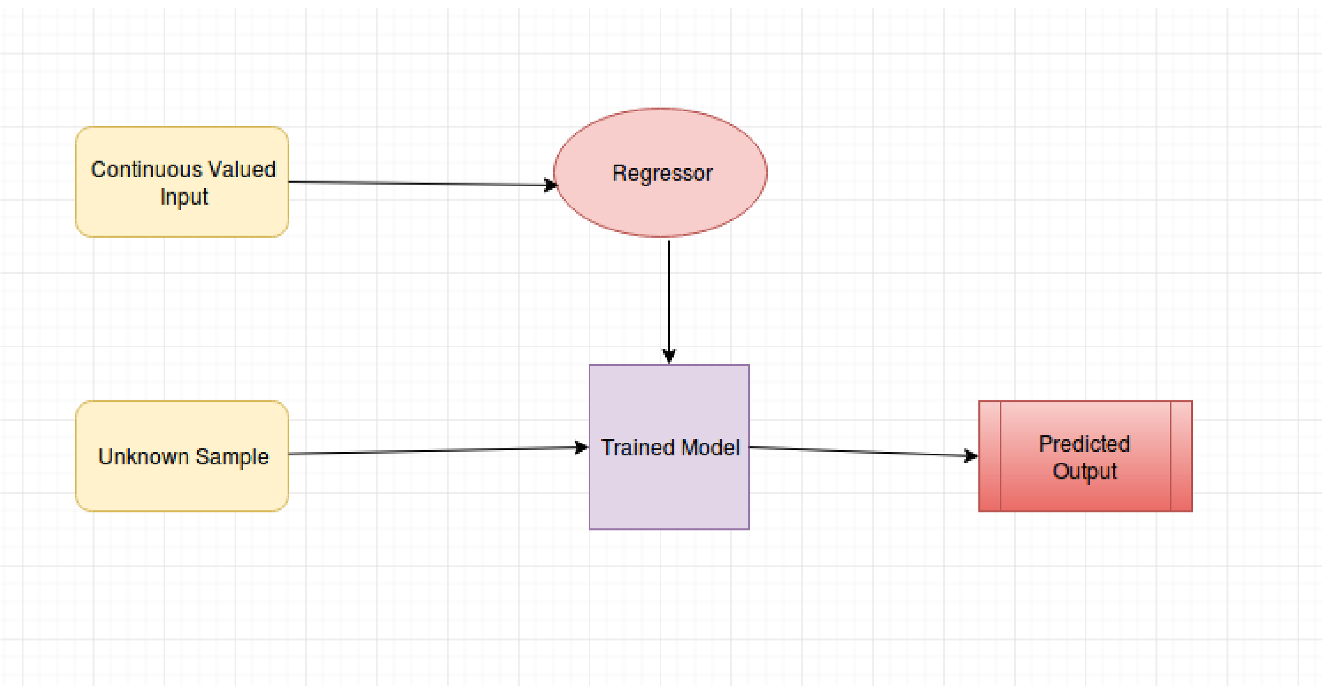
Linear Regression

Linear regression is a basic predictive analytics technique that uses historical data to predict an output variable. It is a method used to define a relationship between a dependent variable **(Y)** and independent variable **(X)** that is, between target and one or more predictors. Which is simply written as:

$$Y = mX + c$$

Where Y is the dependent variable, m is the scale factor or coefficient, c being the bias coefficient and X being the independent variable. The bias coefficient gives an extra degree of freedom to this model. If we are able to determine the optimum values of these two parameters, then we will have the line of best fit that we can use to predict the values of Y, given the value of X.



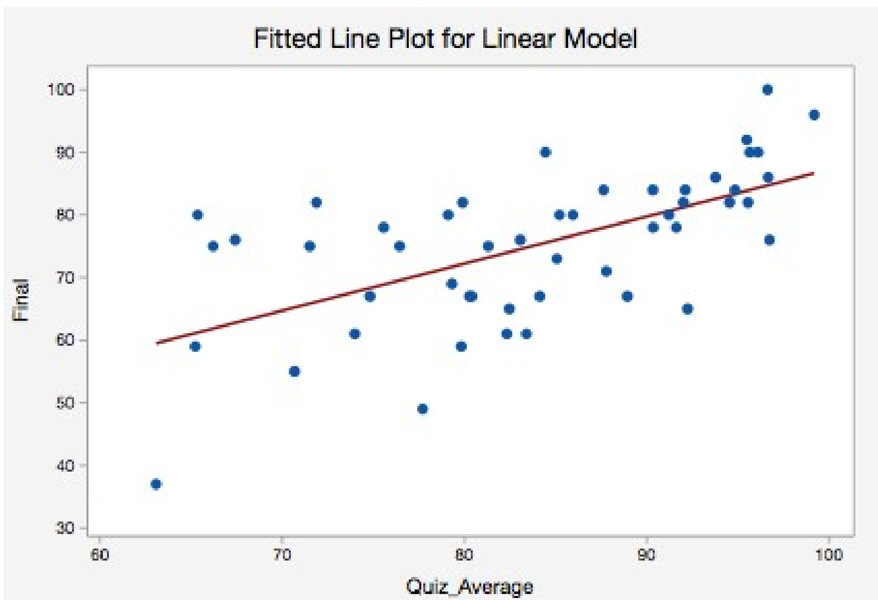
But how do we find these coefficients? There are many methods to do so, the most common being Least Mean Square Method approach and Gradient descent approach.

Uses:

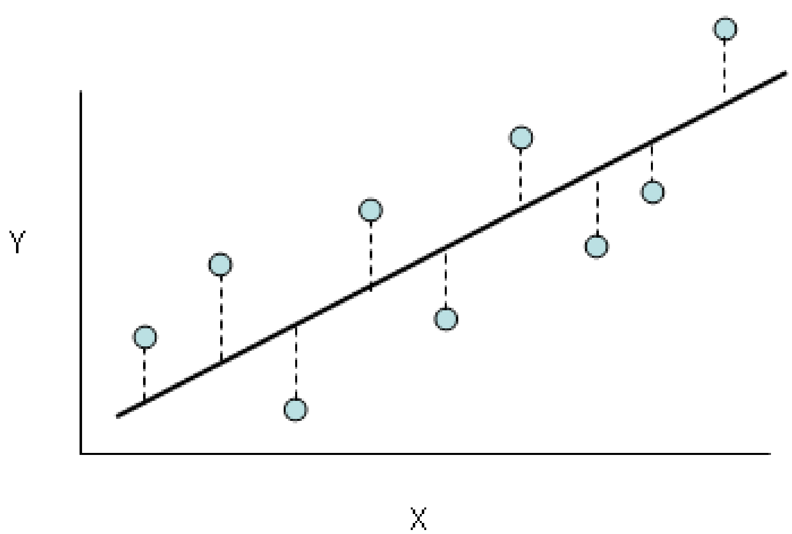
Linear regression models have many real-world applications in an array of industries such as economics (e.g. predicting growth), business (e.g. predicting product sales, employee performance), social science (e.g. predicting political leanings from gender or race), healthcare (e.g. predicting blood pressure levels from weight, disease onset from biological factors), and more.

Ordinary Least Mean Square:

For example, we get sample inputs and outputs and we plot these scatter point on a 2d graph, we something similar to the graph below:



Let's say that the line seen here is the actual relationship we accomplish from our model, and we want to minimize the error of our model. This line is the best fit that passes through most of the scatter points and also reduces error which is the distance from the point to the line itself as illustrated below.



And the total error of the linear model is the sum of the error of each point, i.e. ,

$$\sum_{i=1}^{n} r_i^{2}$$

where, **ri** = Distance between the line and ith point and **n** = total number of points.
The error is squared so as to obtain a positive value, as some points are above and some below the best fit line.
Now, we need to able ot measure how good our model is, that is, how accurate it is. We use the root mean squared error method.
Root Mean Squared Error is the square root of the sum of all errors divided by the number of values, or mathematically:

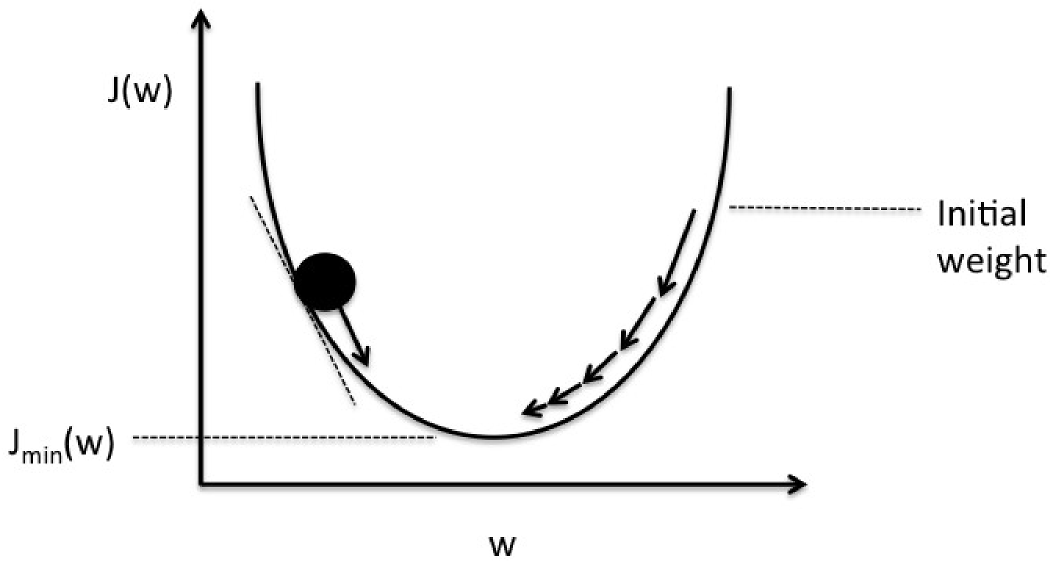
$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

Where **yj^** is the ith predicted output values.

Gradient Descent Algorithm:

Gradient descent is an iterative optimization algorithm to find the minimum of a function. Here that function is our loss Function or the error function that we just saw.
Imagine a valley and a person with no sense of direction who wants to get to the bottom of the valley. He goes down the slope and takes large steps when the slope is steep and small steps when the slope is less steep. He decides his

next position based on his current position and stops when he gets to the bottom of the valley which was his goal.



Schematic of gradient descent.

Let's try applying gradient descent to m and c and approach it step by step:

1) Initially let m = 0 and c = 0. Let L be our learning rate. This controls how much the value of m changes with each step. L could be a small value like 0.0001 for good accuracy.

2) Calculate the partial derivative of the loss function with respect to m, and plug in the current values of x, y, m and c in it to obtain the derivative value D.

$$D_m = \frac{1}{n} \sum_{i=0}^{n} 2(y_i - (mx_i + c))(-x_i)$$

$$D_m = \frac{-2}{n} \sum_{i=0}^{n} x_i(y_i - \bar{y}_i)$$

D_m is the value of the partial derivative with respect to m. Similarly lets find the partial derivative with respect to c, Dc:

3) Now we update the current value of m and c using the following equation:

$$D_c = \frac{-2}{n} \sum_{i=0}^{n} (y_i - \bar{y}_i)$$

$$m = m - L \times D_m$$

$$c = c - L \times D_c$$

4) We repeat this process until our loss function is a very small value or ideally 0 (which means 0 100% accuracy). The value of m and c that we are left with now will be the optimum values.

Gradient descent is one of the simplest and widely used algorithms in machine learning, mainly because it can be applied to any function to optimize it.

Pros of using linear regression:

- Space complexity is very low it just needs to save the weights at the end of training. hence it's a high latency algorithm.
- It is very simple to understand.
- Good interpretability.
- Feature importance is generated at the time model building. With the help of hyperparameter lambda, you can handle features selection hence we can achieve dimensionality reduction

Cons of linear regression:

- The algorithm assumes data is normally distributed in real they are not.

- Before building model multicollinearity should be avoided.
- It is prone to outliers.