In this chapter we introduce an algorithm that is admirably suited for discovering the link between features or cues and some particular outcome: Logistic Regression. Indeed, logistic regression is one of the most important analytic tools in the social and natural sciences. In natural language processing, logistic regression is the baseline supervised machine learning algorithm for classification, and also has a very close relationship with neural networks.

What is a Classification Algorithm?

The idea of *Classification Algorithms* is pretty simple*.* You predict the target class by analyzing the training dataset. This is one of the most, if not *the most* essential concept you study when you *learn Data Science*.

Basic Terminology in Classification Algorithms

- Classifier: An algorithm that maps the input data to a specific category.
- Classification model: A classification model tries to draw some conclusion from the input values given for training. It will predict the class labels/categories for the new data.
- Feature: A feature is an individual measurable property of a phenomenon being observed.
- Binary Classification: Classification task with two possible outcomes. Eg: Gender classification (Male / Female)
- Multi-class classification: Classification with more than two classes. In multi-class classification, each sample is assigned to one and only one target label. Eg: An animal can be a cat or dog but not both at the same time.
- Multi-label classification: Classification task where each sample is mapped to a set of target labels (more than one class). Eg: A news article can be about sports, a person, and location at the same time.

Applications of Classification Algorithms

- Email spam classification
- Bank customers loan pay willingness prediction.
- Cancer tumor cells identification.
- Sentiment analysis
- Drugs classification
- Facial key points detection
- Pedestrians detection in an automotive car driving.

Types of Classification Algorithms

Classification Algorithms could be broadly classified as the following:

- *Linear Classifiers*
    - Logistic regression
    - Naive Bayes classifier
    - Fisher's linear discriminant
- *Support vector machines*
    - Least squares support vector machines
- *Quadratic classifiers*
- *Kernel estimation*
    - k-nearest neighbour
- *Decision trees*
    - Random forests
- *Neural networks*
- *Learning vector quantization*

What is Logistic Regression?

As confusing as the name might be, you can rest assured. Logistic Regression is a classification and not a regression algorithm. It estimates discrete values (Binary values like 0/1, yes/no, true/false) based on a given set of independent variable(s). Simply put, it basically, predicts the probability of occurrence of an event by fitting data to a *logit function*. Hence, it is also known as *logit regression*. The values obtained would always lie within 0 and 1 since it predicts the probability.

Let us try and understand this through another example.

Let's say there's a sum on your math test. It can only have 2 outcomes, right? Either you solve it or you don't (and let's not assume points for method here). Now imagine, that you are being given a wide range of sums in an attempt to understand which chapters have you understood well. The outcome of this study would be something like this – if you are given a

trigonometry based problem, you are 70% likely to solve it. On the other hand, if it is an arithmetic problem, the probability of you getting an answer is only 30%. This is what Logistic Regression provides you.

If I had to do the math, I would model the log odds of the outcome as a linear combination of the predictor variables.

Logistic Regression Model

ln[p/(1-p)] = β_0 **+** β_1**x**

p is the probability that the event Y occurs, p(Y=1)
[range=0 to 1]

p/(1-p) is the "odds ratio" [range=0 to ∞]

ln[p/(1-p)]: log odds ratio, or "logit"
[range=-∞ to +∞]

We have:

f(x) = 1 / (1 + $e^{-w.x}$) and we will interpret it as f(x) = P(y=1 | x) (in short p)

Thus we have:

P(y=1 | x) = f(x)

P(y=0 | x) = 1 − f(x)

Which can be written more compactly by unifying the two rules :

P(y | x) = $(f(x))^y(1-f(x))^{1-y}$ where y ∈ {0, 1}