

TUTORIAL

Foundations of Machine Learning and Data Science for Developers

By Ajit Jaokar

TABLE OF CONTENTS

Introduction	3
An Overview of Data Science and Machine Learning	4
What is Machine Learning?	5
What Algorithms Does Machine Learning Involve?	6
Supervised vs. Unsupervised Learning	7
Data Science Flow/Steps	7
Pre-Processing	8
Exploratory Data Analysis	8
Feature Engineering	9
Machine Learning	10
Machine Learning Algorithm Performance Metrics	10
Ensemble Strategies	10
Visualization	11
Steps in a Typical Data Science Program	12
Understanding Linear Regression	12
The Equation of a Straight line	12
Linear Regression	14
The Normal Distribution	17
Generalized Linear Model	18
Non-linear Regression and Non-linear classification	20
Additional Concepts	22
Model Fitting or Curve Fitting	22
Cross Validation	22
Algorithms	23
Linear Regression	23
Linear Classification : Logistic Regression	24
Non-Linear Regression – k-Nearest Neighbour	26
Non-Linear Classification – Neural Networks	27
Unsupervised Learning: Clustering – k-Means Algorithm	27
Which Maths and Stats Techniques do You Need for Data Science?	28
Conclusion	29

Introduction

This tutorial introduces machine learning and Data Science concepts for developers.

On the Web, we already have many excellent resources for learning Data Science, however, the sheer amount of material can, in itself, be daunting.

This tutorial draws on the content on the Web and adds three new elements:

- a) We explain concepts simply but in context. Many tutorials explain one specific aspect but do not show how it fits into the wider picture.
- b) We approach maths and stats from the perspective/lens of linear regression. This approach has an advantage because linear regression is familiar to many from their high school days.
- c) Less is more – we discuss a small number of examples. You can always find more examples of an algorithm type on the Web.

Future versions of this tutorial will expand on Maths and Science in context of programming.

If you want to engage with me and are based in the UK, I teach at [Oxford University \(Data Science for Internet of Things\)](#). If you are based globally, I have a new online course on [Enterprise AI Applications](#).

Comments and feedback welcome at ajit.jaokar@futuretext.com

An important disclaimer: This tutorial is a personal effort. It is not associated with any organizations I am associated with. Thus, all errors and omissions are mine alone.

An Overview of Data Science & Machine Learning

Data Science is a multidisciplinary domain involved in solving analytically complex problems using machine learning algorithms. Unlike traditional algorithms, machine learning algorithms are data driven. Thus, the data determines the algorithm's response. ¹For example, for an application to detect photos, a non-machine learning algorithm would try to first define what a face is. In contrast, a machine learning algorithm would not have such a pre-determined definition but rather it would "learn-by-examples".

Data Science includes a formal set of steps. Hilary Mason and Chris Wiggins describe a process called OSEM²N which is summarized as

1. **Obtain Data**
2. **Scrub Data**
3. **Explore Data:** 'Explore' in this context refers to exploratory data analysis; i.e. no hypothesis that is being tested and no predictions that are being evaluated.
4. **Model Data (machine learning stage):** Here, we apply the predictive models (regression, classification,etc.). A model's predictive accuracy can be evaluated by how well it performs on unseen data.
5. **Interpret Results**

Broadly, we will also follow this sequence in this tutorial.

What is Machine Learning?

As we see above, Machine learning can be seen as a step in the wider Data Science process.

Tom Mitchell in his book Machine Learning³ :

“The field of machine learning is concerned with the question of how to construct computer programs that automatically improve with experience.”

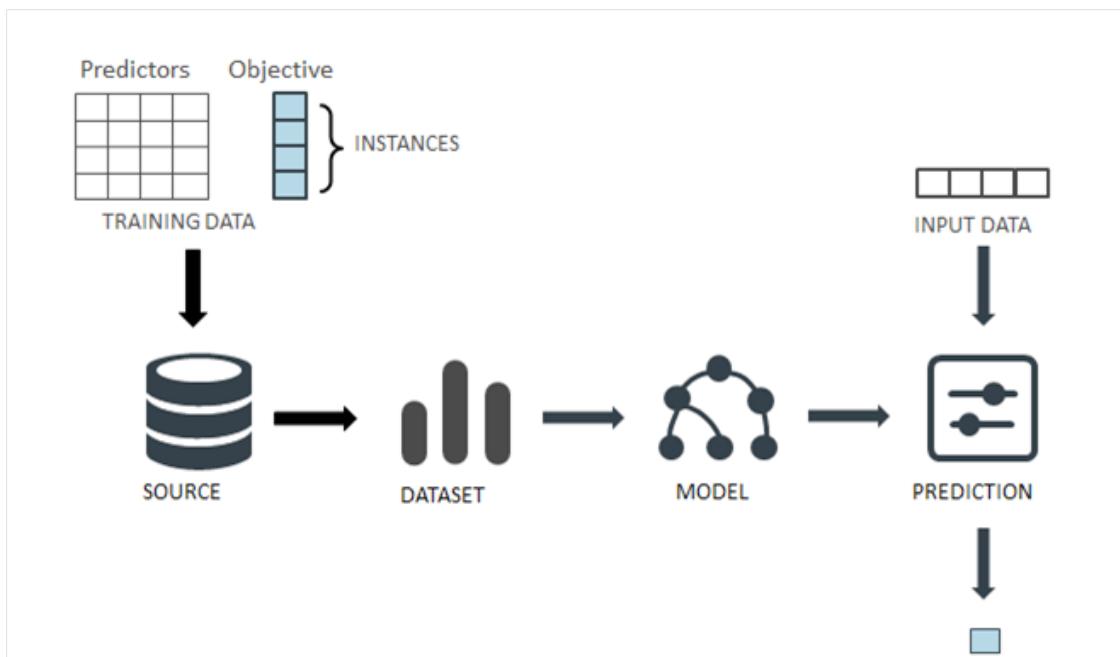
Formal definition of machine learning:

“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”

From a programmer’s perspective Machine Learning Involves:

- Training of a model from data.
- Predict/extrapolate a decision.
- Measure against a performance measure.

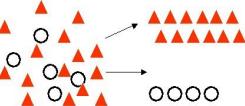
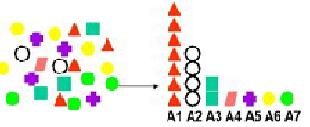
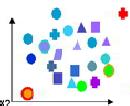
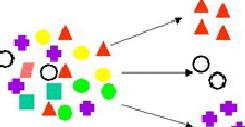
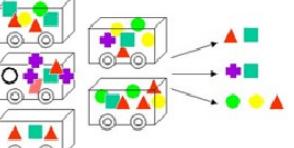
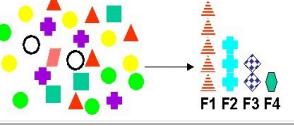
Visually, we can see this process as:



Source unknown

What Algorithms Does Machine Learning Involve?

The basic idea of machine learning is to build a mathematical model based on data and then use that model to predict results for new data. As the process is repeated, the model itself evolves based on new conditions. Below is a summary of the common models used in machine learning:

Technique	Applicability
Classification 	Most commonly used technique for predicting a specific outcome such as response/no-response, high/medium/low-value customer, likely to buy/not buy.
Regression 	Technique for predicting a continuous numerical outcome such as customer lifetime value, house value, or process yield rates.
Attribute Importance 	Ranks attributes according to strength of relationship with target attribute. Use cases including factors most associated with customers who respond to an offer, factors most associated with healthy patients.
Anomaly Detection 	Identifies unusual or suspicious cases based on deviation from the norm. Common examples include health care fraud, expense report fraud, and tax compliance.
Clustering 	Useful for exploring data and finding natural groupings. Members of a cluster are more like each other than they are like members of a different cluster. Common examples include finding new customer segments and life sciences discovery.
Association 	Finds rules associated with frequently co-occurring items, used for market basket analysis, cross-sell, and root cause analysis. Useful for product bundling, in-store placement, and defect analysis.
Feature Selection and Extraction 	Produces new attributes as a linear combination of existing attributes. Applicable for text data, latent semantic analysis, data compression, data decomposition and projection, and pattern recognition.

Source: Oracle

Supervised vs. Unsupervised Learning

Machine learning algorithms are classified into **supervised** and **unsupervised**. In supervised learning, “learn-by-examples,” output datasets are provided which are used to train the machine and get the desired outputs. In unsupervised learning, no datasets are provided; instead, the data is clustered into different classes. In an unsupervised algorithm your examples are not labelled; i.e. the algorithm tries to cluster similar concepts. For example, in image detection, the algorithm would differentiate between images of ‘faces’ and images of ‘horses’.

In addition, we also have **semi-supervised** and **active learning**. Technically, these are also supervised methods but with the difference that there is some way to avoid a large number of labelled examples. **Semi-supervised learning** is halfway between supervised and unsupervised learning. In addition to unlabeled data, the algorithm is provided with some supervision information – but not necessarily for all examples. For instance, there may be constraints such as “these points have (or do not have) the same target.”⁴ **In active learning**, the algorithm itself decides which thing you should label, but it might ask you to confirm. In addition, there are other kinds of learning, such as **reinforcement learning**, whereby the learning method interacts with its environment by producing actions based on a system of rewards and punishments.⁵

Data Science Flow/Steps

In this section, we outline flow/steps for Data Science. Note that these processes may be often iterative.

Pre-Processing

In Data Science, pre-processing refers to the transformations applied to your data before feeding it into the algorithm.⁶ The step is needed because algorithms perform better with data in specific formats. Also, there may be issues in the data itself (i.e. differing formats) that need to be corrected before we apply predictive algorithms. There are many options for pre-processing; for example, scikit-learn⁷. Pre-processing includes steps such as bulk data loading, dealing with problematic data, dealing with large datasets, managing data formats, working with categorical and textual data, converting data to numbers, data transforms, rescale data, standardize data, normalize data, etc.

Exploratory Data Analysis

In statistics, exploratory data analysis (EDA) is an approach for analyzing data sets to summarize their main characteristics, often with visual methods. EDA uses a number of techniques such as: Box plot, histogram, scatter plot, principal component analysis, etc.

EDA techniques are used to detect the underlying structure of data, including:⁸

1. Maximize insight into a data set.
2. Uncover underlying structure.
3. Extract important variables.
4. Detect outliers and anomalies.
5. Test underlying assumptions.
6. Develop parsimonious models.
7. Determine optimal factor settings.

Note: A parsimonious model is a model that accomplishes a desired level of explanation or prediction with as few predictor variables as possible⁹. In EDA, the goal is to postpone the assumptions about the kind of model and allow the data itself to reveal its underlying structure and model. Graphics help to communicate and explore the underlying data.

Feature Engineering

In machine learning and pattern recognition, a feature is an individual measurable property of a phenomenon being observed. Choosing informative, discriminating, and independent features is a crucial step for effective algorithms in pattern recognition, classification, and regression. Features are usually numeric (or converted to numeric values). The initial set of features identified can be redundant and too large to be managed. Therefore, before we run machine learning algorithms, we often select a subset of features, or we construct a new and reduced set of features. This process is called feature engineering. Feature engineering is subjective and the features you choose and combine make a difference to the output. For example, in spam detection algorithms, features may include the presence or absence of certain email headers, email structure, language, frequency of specific terms, grammatical correctness of the text, etc.¹⁰

Feature engineering can be both difficult and expensive. The process is also iterative. Since a feature is a piece of information that might be useful for prediction; any attribute could be a feature, as long as it is useful to the model. Features should be understood in the context of a problem. A feature is a characteristic that might help when solving the problem. Data often contains many features that are either redundant or irrelevant, and can thus be removed without incurring much loss of information.

Feature selection techniques should be distinguished from feature extraction. Feature extraction creates new features from functions of the original features, whereas feature selection returns a subset of the features. Feature selection techniques are often used in domains where there are many features and comparatively few samples (or data points). Finally, deep learning networks can be seen to automatically extract features without human intervention.¹¹ Dimensionality reduction, principal component analysis etc are all techniques used in feature engineering.

Machine Learning

In the machine learning stage, we actually apply the predictive algorithms, which we describe in other sections of this tutorial.

Machine Learning Algorithm Performance Metrics

Once we have a predictive model, we need to understand how well the model performs. In this stage, we consider how the model performs. This includes operations like splitting into train and test sets, k-fold cross validation, classification metrics, and regression metrics

Ensemble Strategies

To improve the accuracy of your results further, we consider **ensemble modelling**.

Ensemble modelling is the process of running two or more related but different analytical models and then synthesizing the results into a single score or spread in order to improve the accuracy of predictive analytics and data mining applications.¹²

Consider the standard supervised learning problem. A learning program is given training examples for some unknown function (examples are called features) and the expected (y) values.

For classification, given a set S of training examples, a learning algorithm outputs a classifier. The classifier is a hypothesis about the true function f. Given new X values, the classifier predicts the corresponding Y values. In this scenario, an **ensemble of classifiers** is a set of classifiers whose individual

decisions are combined in some way. Ensembles are often much more accurate than the individual classifiers that constitute them. For this to happen, the constituent classifiers must be accurate and diverse. An accurate classifier is one that has an error rate of better than random guessing on new X values. Two classifiers are diverse if they make different errors on new data points

There are three reasons why ensembles work:

- a) Statistically, a learning algorithm can be viewed as searching a space H of hypotheses to identify the best hypothesis in the space. The statistical problem arises when the amount of training data available is too small compared to the size of the hypothesis space. Without sufficient data, the learning algorithm can find many different hypotheses in H that all give the same accuracy on the training data. By constructing an ensemble out of all of these accurate classifiers, the algorithm can average their votes and reduce the risk of choosing the wrong classifier.
- b) Many learning algorithms work by performing some form of local search. These may get stuck in local optima. For example, neural network algorithms employ gradient descent to minimize an error function over the training data. An ensemble constructed by running the local search from many different starting points may provide a better approximation to the true unknown function than any of the individual classifiers.
- c) In most applications of machine learning, the true function f cannot be represented by any of the hypotheses. Thus, by forming weighted sums of hypotheses, it may be possible to expand the space of representable functions to create an accurate result.

Source adapted from ¹³

Visualization

Visualization has a key role to play in Data Science because it helps to convey the story. Story telling is regarded as one of the vital functions in Data Science.¹⁴

Steps in a Typical Data Science Program

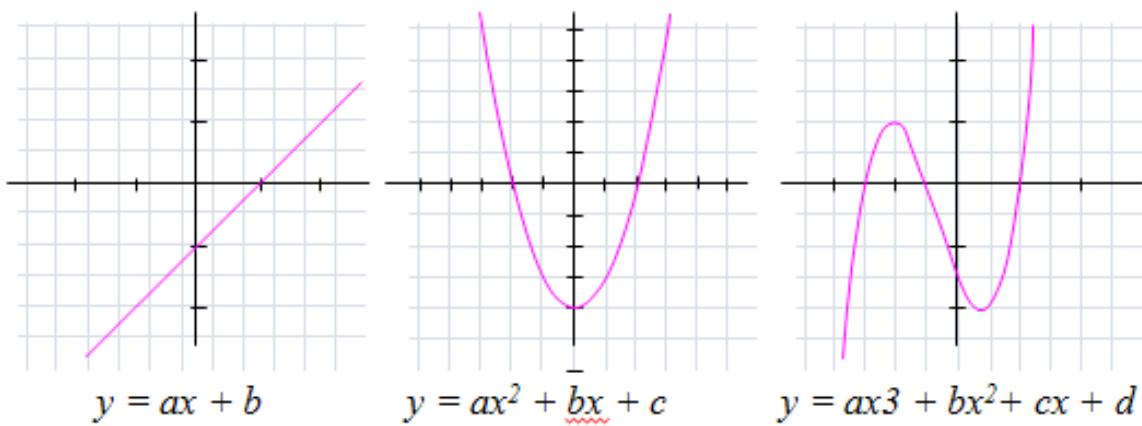
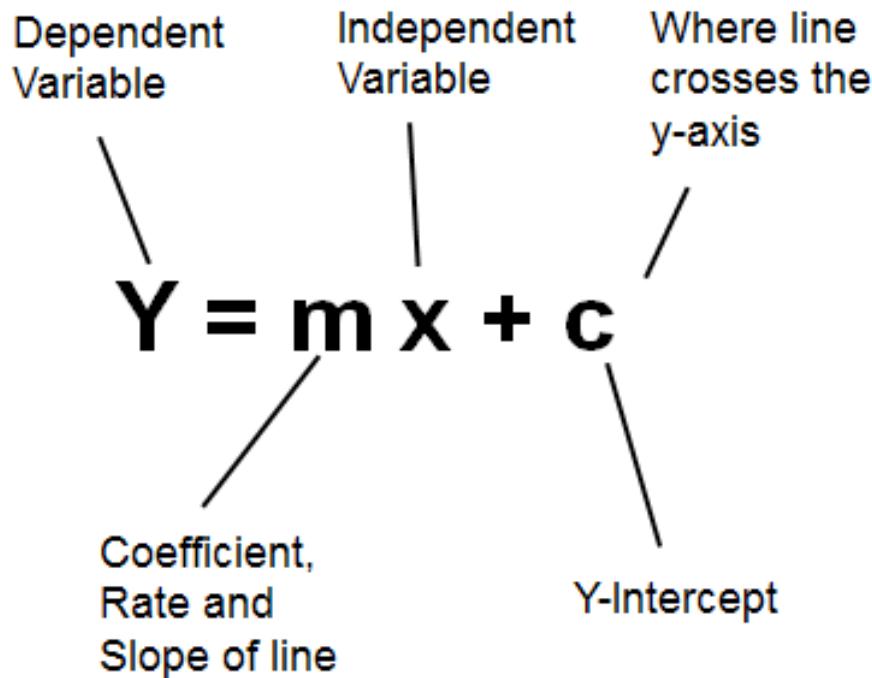
We now look at a set of steps for a typical Data Science program:
A Data Science program typically includes the following steps:

- Load libraries.
- Load data.
- Understand the data and also the features and the predictors.
- Create the data and the target objects.
- Create a linear regression object.
- Train the model (fit ..).
- Make predictions.
- Compute the error (i.e. ex root mean square error).

UNDERSTANDING LINEAR REGRESSION

The Equation of a Straight line

We approach the study of Data Science from the ‘lens’ of linear regression; i.e. we study linear regression in detail and explore other forms of regression and classification as from the perspective of linear regression. This approach has some advantages, mainly that most people learned the equation of a straight line in school. The equation of a straight line is represented as $y=mx+c..$. The slope of the line is known as the gradient and is represented by m in the equation. The point at which the line crosses the y -axis is represented by c^{15} .



Before we explore this equation further, it is important to realize that a straight line is a special case of a curve (with the higher order terms with parameters zero). This can be seen above. Of all the curves, only straight lines have the property that the shortest distance between two points on the curve can be found by traveling along the curve. Even Euclid, in his book Elements¹⁶, used one word, γραμμή, which covered all lines, both straight and curved¹⁷. He added a second word, εὐθεία, an adjective for straight, whenever he referred to a straight line¹⁸.

Linear Regression

The above property can be used to extend the idea of the equation $y = mx + c$ to a more generic form. We have seen that the equation of a straight line takes the form $y = mx + c$.

More generically, we could extend this equation to n dimensions by saying:

$$y_1 = c + B_1 * X_1 + B_2 * X_2 + B_3 * X_3 + \dots + B_n * X_n$$

where:

- B_1, B_2, \dots, B_n are coefficients of the nth attribute
- $X_1, X_2, X_3, \dots, X_n$ are values of the nth attribute and
- C is the intercept and is constant.

Hence, we could say that linear regression is an approach for modelling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables) denoted X . A linear regression equation is modelled as a straight line or as a plane in three dimensions (hence the name linear)

Linear regression was the first type of regression analysis to be studied rigorously, and to be used extensively in practical applications. This is because models which depend linearly on their unknown parameters are easier to fit than models which are non-linearly related to their parameters, and because the statistical properties of the resulting estimators are easier to determine.

Linear regression has many practical uses. Most applications fall into one of the following two broad categories:

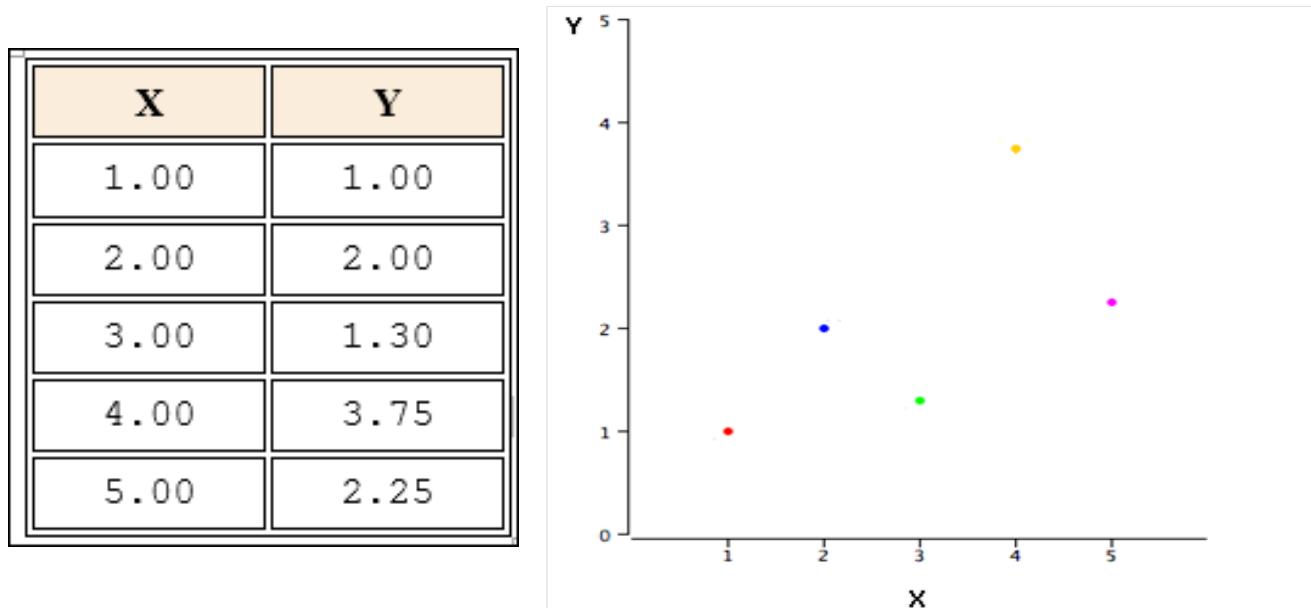
- If the goal is prediction, forecasting, or error reduction, linear regression can be used to fit a predictive model to an observed data set of y and X values. After developing such a model, if an additional value of X is given without its accompanying value of y , the fitted model can be used to make a prediction of the value of y .
- Given a variable y and a number of variables X_1, \dots, X_p that may be related to y , linear regression analysis can be applied to quantify the

strength of the relationship between y and the X_j , to assess which X_j may have no relationship with y at all, and to identify which subsets of the X_j contain redundant information about y .

Adapted from Wikipedia

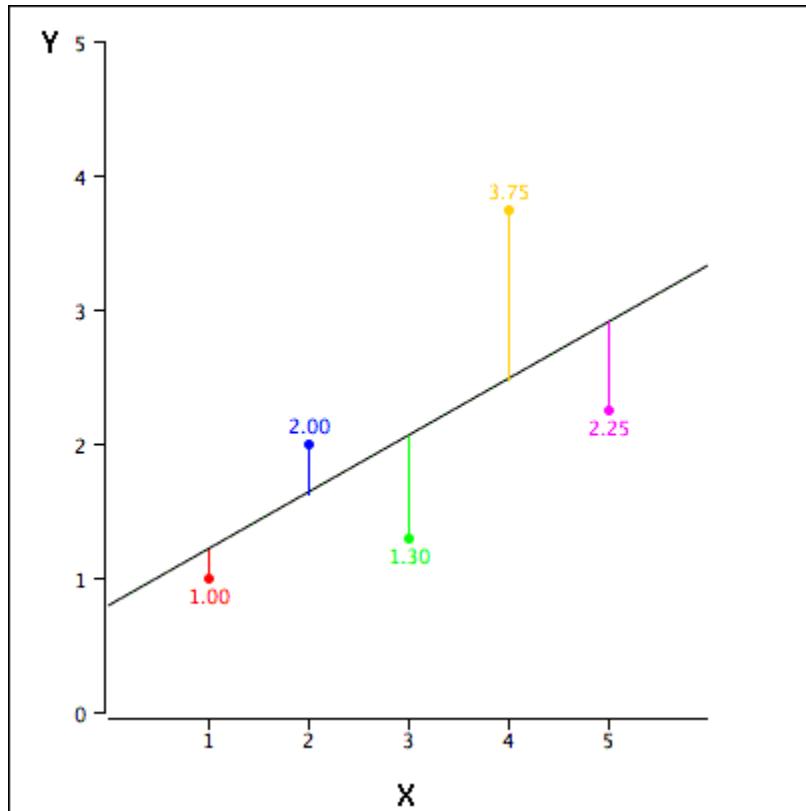
A worked example for Linear Regression is presented in the book by David Lane (adapted below).¹⁹

You can see that there is a positive relationship between X and Y . If you were going to predict Y from X , the higher the value of X , the higher your prediction of Y .



(A scatter plot example of above data.)

Linear regression consists of finding the best-fitting straight line through the points. The best-fitting line is called a *regression line*. The black diagonal line below is the regression line and consists of the predicted score on Y for each possible value of X . The vertical lines from the points on the regression line represent the errors of prediction. As you can see, the red point is very near the regression line; its error of prediction is small. By contrast, the yellow point is much higher than the regression line and therefore its error of prediction is large.



The black line contains predictive values, the points are the actual data, and the vertical lines between the points and the black line represent errors of prediction. The error of prediction for a point is the value of the point minus the predicted value (the value on the line).

Below, we see the predicted values (\hat{Y}) and the errors of prediction ($Y - \hat{Y}$). For example, the first point has a Y of 1.00 and a predicted Y (called \hat{Y}) of 1.21. Therefore, its error of prediction is -0.21.

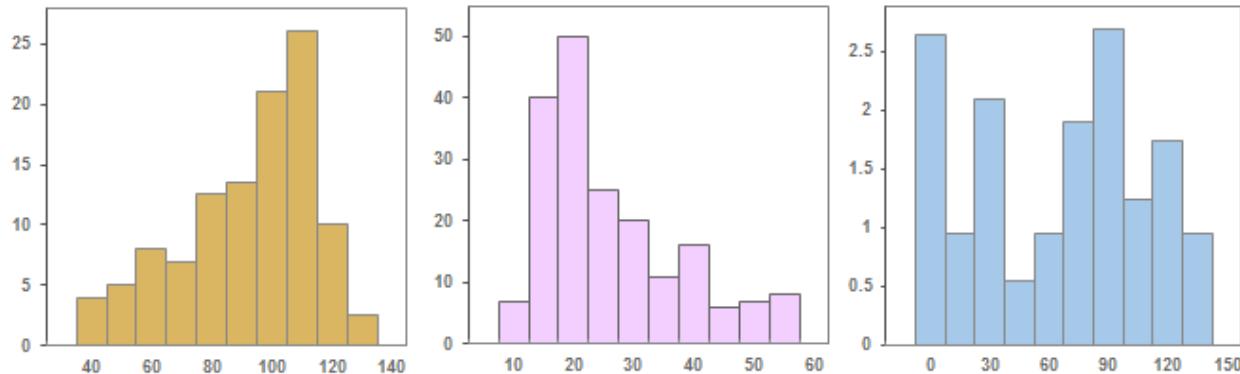
X	Y	\hat{Y}	$Y - \hat{Y}$	$(Y - \hat{Y})^2$
1.00	1.00	1.210	-0.210	0.044
2.00	2.00	1.635	0.365	0.133
3.00	1.30	2.060	-0.760	0.578
4.00	3.75	2.485	1.265	1.600
5.00	2.25	2.910	-0.660	0.436

The most commonly-used criterion for the best-fitting line is the line that minimizes the sum of the squared errors of prediction. That is the criterion that was used to find the line. The last column shows the squared errors of prediction. The sum of the squared errors of prediction shown in the table is lower than it would be for any other regression line.

The Normal Distribution

To understand more about linear regression, we have to understand normal distribution.

Data can be distributed /spread out in many ways:



However, there are many cases where the data tends to be around a central value with no bias left or right. This distribution is called a “normal distribution” (also called a Bell curve because of the shape of the Graph).

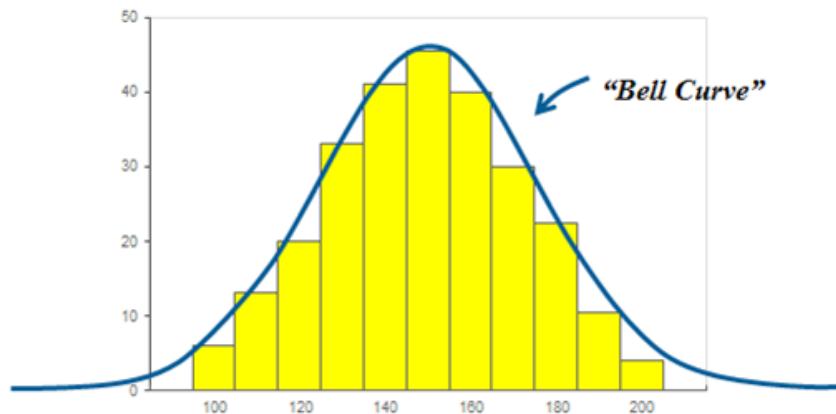
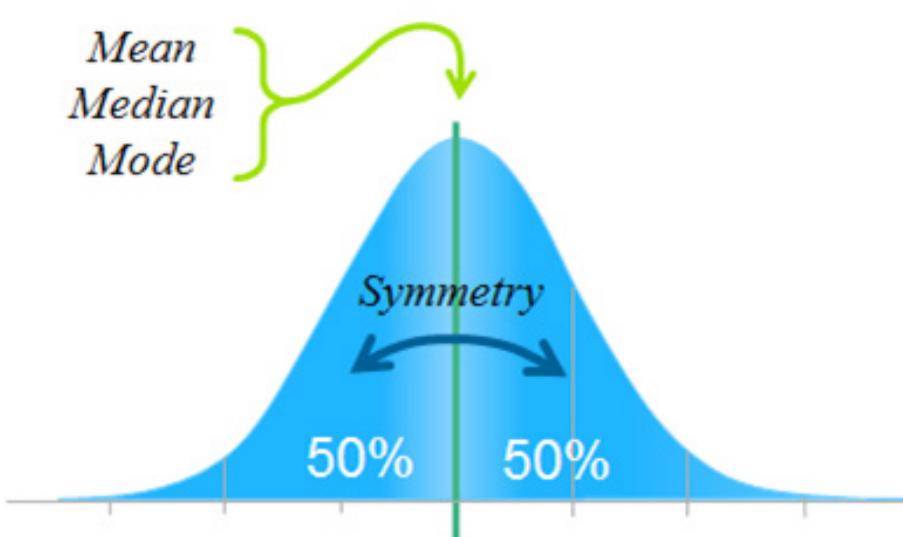


Image source²⁰

Many things closely follow normal distribution patterns, including heights of people, the size of things produced by machines, errors in measurements, blood pressure, marks on a test, etc.

When collected data is clustered in such a manner, we say the data is “normally distributed.”



The **normal distribution** has mean = median = mode and thus a symmetry around the centre, which means 50% of the values are less than the mean and 50% of the values are more than the mean.

Generalized Linear Model

To recap, when we say that linear regression requires a linear model, we mean that each term is either a constant or the product of a parameter and a predictor; i.e. a linear combination of the form:

Response = constant + parameter * predictor + ... + parameter * predictor

$$Y = b_0 + b_1 X_1 + b_2 X_2 + \dots + b_k X_k$$

Ordinary linear regression predicts the expected value of a given unknown quantity (the response variable, a random variable) as a linear combination of a set of observed values (predictors). This implies that a constant change in a predictor leads to a constant change in the response variable (i.e. a linear-response model). This is appropriate when the response variable has a normal distribution (this works for any quantity that only varies by a relatively small amount; e.g. human heights).

However, these assumptions are inappropriate for some types of response variables. For example, constant input changes lead to geometrically varying, rather than constantly varying, output changes. As an example, a prediction model might predict that 10 degree temperature decrease would lead to 1,000 fewer people visiting the beach is unlikely to generalize well over both small beaches (e.g. those where the expected attendance was 50 at a particular temperature) and large beaches (e.g. those where the expected attendance was 10,000 at a low temperature). The problem with this kind of prediction model would imply a temperature drop of 10 degrees would lead to 1,000 fewer people visiting the beach, a beach whose expected attendance was 50 at a higher temperature would now be predicted to have the impossible attendance value of -950. (Example adapted from Wikipedia.)

Logically, a more realistic model would instead predict a constant rate of increased beach attendance (e.g. an increase in 10 degrees leads to a doubling in beach attendance, and a drop in 10 degrees leads to a halving in attendance). Such a model is termed an exponential-response model (or log-linear model, since the logarithm of the response is predicted to vary linearly).

This idea that a function (in this case, a logarithm of X) would transform a non-normal distribution into a distribution which varies linearly with the Y variable is critical to understanding Generalized Linear Models

If you understand this, then you can understand that

- a) Linear regression becomes a special case where there is no 'link function' so to speak and, further;
- b) Including a link function allows us to include a range of distributions and models because they can be transformed into linear functions)

Similarly, a model that predicts a probability of making a yes/no choice (a Bernoulli variable) is even less suitable as a linear-response model, since probabilities are bounded on both ends (they must be between 0 and 1). Imagine, for example, a model that predicts the likelihood of a given person going to the beach as a function of temperature. A reasonable model might predict, for example, that a change in 10 degrees makes a person two times more or less likely to go to the beach. But what does "twice as likely" mean in terms of a probability? It cannot literally mean to double the probability value (e.g. 50% becomes 100%, 75% becomes 150%, etc.). Such a model can also be represented as transformation to a linear regression

Thus, the **generalized linear model (GLM)** is a flexible generalization of ordinary linear regression that allows for response variables that have error distribution models other than a normal distribution. The GLM generalizes linear regression by allowing the linear model to be related to the response variable via a link function and by allowing the magnitude of the variance of each measurement to be a function of its predicted value. GLMs thus unify linear regression with other models including Logistic regression and Poisson regression.

Generalized linear models cover all these situations by

- a) Allowing for response variables that have arbitrary distributions (rather than simply normal distributions).
- b) Using an arbitrary function of the response variable (the link function) such that the output of the link function varies linearly with the predicted values (rather than assuming that the response itself must vary linearly).

Non-linear Regression and Non-linear classification

To elaborate, when we say that linear regression requires a linear model, we mean that each term is either a constant or the product of a parameter and a predictor i.e. of the form:

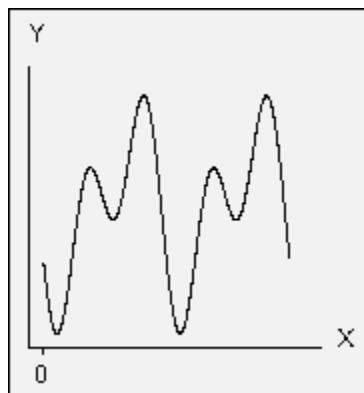
Response = constant + parameter * predictor + ... + parameter * predictor

$$Y = b_0 + b_1 X_1 + b_2 X_2 + \dots + b_k X_k$$

For an equation to be linear, the parameters have to be linear.

The simplest way to understand **non-linear** equations is to consider equations that do not follow the format above. For example, A Fourier equation:

Fourier: $\Theta_1 * \cos(X + \Theta_4) + (\Theta_2 * \cos(2*X + \Theta_4) + \Theta_3$ with the graph of form,



is a nonlinear equation because the equation is not of the form $Y = b_0 + b_1X_1 + b_2X_2 + \dots + b_kX_k$.

Thus, while a linear equation has a single basic format, nonlinear equations can take many different forms. This provides both power and complexity. Note also that as we discussed before, some nonlinear regression problems can be moved to a linear domain by a suitable transformation i.e. through a link function.

Thus, for non-linear equations, we have two possibilities:

- a) We can either convert the non-linear equation to a linear equation and then apply linear regressions OR
- b) We can consider a non-linear equation which best models the problem.

In general, you should try linear regression first because it is both easier to use and easier to interpret. However, if you simply are not able to get a good fit with linear regression, then you should consider nonlinear regression. Because nonlinear regression allows a nearly infinite number of possible functions, it can be more difficult to setup. Potential nonlinear shapes include concave, convex, exponential growth or decay, sigmoidal (S), and asymptotic curves.

Both linear and nonlinear regression models minimize the sum of squares of the residual error (SSE) to estimate the parameters. However, for nonlinear regression, there is no direct solution for minimizing the sum of squares of the residual error. The data is fitted using other means, such as successive approximations. For each iteration, the algorithm adjusts the parameter estimates so as to minimize the error compared to the previous iteration. Different algorithms use different approaches to determine the adjustments at each iteration. The iterations continue until the algorithm converges on the minimum sum of squares of the residual error or the maximum number of iterations are reached. A similar approach (iterative operations) also applies to non-linear classification.²¹

Additional Concepts

Here, we discuss some additional concepts critical to understanding typical Data Science programs.

Model Fitting or Curve Fitting

The process of model fitting involves finding the curve(represented by a function) which best fits the data. This involves three steps: First, you need a function that takes in a set of parameters and returns a predicted data set. Second, you need an ‘error function’ that provides a number representing the difference between your data and the model’s prediction for any given set of model parameters. Third, you need to find the parameters that minimize this difference.

Cross Validation

Conventional validation involves splitting the data into two sets of 70% for test and 30% for training. One of the main reasons for using cross-validation instead of using the conventional validation (e.g. partitioning the data set into two sets of 70% for training and 30% for test) is that there is not enough data available to partition it into separate training and test sets without losing significant modelling or testing capability. In k-fold cross-validation, the original sample is randomly partitioned into k equal sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining $k - 1$ subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results from the folds can then be averaged to produce a single estimation.²²

Algorithms

In this section, we discuss implementations of some algorithms. Considering our ‘lens’ of linear regression, we consider the following algorithms. We also consider one example of each. Thus, this is a small subset of the possibilities. However, for early stage learning, it provides a concise roadmap.

- Linear regression (ordinary least squares regression)
- Linear classification (logistic regression.)
- Non-linear regression(k-nearest neighbour)
- Non-linear classification (neural network)
- Clustering

Linear Regression

Ordinary Least Squares (OLS) regression is a linear model that seeks to find a set of coefficients for a line/hyperplane that minimizes the sum of the squared errors. The following code demonstrates linear regression on the Longley dataset. Source²³

```
# load data
data ( longley)
# fit model
fit <- lm ( Employed ~., longley)
# summarize the fit
summary ( fit)
# make predictions
predictions <- predict ( fit , longley)
# summarize accuracy
mse <- mean (( longley$Employed - predictions )^ 2)
print ( mse)
```

Notes:

- longley[] – shows all Rows and columns
- longley [, 1 : 6] – shows all rows and the first 6 columns
- longley[7] – shows the seventh column

- summary(longley) shows a summary
- # fit model fit <- lm (Employed ~., longley)
- In R, the lm(), or “linear model,” function can be used to create a simple regression model.²⁴
- formula: describes the model. formula argument follows a specific format. For simple linear regression, this is “YVAR ~ XVAR” where YVAR is the dependent, or predicted, variable and XVAR is the independent, or predictor, variable.²⁵
- On the right of <- is a formula object. It is often used to denote a statistical model, where the entity on the left of the ~ is the response and the things on the right of the ~ are the explanatory variables. So in English you’d say something like “Species depends on Sepal Length, Sepal Width, Petal Length and Petal Width” for Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width
- The output is as follows and can be interpreted as Employed = Intercept + parameter * feature .. etc as shown

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -3.482e+03 8.904e+02 -3.911 0.003560 ***
GNP.deflator 1.506e-02 8.492e-02  0.177 0.863141    
GNP          -3.582e-02 3.349e-02 -1.070 0.312681    
Unemployed   -2.020e-02 4.884e-03 -4.136 0.002535 ***
Armed.Forces -1.033e-02 2.143e-03 -4.822 0.000944 ***
Population    -5.110e-02 2.261e-01 -0.226 0.826212    
Year          1.829e+00 4.555e-01  4.016 0.003037 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Employed = (Intercept) + 1.506e-02 * GNP.deflator +
           -3.582e-02 * GNP and so on ..
```

Linear Classification : Logistic Regression

Logistic regression is an example of linear classification. It is a classification method that models the probability of an observation belonging to one of two classes. Below is an example of using logistic regression for multinomial classification. *Source*²⁶

```

# load the package
library(VGAM)
# load data
data(iris)
# fit model
fit <- vglm(Species~., family=multinomial, data=iris)
# summarize the fit
summary(fit)
# make predictions
probabilities <- predict(fit, iris[,1:4], type="response")
predictions <- apply(probabilities, 1, which.max)
predictions[which(predictions=="1")] <- levels(iris$Species)[1]
predictions[which(predictions=="2")] <- levels(iris$Species)[2]
predictions[which(predictions=="3")] <- levels(iris$Species)[3]
# summarize accuracy
table(predictions, iris$Species)

```

Notes:

- The program uses the Iris dataset. Iris is perhaps the best known database to be found in the pattern recognition literature. The data set contains three classes of 50 instances each, where each class refers to a type of iris plant. Predicted attribute is the class of iris plant.²⁷
- Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Examples include: Do body weight calorie intake, fat intake, and age have an influence on heart attacks (yes vs. no)?
- The output is for the format

predictions setosa versicolor virginica

setosa	50	0	0
versicolor	0	47	2
virginica	0	3	48

We interpret this as: 47 versicolor flowers were classified correctly and three were classified incorrectly (as virginica).

Non-Linear Regression – k-Nearest Neighbour

Having considered two linear methods (regression and classification), we now consider two non-linear methods (non-linear regression and non-linear classification). As we have discussed before, the methods to fit these equations are not the same as their Linear counterparts because they involve incremental operations such as successive approximations.

KNN(K-nearest neighbor) is an example of a non-linear regression algorithm. KNN is a non-parametric lazy learning algorithm. ‘**Non parametric**’ means that it does not make any assumptions on the underlying data distribution. This is useful because real world data does not obey the typical theoretical assumptions made by algorithms (e.g. Gaussian distribution, linearly separable etc). A **lazy algorithm** means that the algorithm does not use the training data because there is no explicit training phase (in practice, the training phase is minimal or is fast).

KNN assumes that the data is in a feature space with **Euclidean distance** being the common measure. The number ‘k’ in k-nearest decides how many neighbors (based on the distance metric) influence the classification. For each data point, the algorithm finds the k closest observations, and then classifies the data point to the majority. For example, if k = 3, and the three nearest observations to a specific data point belong to the classes A, B, and A respectively, the algorithm will classify the data point into class A. Applications of KNN include content retrieval, computer vision, gene expression, protein-protein interaction, and 3D structure prediction.

The following code demonstrates kNN for regression on the longley dataset.²⁹

```

# load the package
library ( caret)
# load data
data ( longley)
# fit model
fit <- knnreg ( longley [ , 1 : 6 ] , longley [ , 7 ] , k = 3)
# summarize the fit
summary ( fit)
# make predictions
predictions <- predict ( fit , longley [ , 1 : 6 ])
# summarize accuracy
mse <- mean (( longley$Employed - predictions )^ 2)
print ( mse)

```

Non-Linear Classification – Neural Networks

A neural network is an example of a non-linear classification. A basic understanding of the terminology of neural networks is HERE³⁰ In subsequent versions of the tutorial, I will expand on this. A note: The inclusion of neural networks as supervised is strictly not true because it depends on the type of neural network we use. Autoencoders³¹ are unsupervised where it sets the target values to be equal to input values. While others CNN³² and RNN³³ are supervised as they need the target(Y) and inputs(X) to be specified explicitly. Acknowledgments³⁴

Unsupervised Learning: Clustering – k-Means Algorithm

So far, we have seen supervised algorithms. k-means (not to be confused with k-nearest) is an example of an unsupervised algorithm. K-means clustering

tries to cluster data based on similarity. We first specify the number of clusters we want the data to be grouped into. K-means does not predict an outcome. Rather, the algorithm just tries to find patterns in the data. First, the algorithm randomly assigns each observation to a cluster, and finds the centroid of each cluster.

Then, the algorithm iterates through two steps:

- Reassign data points to the cluster whose centroid is closest.
- Calculate new centroid of each cluster.

These two steps are repeated until the within-cluster variation cannot be reduced any further. The within-cluster variation is calculated as the sum of the Euclidean distance between the data points and their respective cluster centroids.³⁵

Which Maths and Stats Techniques do You Need for Data Science?

As we come to the end of this tutorial, let us look at the wider maths and stats techniques needed for Data Science. A knowledge of algorithms (maths and stats) is the main differentiator between traditional programming and analytics -based programming. Having said that, it helps to start with programming and approach the maths (initially) through APIs and libraries. I find that this technique works better because more people are familiar with programming than with maths. As we have seen above, there are various places in the Data Science ecosystem where maths and stats apply.

Here is a list of maths and stats techniques useful for Data Scientists:

- Linear Algebra (e.g. vector algebra, matrices, transformations, eigenvalues).
- Probability Theory.
- Optimization techniques (eg. gradient descent).
- Descriptive Statistics (eg. means, modes, standard deviations, variances, distributions).

engineering, ensemble strategies, and visualization (story telling) all involve maths and stats.

Future versions of this tutorial will elaborate on this. I will take the following approach

- a) Understand the step/stage in the Data Science process.
- b) Approach the problem from the programming perspective(Python/R APIs).
- c) Understand the underlying maths behind the process.

Conclusion

This tutorial draws on the content on the Web and adds three new elements:

- a) We explain concepts simply but in context. Many tutorials explain one specific aspect but do not show how it fits into the wider picture
- b) We approach Maths and Stats from the perspective/lens of linear regression. This approach has an advantage because linear regression is familiar to many from their high school days.
- c) Less is more – we discuss a small number of examples. You can always find more examples of an algorithm type on the Web.

Future versions of this tutorial will expand on Maths and Science in context of Programming

If you want to engage with me and are based in the UK, I teach at [Oxford University \(Data Science for Internet of Things\)](#). If you are based globally, I have a new online course on [Enterprise AI Applications](#)

Comments and feedback welcome at ajit.jaokar@futuretext.com

¹Adapted from Stack overflow – What is the difference between Supervised learning and Unsupervised learning <http://stackoverflow.com/questions/1832076/what-is-the-difference-between-supervised-learning-and-unsupervised-learning>

²<http://www.dataists.com/2010/09/a-taxonomy-of-data-science/>

³Adapted from <http://www.cs.cmu.edu/~tom/>

⁴<http://pages.cs.wisc.edu/~jerryzhu/pub/sslicml07.pdf>

⁵<http://stackoverflow.com/questions/1832076/what-is-the-difference-between-supervised-learning-and-unsupervised-learning>

⁶<https://www.analyticsvidhya.com/blog/2016/07/practical-guide-data-preprocessing-python-scikit-learn/>

⁷<http://scikit-learn.org/stable/modules/preprocessing.html>

⁸<http://www.itl.nist.gov/div898/handbook/eda/section1/edall.htm>

⁹<http://stats.stackexchange.com/questions/17565/choosing-the-best-model-from-among-different-best-models>

¹⁰Adapted from [https://en.wikipedia.org/wiki/Feature_\(machine_learning\)](https://en.wikipedia.org/wiki/Feature_(machine_learning))

¹¹<https://deeplearning4j.org/neuralnet-overview>

¹²<http://searchbusinessanalytics.techtarget.com/definition/Ensemble-modeling>

¹³Adapted from <http://Web.engr.oregonstate.edu/~tgd/publications/mcs-ensembles.pdf>

¹⁴<http://www.kdnuggets.com/2016/07/storytelling-power-influence-data-science.html>

¹⁵Image adapted from source <https://www.youtube.com/watch?v=LKh3UdA8Gcc>

¹⁶<http://aleph0.clarku.edu/~djoyce/java/elements/elements.html>

¹⁷<https://www.quora.com/I-read-that-a-line-is-a-special-curve-Is-that-true-If-so-can-anybody-explain>

¹⁸<http://www.themathpage.com/aprecalc/equation-of-a-line.htm>

¹⁹<http://onlinestatbook.com/2/regression/intro.html>

²⁰<https://www.mathsisfun.com/data/standard-normal-distribution.html>

²¹Adapted from minitab documents <http://blog.minitab.com/blog/adventures-in-statistics/what-is-the-difference-between-linear-and-nonlinear-equations-in-regression-analysis>

²²http://scikit-learn.org/stable/modules/cross_validation.html

²³Linear Regression in R – Machine Learning Mastery <http://machinelearningmastery.com/linear-regression-in-r/>

²⁴R Tutorial series on simple linear Regression – R bloggers <https://www.r-bloggers.com/r-tutorial-series-simple-linear-regression/>

²⁵Use of Tilde in R Programming – Stack overflow <http://stackoverflow.com/questions/14976331/use-of-tilde-in-r-programming-language>

²⁶Linear Classification in R Source: <http://machinelearningmastery.com/linear-classification-in-r/>

²⁷The IRIS Dataset <https://archive.ics.uci.edu/ml/datasets/Iris>

²⁸saravananthirumuruganathan – A detailed introduction to k nearest neighbor knn algorithm/ <https://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm/>

²⁹Non Linear regression in R – Machine Learning Mastery <http://machinelearningmastery.com/non-linear-regression-in-r/>

³⁰A Beginners Guide to Neural Networks - KD Nuggets <http://www.kdnuggets.com/2016/08/beginners-guide-neural-networks-r.html>

³¹Autoencoders <https://en.wikipedia.org/wiki/Autoencoder>

³²Convolutional Neural Networks https://en.wikipedia.org/wiki/Convolutional_neural_network

³³Recurrent Neural Networks https://en.wikipedia.org/wiki/Recurrent_neural_network

³⁴Sibanjan Das - <https://www.linkedin.com/in/sibanjan>

³⁵R bloggers – kmeans clustering in R - <https://www.r-bloggers.com/k-means-clustering-in-r/>