**Logistic Regression:**

* It is used for binary classification problems (problems with two class values).
* The [logistic function](https://en.wikipedia.org/wiki/Logistic_function), also called the sigmoid function
* It’s an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

1 / (1 + e^-value)

* Input values (x) are combined linearly using weights or coefficient values (m/ Beta) to predict an output value (y). A key difference from linear regression is that the output value being modeled is a binary values (0 or 1) rather than a numeric value.

y = e^(b0 + b1\*x) / (1 + e^(b0 + b1\*x))

* Where y is the predicted output, b0 is the bias or intercept term and b1 is the coefficient for the single input value (x). Each column in your input data has an associated b coefficient (a constant real value) that must be learned from your training data.
* the logit is the inverse of the standard logistic function

## Logistic Regression Predicts Probabilities (Technical Interlude)

* Logistic regression models the probability of the default class (e.g. the first class).
* For example, if we are modeling people’s sex as male or female from their height, then the first class could be male and the logistic regression model could be written as the probability of male given a person’s height, or more formally:
* Logistic regression is a linear method, but the predictions are transformed using the logistic function. But predictions are not the linear combination of input like linear regression

p(X) = e^(b0 + b1\*X) / (1 + e^(b0 + b1\*X))

ln(p(X) / 1 – p(X)) = b0 + b1 \* X (ln🡪 natural log for remove e)

* This is useful because we can see that the calculation of the output on the right is linear again (just like linear regression), and the input on the left is a log of the probability of the default class.

ln(odds) = b0 + b1 \* X

* Because the odds are log transformed, we call this left hand side the **log-odds** or the **probit.**
* We can rewrite above function

odds = e^(b0 + b1 \* X)

* [Maximum-likelihood estimation](https://en.wikipedia.org/wiki/Maximum_likelihood) is a common learning algorithm used by a variety of machine learning algorithms for estimation of coefficient values. although it does make assumptions about the distribution of your data
* The best coefficients would result in a model that would predict a value very close to 1 (e.g. male) for the default class and a value very close to 0 (e.g. female) for the other class. The intuition for maximum-likelihood for logistic regression is that a search procedure seeks values for the coefficients (Beta values) that minimize the error in the probabilities predicted by the model to those in the data (e.g. probability of 1 if the data is the primary class).
* minimization algorithm is used to optimize the best values for the coefficients for your training data.

## Making Predictions with Logistic Regression

* We have learned the coefficients of b0 = -100 and b1 = 0.6. Using the equation above we can calculate the probability of male given a height of 150cm or more formally P(male|height=150). We will use EXP() for e, because that is what you can use if you type this example into your spreadsheet:

y = e^(b0 + b1\*X) / (1 + e^(b0 + b1\*X))

y = exp(-100 + 0.6\*150) / (1 + EXP(-100 + 0.6\*X))

y = 0.0000453978687

Or a probability of near zero that the person is a male.

* In practice we can use the probabilities directly. Because this is classification and we want a crisp answer, we can snap the probabilities to a binary class value, for example:
* 0 if p(male) < 0.5

1 if p(male) >= 0.5

## Prepare Data for Logistic Regression

* Binary Output Variable: This might be obvious as we have already mentioned it, but logistic regression is intended for binary (two-class) classification problems. It will predict the probability of an instance belonging to the default class, which can be snapped into a 0 or 1 classification.
* Remove Noise: Logistic regression assumes no error in the output variable (y), consider removing outliers and possibly misclassified instances from your training data.
* Gaussian Distribution: Logistic regression is a linear algorithm (with a non-linear transform on output). It does assume a linear relationship between the input variables with the output. Data transforms of your input variables that better expose this linear relationship can result in a more accurate model. For example, you can use log, root, Box-Cox and other univariate transforms to better expose this relationship.
* Remove Correlated Inputs: Like linear regression, the model can overfit if you have multiple highly-correlated inputs. Consider calculating the pairwise correlations between all inputs and removing highly correlated inputs.
* Fail to Converge: It is possible for the expected likelihood estimation process that learns the coefficients to fail to converge. This can happen if there are many highly correlated inputs in your data or the data is very sparse (e.g. lots of zeros in your input data).