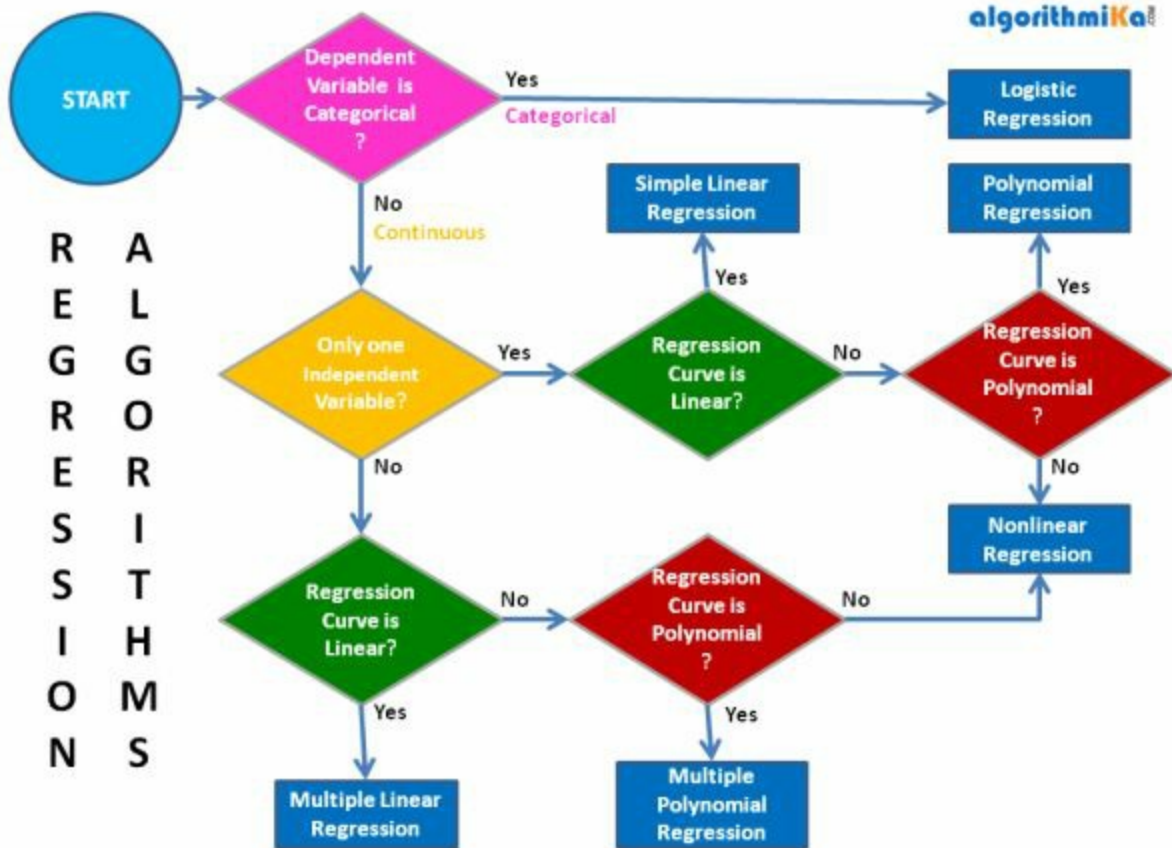
The common form of the relationship is:

$$y = f(x_i)$$

Type of regression to be used will depend on 3 parameters:

1. Type of dependent variable
2. Number of independent variables
3. Shape of the regression line to be fit

The flowchart below shows the classification of regression algorithms.



In the case where the dependent variable is continuous in nature, we have the following techniques:

- Linear Regression
- Multiple Linear Regression
- Polynomial Regression
- Multiple Polynomial Regression
- Nonlinear Regression

In the case where the dependent variable is discrete (categorical) in nature, we can use:

- Logistic Regression

Each of the techniques or algorithms is explained in detail in the later part of this book.

Classification Algorithms

Classification is the process of identifying to which category a new observation belongs, on the basis of a set of training data whose category classification is known.

The commonly used classification algorithms in machine learning are:

- Naive Bayes (NB)
- K-Nearest Neighbors (KNN)
- Support Vector Machines (SVM)
- Decision Trees (DT)
- Random Forests (RF)
- Artificial Neural Networks(ANN)

Clustering Algorithms

Clustering is the process of grouping a set of objects such a way that the objects in that group are more similar to each other than those in other groups (clusters). Means, the objects are as similar as possible within each cluster, and the clusters differ as much as possible with each other.

The commonly used clustering algorithms in machine learning are:

- K-Means Clustering (KMC)
- Hierarchical Clustering (HC)

Programming Languages for Machine Learning Algorithms

The programming languages, R and Python, have extensive Machine Learning libraries. They are also free to use, being part of the GNU General Public License and Python Software Foundation, respectively. These being the choice of academicians and researchers, they are also updated faster for the new developments in Data Science.

R



R is an open source programming language for statistical computing and graphics that is supported by the R Foundation. R is freely available under the GNU General Public License.

R is an implementation of the S programming language, and was created by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand. S was created by John Chambers while at Bell Labs.

R can be downloaded from:

<https://www.r-project.org/>

RStudio download (Rstudio is a set of integrated tools to be more productive; see License terms):

<https://www.rstudio.com/products/rstudio/download/>

Python



Python is a high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. It is gaining wider and growing usage as the language of choice for machine learning computing, due to its large library of machine learning algorithms. Python is available for free from Python Software Foundation.

Python can be downloaded from: <https://www.python.org/downloads/>

Anaconda Python Distribution (includes IPython/Jupyter environment, and NumPy, SciPy, Matplotlib, Pandas & scikit-learn libraries):
<https://www.continuum.io/downloads>

Regression Algorithms

Simple Linear Regression

Simple Linear Regression function has the dependent variable which takes continuous values, only one independent variable with the power of one, and the shape of the regression line is linear. Hence, the three parameters are:

1. Type of dependent variable: Continuous
2. Number of independent variables: One
3. Shape of the regression line to be fit: Linear

The relationship between the dependent variable and the independent variable is shown as:

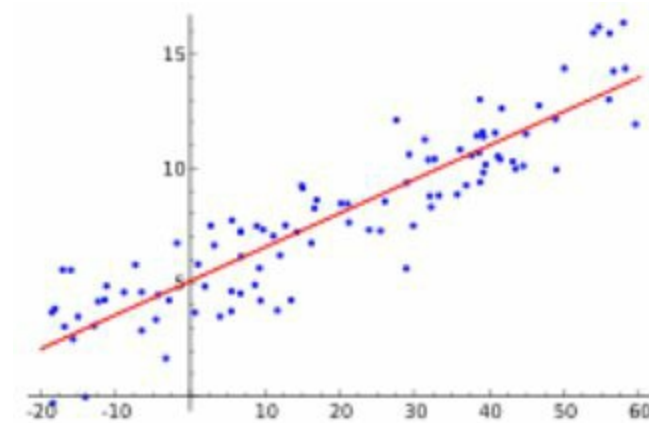
$$y = m x + c \quad m: \text{slope of the regression line, } c: \text{y-intercept (or)}$$

$$y = a + b x \quad a: \text{y-intercept, } b: \text{slope of the regression line (or)}$$

$$y = \alpha + \beta x \quad \alpha: \text{y - intercept, } \beta: \text{slope of the regression line}$$

where y is the dependent variable

and x is the independent variable



	R	Python
Package		sklearn
Library		linear_model
Function	lm	LinearRegression
Usage	<code>m <- lm(y ~x, data=mydata)</code>	<code>m = linear_model.LinearRegression().fit(x, y)</code>
Check	<code>summary(m)</code> <code>anova(m)</code>	<code>print('Coefficients: ', m.coef_)</code> <code>print('Variance score: ', m.score(x, y))</code>

Multiple Linear Regression

Multiple Linear Regression is form of modeling the relationship between a dependent variable and one or more independent variables with power of one.

Multiple Linear Regression function has the dependent variable which takes continuous values, many independent variables with the power of one, and the shape of the regression line is linear. Hence, the three parameters are:

1. Type of dependent variable: Continuous
2. Number of independent variables: Multiple
3. Shape of the regression line to be fit: Linear

The relationship between the dependent variable and the independent variables is shown as:

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots$$

where y is the dependent variable

and x_1, x_2, x_3, \dots are independent variables

and α is the y-intercept

and β_i is the coefficient of x_i

	R	Python
Package		sklearn
Library		linear_model
Function	lm	LinearRegression
Usage	<code>m <- lm(y~X, data=mydata)</code>	<code>m = LinearRegression().fit(X, y)</code>
Check	<code>summary(m)</code>	<code>print('Coefficients: ', m.coef_)</code>

	<code>anova(m)</code>	<code>print('Variance score: ', m.score(X, y))</code>
--	-----------------------	---

Significance of the variables need to be checked to make sure their p-values are less than the significance level. Insignificant variables need to be removed from the model to make the fit better, comparing the Adjusted R-Squared each time.

The following methods can be used to arrive at the best model:

- Forward Selection
- Backward Elimination
- Step-wise

Simple Polynomial Regression

Simple Polynomial Regression is a form of linear regression in which the relationship between the independent variable x and the dependent variable y is modelled as an n th degree polynomial in x .

Simple Polynomial Regression function has the dependent variable which takes continuous values, only one independent variable with the highest power more than one (n^{th} degree), and the shape of the regression line is non-linear. Hence, the three parameters are:

1. Type of dependent variable: Continuous
2. Number of independent variables: One
3. Shape of the regression line to be fit: Polynomial

The relationship between the dependent variable and the independent variable is shown as:

$$y = \alpha + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \dots$$

where y is the dependent variable

and x is the independent variable

and α is the y-intercept

and β_i is the coefficient of x^i

	R	Python
Package		numpy
Library		numpy
Function	lm	polyfit()

Usage	<pre>m <- lm(y ~ poly(x,3), data=mydata) (OR) m <- lm(y ~ x + I(x^2) + I(x^3), data=mydata)</pre>	<pre>m = numpy.polyfit(x, y, 3)</pre>
Check	<pre>summary(m) anova(m)</pre>	

Multiple Polynomial Regression

Multiple Polynomial Regression is similar to the Polynomial Regression, with multiple independent variables.

Polynomial Regression function has the dependent variable which takes continuous values, multiple independent variables with the highest power more than one (n^{th} degree), and the shape of the regression line is non-linear. Hence, the three parameters are:

1. Type of dependent variable: Continuous
2. Number of independent variables: Multiple
3. Shape of the regression line to be fit: Polynomial

The relationship between the dependent variable and the independent variable is shown as:

$$y = \alpha + \beta_{11} x_1 + \beta_{12} x_1^2 + \dots + \beta_{21} x_2 + \beta_{22} x_2^2 + \dots + \beta_{31} x_1 x_2 + \dots$$

where y is the dependent variable

and x is the independent variable

and α is the y-intercept

and β_{ij} is the coefficient of x^i

	R	Python
Package		numpy
Library		numpy
Function	lm	polyfit()

Usage	<pre>m <- lm(y ~ polym(x1, x2, degree=2, raw=TRUE)) (OR) m <- lm(y ~ x1 + x2 + I(x1^2) + I(x2^2) + x1:x2)</pre>	<pre>m = numpy.polyfit(X, y, 2)</pre>
Check	<pre>summary(m) anova(m)</pre>	

Nonlinear Regression

Nonlinear regression is a form of regression analysis in which dependent variable data are modeled by a function which is a nonlinear combination of one or more independent variables.

Nonlinear Regression function has the dependent variable which takes continuous values, one or more independent variables, and the shape of the regression line is non-linear. Hence, the three parameters are:

1. Type of dependent variable: Continuous
2. Number of independent variables: One or Multiple
3. Shape of the regression line to be fit: Nonlinear

The relationship between the dependent variable and the independent variable is shown as:

$$y = a x / (b + x) \quad (OR)$$

$$y = a + b e^x + c e^{-x} \quad (OR)$$

$$y = \frac{(a + b x)}{(c e^x + d e^{-x})}$$

where y is the dependent variable

and x is the independent variable

and a, b, c, d are the coefficients of respective parameters

	R	Python
Package		scipy
Library		optimize
Function	nls	curve_fit

Usage	<code>m <- nls(y ~ a*x/(b+x), data = mydata)</code>	<code>popt, pcov = curve_fit(func, x, y)</code>
Check	<code>summary(m)</code> <code>anova(m)</code> <code>cor(y,predict(m))</code>	

Logistic Regression

Logistic Regression (also called Logit Regression) is a regression model where the dependent variable is categorical with binary outcome, one or more independent variables, and the shape of the regression line is discrete. Hence, the three parameters are:

1. Type of dependent variable: Categorical
2. Number of independent variables: One or Multiple
3. Shape of the regression line to be fit: Discrete

The relationship between the dependent variable and the independent variables is shown as:

$$y = f(x) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots$$

where y is the dependent variable, **taking only Categorical values**

and x_1, x_2, x_3, \dots are independent variables

and α is the y-intercept

and β_i is the coefficient of x_i

In Logistic Regression, the dependent variable is binary or dichotomous, which takes the values of 1 (TRUE, Yes, Success, etc.) or 0 (FALSE, No, Failure, etc.)

The goal of logistic regression is to find the best fitting model to predict the value of the interpretable variable, which can be used to interpret the value of the dependent variable.

Transforming the relationship of the dependent and independent variables,

using logistic function, the above equation can be written as:

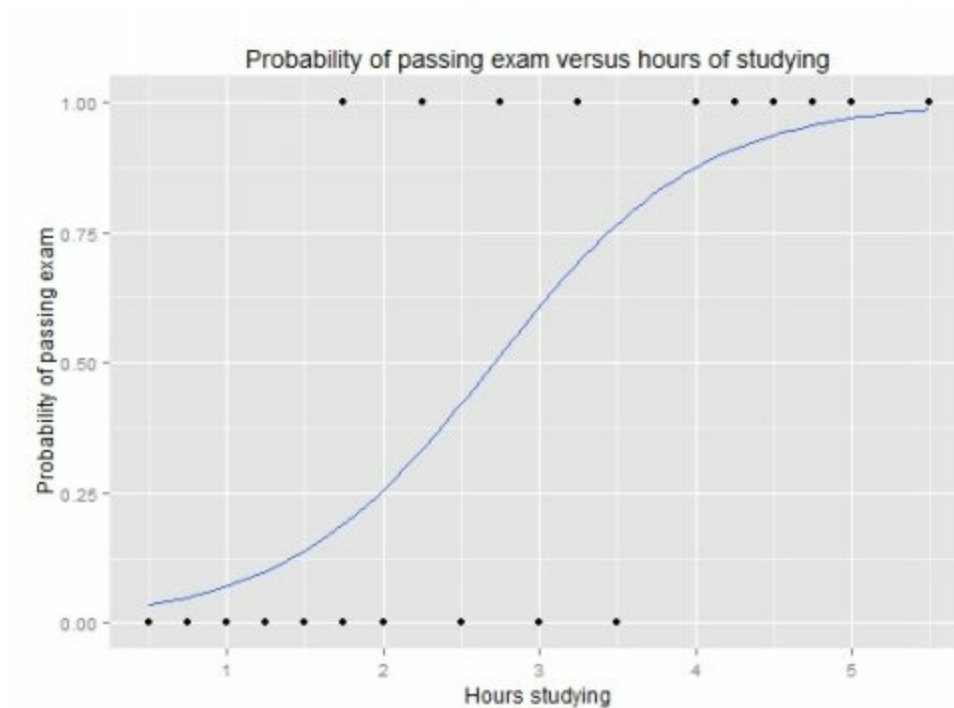
$$y = \text{logit}(p) = \ln\left(\frac{p}{1-p}\right) = f(x) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots$$

where p is the probability of presence of characteristic

$p / (1 - p)$ is called odds

Rearranging, we can obtain:

$$p = \frac{1}{1 + e^{-\text{logit}(p)}} = \frac{1}{1 + e^{-y}}$$



	R	Python
Package	caTools	sklearn
Library	caTools	linear_model
Function	glm	LogisticRegressionCV
Usage	<pre>m <- glm(y ~ x₁ + x₂ + x₃, data = mydata, family = “binomial”)</pre>	<pre>m = LogisticRegressionCV().fit(X, y)</pre>

Check	<code>summary(m)</code> <code>table(predict(m,X),y)</code> <code>#ConfusionMatrix</code>	<code>print('Coefficients: ', m.coef_)</code> <code>print('Variance score: ', m.score(X, y))</code>
--------------	--	--

Classification Algorithms

Naive Bayes Classifier

Naive Bayes classifier is a family of simple probabilistic classifiers, based on applying the Bayes' Theorem, with strong (naive) independence assumption between the features.

Naive Bayes model is simple and easy to build. It is highly scalable and outperforms even highly sophisticated classification models with large datasets. Maximum-likelihood training can be done by evaluating a closed-form expression which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

Bayes' Theorem:

$$P(A | B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B | A) P(A)}{P(B)}$$

where $P(A | B)$ is the Posterior Probability

$P(B | A)$ is the Likelihood

$P(A)$ is the Predicted Prior Probability

$P(B)$ is the predicted Prior Probability

	R	Python
Package	e1071	sklearn
Library	e1071	naive_bayes
Function	naiveBayes	GaussianNB
Usage	m <- naiveBayes(y ~ ., data = mydata)	m = GaussianNB().fit(X, y)
Check	summary(m)	print('Coefficients: ', m.coef_)

```
table(predict(m,X),y)
#ConfusionMatrix
```

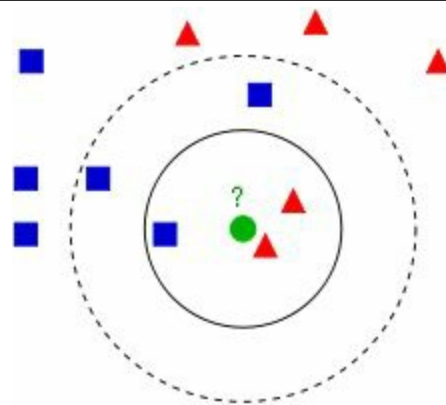
```
print('Variance score: ', m.score(x, y))
```

K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) classification uses the distance (Euclidian distance in most cases) from the test subject to its nearest k-neighbors, to classify it into the sub-class.

In the example given, test subject is shown in green. If $k=3$ (3 nearest neighbors), 3 neighbors shown in the solid circle are considered, to classify the test subject into red triangle. If $k=5$ is used, 5 neighbors shown in the dotted-line are considered, to classify the test subject as Blue square.

k needs to be chosen such a way that the error on test data will be minimized, and over-fitting is reduced.



K-Nearest Neighbors Classification

The Euclidian distance between two points p and q is the length of the line connecting them.

In Cartesian coordinates, if $p = (p_1, p_2, p_3, \dots, p_n)$ and $q = (q_1, q_2, q_3, \dots, q_n)$ are two points in n -space, The distance d between p and q is given by:

$$d(p, q) = d(q, p) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \dots + (q_n - p_n)^2}$$

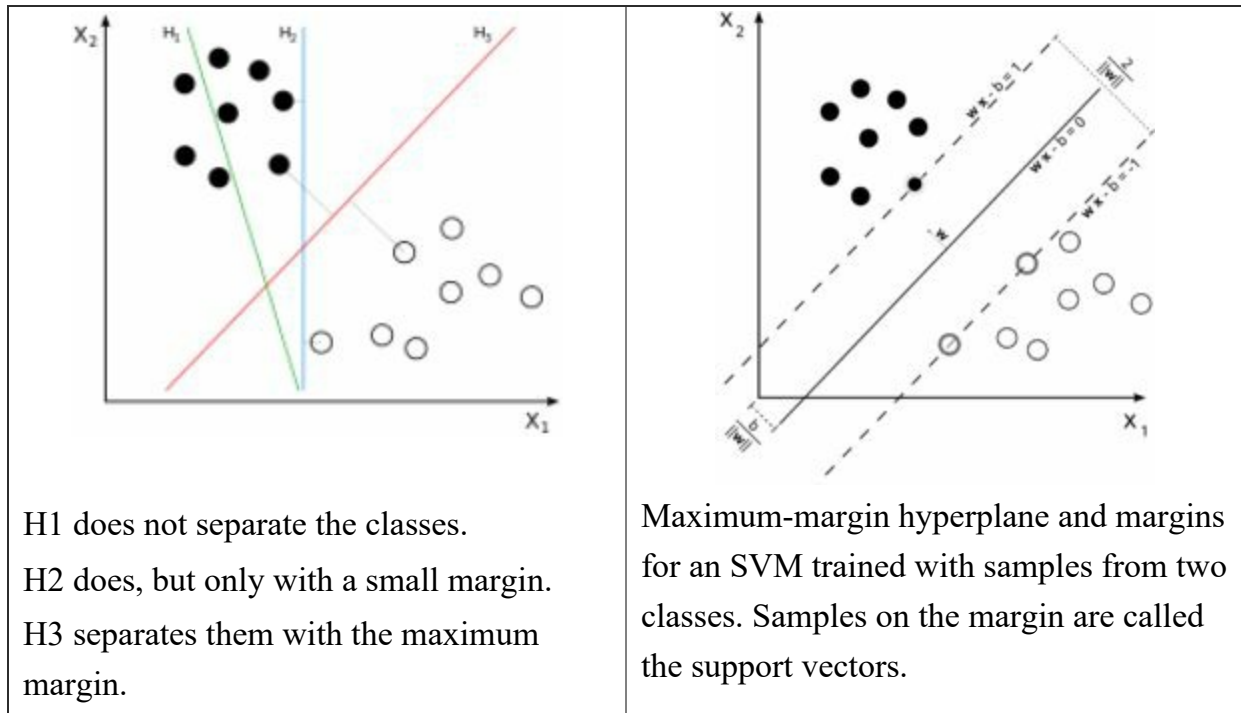
$$= \sqrt{\sum_{i=1}^n (q_i - p_i)^2}$$

	R	Python

Package	class	sklearn
Library	class	neighbors
Function	knn	NearestNeighbors
Usage	m <- knn(train, test, cl, k = 3)	m = NearestNeighbors(n_neighbors=3, algorithm='ball_tree').fit(X) # algorithm = {'auto', 'ball_tree', 'kd_tree', 'brute'}
Check	summary(m) table(predict(m,X),y) #ConfusionMatrix	

Support Vector Machines (SVM)

Support Vector Machines (SVM) model is a classification model which separates different categories, such a way that a hyperplane separates the categories with maximum margin between them.



The mathematics behind this model uses linear algebra and is a bit complicated, and beyond the scope of this booklet. It would be included in the book on ‘Data Science & Machine Learning using R and Python’ by this author, which is in works.

	R	Python
Package	e1071	sklearn
Library	e1071	svm

Function	svm	SVC
Usage	<code>m <- svm(y ~ X, data = mydata)</code>	<code>m = SVC(kernel='linear').fit(X, y)</code>
Check	<code>summary(m)</code> <code>table(predict(m,X),y)</code> <code>#ConfusionMatrix</code>	

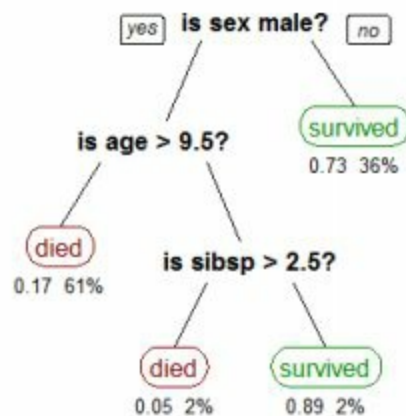
Decision Trees

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute, each branch represents the outcome of the test and each leaf node represents a class label. The paths from root to leaf represent classification rules.

The main problem with Decision Trees is that the model overfits to the training data. In order to avoid the overfitting to the training data, which gives adverse results for the test data, decision trees are pruned to remove the sections of the trees that are obscure or provide little power to classification.

There are other classification techniques based on the decision trees that work very well for classification: Random Forests classification, Bagging Decision Trees, Boosted Trees, and Rotation Forest.



	R	Python
Package	party	sklearn
Library	party	tree
Function	ctree	DecisionTreeClassifier
Usage	m <- ctree(y ~ X, data = mydata)	m = DecisionTreeClassifier().fit(X, y)
Check	summary(m) table(predict(m,X),y) #ConfusionMatrix	

Random Forests

Random Forest algorithm is an ensemble technique for classification, regression and other tasks. It operates by using Decision Trees to obtain the results and averaging them to obtain the ensemble result. Random Forest algorithm corrects the overfitting which is the main problem in Decision Tree algorithm.

An ensemble technique is a technique in which multiple results are obtained with different subsets of data using the weaker algorithm, and the results are averaged to obtain the better and stronger result.

	R	Python
Package	randomForest	sklearn
Library	randomForest	ensemble
Function	randomForest	RandomForestClassifier
Usage	<code>m <- randomForest(y ~ X, data = mydata)</code>	<code>m = RandomForestClassifier(n_estimators=100).fit(X, y)</code>
Check	<code>summary(m)</code> <code>table(predict(m,X),y)</code> <code>#ConfusionMatrix</code>	

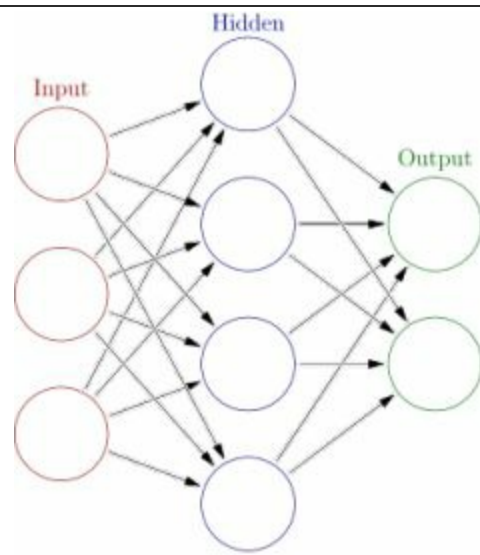
Artificial Neural Networks (ANN)

Artificial Neural Networks (ANN) is a computational approach that mimics the functioning of brain in solving problems with large clusters of biological neurons connected by axons. Each artificial neural unit is called node and is connected with many others and have a summation function which combines all of input values together.

An artificial neural network (ANN) is an interconnected group of nodes, similar to the large network of neurons in a brain. Here, each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another.

ANN will generally have one Input layer, several Hidden layers, and one Output layer.

Neural networks are based on real numbers, with the value of the core and of the axon typically being a representation between 0.0 and 1.



Artificial Neural Networks (ANN)

Artificial Neural Networks (ANN) are being used to solve wide variety of problems, like Computer Vision and Speech Recognition, that are hard to solve using the regular rule-based programming techniques.

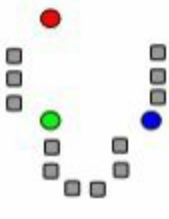
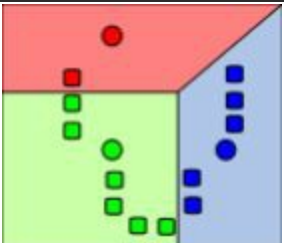
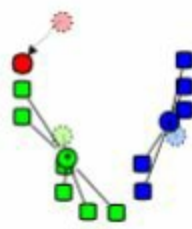
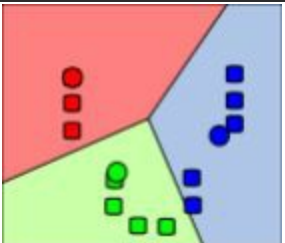
	R	Python
Package	neuralnet	sklearn
Library	neuralnet	neural_network
Function	neuralnet	MLPClassifier
Usage	m <- neuralnet(y ~ X, data = mydata)	m = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2), random_state=1).fit(X, y)
Check	summary(m) table(predict(m,X),y) #ConfusionMatrix	

Clustering Algorithms

K-Means Clustering

K-Means Clustering is the clustering technique that is used to partition n observations into k clusters, in which each observation is assigned to the cluster with the nearest mean.

Initial centroids are randomly generated within the given data. Each observation is assigned to the closest centroid, forming the clusters. For each cluster, the centroids are computed again. The process repeats until there is no change in the assignment of observations to the clusters.

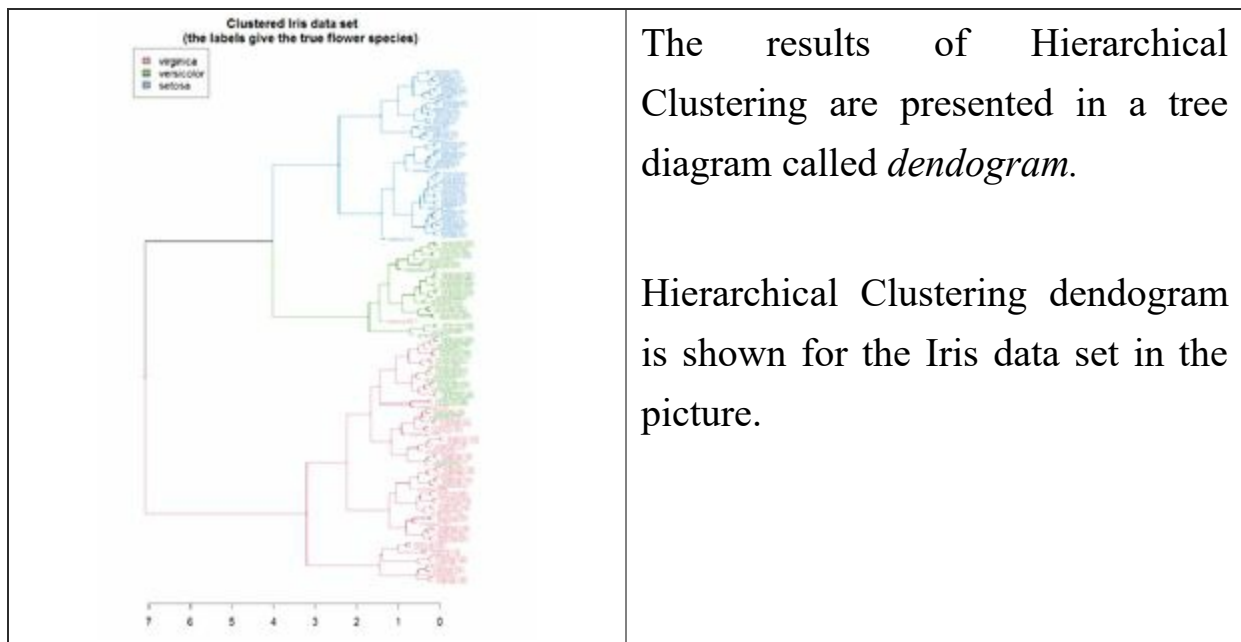
 <p>1. k initial "means" (in this case $k=3$) are randomly generated within the data domain (shown in color)</p>	 <p>2. k clusters are created by associating every observation with the nearest mean.</p>	 <p>3. The centroid of each of the k clusters becomes the new mean.</p>	 <p>4. Steps 2 and 3 are repeated until convergence has been reached.</p>
	R	Python	
Package	cluster	sklearn	
Library	cluster	cluster	
Function	kmeans	KMeans	
Usage	<code>m <- kmeans(X, centers =3, nstart=20)</code>	<code>m = KMeans(n_clusters=3, random_state=0).fit(X)</code>	

Check	summary(m) table(m, y) #Confusion Matrix	
--------------	---	--

Hierarchical Clustering

Hierarchical Clustering is a type of clustering analysis which seeks to build hierarchy of clusters. The Hierarchical Clustering approach generally falls into two types:


1. Agglomerative: This is a “bottom up” approach. Each observation starts in its own cluster and pairs of clusters are merged as one moves up the hierarchy.
2. Divisive: This is a “top-down” approach. All observations are started in one cluster, and splits are performed recursively as one moves down the hierarchy.



	R	Python
Package	e1071	sklearn
Library	e1071	cluster
Function	hclust	AgglomerativeClustering
Usage		

	<pre>m <- hclust(dist(X), method="ave")</pre>	<pre>m = AgglomerativeClustering(n_clusters=n_clusters, linkage="average", affinity="Euclidian").fit(X) # affinity = {"euclidean", "l1", "l2", "manhattan", "cosine"} # linkage = {"ward", "complete", "average"}</pre>
Check	<pre>summary(m) plot(m) #Dendogram table(m, y) #Confusion Matrix</pre>	

RESOURCES

	<p data-bbox="597 401 948 457">algorithmika.com</p> <p data-bbox="430 478 1117 661"><i>salutes the authors and publishers who are providing these resources for FREE, keeping the mission of educating humanity over their own financial benefits.</i></p>
---	---

E-BOOKS (FREE)

1. An Introduction to Statistical Learning with Applications in R: by Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani (Springer, Reg. \$79.99) <http://www-bcf.usc.edu/~gareth/ISL/>
2. The Elements of Statistical Learning: Data Mining, Inference, and Prediction: by Trevor Hastie, Robert Tibshirani, and Jerome Friedman (Springer, Reg. \$89.95) <http://statweb.stanford.edu/~tibs/ElemStatLearn/>
3. An Introduction to R
<https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf>
4. R for Beginners
https://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf
5. R Graphics Cookbook; by Winston Chang (O'Reilly, Reg. \$39.99)
<https://ase.tufts.edu/bugs/guide/assets/R%20Graphics%20Cookbook.pdf>
6. A Byte of Python: by Swaroop C H
<https://www.gitbook.com/download/pdf/book/swaroopch/byte-of-python>
7. Think Python: by Allen B. Downey
<http://greenteapress.com/thinkpython/thinkpython.pdf>

ONLINE, OPEN-ACCESS TEXT BOOKS

1. Statistical foundations of machine learning: by Gianluca Bontempi and Souhaib Ben Taieb:

<https://www.otexts.org/book/sfml>

2. Applied biostatistical analysis using R: by Stephen B. Cox

<https://www.otexts.org/book/biostat>

3. Online Learning – RStudio

<https://www.rstudio.com/online-learning/>

4. Forecasting: Principles and Practice: by Hyndman & Athanasopoulos (OTexts, Reg. \$43.23)

<https://www.otexts.org/book/fpp>

KNOWLEDGE RESOURCES (Indispensable for a Data Scientist)

1. Kaggle: Home of Data Science Datasets, Competitions, and more...

<https://www.kaggle.com>

2. Github: Development Platform for developers, from OpenSource to Business. Most recent developments of functions may be viewed and obtained from Github.

<https://www.kaggle.com>

3. algorithmiKa.com: Web-site by the same author to learn Data Science, and expand the knowledge.

<http://www.algorithmiKa.com>

4. AnalyticsVidhya.com: Web-site to further the knowledge and new developments in Data Science.

<http://www.AnalyticsVidhya.com>

5. DataScienceCentral.com: Web-site to further the knowledge and new developments in Data Science

<http://www.DataScienceCentral.com>

REFERENCES

1. Microsoft Machine Learning Algorithms Cheat Sheet:
<https://docs.microsoft.com/en-us/azure/machine-learning/machine-learning-algorithm-cheat-sheet>
2. Solving a Machine Learning Problem:
http://scikit-learn.org/stable/tutorial/machine_learning_map/
3. Exploratory Data Analysis:
<http://www.itl.nist.gov/div898/handbook/eda/eda.htm>
4. UCLA Resources to learn and use R:
<http://www.ats.ucla.edu/stat/R/>
5. Princeton University – Statistical Packages – R:
http://dss.princeton.edu/online_help/stats_packages/r

DATASETS

The following are the sources of the datasets for analysis:

1. UCI (University of California at Irvine) Machine Learning Repository:

<https://archive.ics.uci.edu/ml/datasets.html>

<http://mlr.cs.umass.edu/ml/datasets.html>

2. Official U.S. government site providing increased public access to federal government datasets:

<https://www.data.gov/>

3. Kaggle datasets:

<https://www.kaggle.com/datasets>

4. Amazon Web Services (AWS) Public Datasets:

<https://aws.amazon.com/datasets/>

<https://aws.amazon.com/public-datasets/> (Large Datasets Repository)

5. IMF Data:

<http://datahelp.imf.org/knowledgebase/articles/505304-dataset-portal>

<http://www.imf.org/en/Data>

6. World Bank Data:

<http://data.worldbank.org/>

7. Columbia Million Songs dataset:

<http://labrosa.ee.columbia.edu/millionsong/>

8. CMU StatLib Datasets:

<http://lib.stat.cmu.edu/datasets/>

9. 100,000 ratings and 1,300 tag applications applied to 9,000 movies by 700 users.

<http://grouplens.org/datasets/movielens/ml-latest-small.zip>

<http://grouplens.org/datasets/movielens/>

<http://grouplens.org/datasets/> (many other types of datasets)

Please
let us know about
your valuable
suggestions at:

algorithmika.com@gmail.com

THANK YOU!