***Linear Regression from Scratch Objective***

The main objective of the Linear Regression from Scratch is to predict the value of a variable

based on the value of another variable. The variable we want to predict is called the

dependent variable.

The variable we are using to predict the dependent variable’s value is called the

independent variable.

The simplest form of the regression equation with one dependent and one

independent variable.

y = m \* x + b

where,

• y = estimated dependent value.

• b = constant or bias.

• m = regression coefficient or slope.

• x = value of the independent variable.

Linear Regression from Scratch

In this article, we will implement the Linear Regression from scratch using only

Numpy.

1. Understanding Loss Function

While there are many loss functions to implement, we will use the Mean Squared Error function as our loss function.

A mean squared error function as the name suggests is the mean of squared sum of difference between true and predicted value.

As the predicted value of y depends on the slope and constant, hence our goal is to find the values for slope and constant that minimize the loss function or in other words, minimize the difference between y predicted and true values.

2. Optimization Algorithm

Optimization algorithms are used to find the optimal set of parameters given a training dataset that minimizes the loss function, in our case we need to find the optimal value of slope (m) and constant (b).

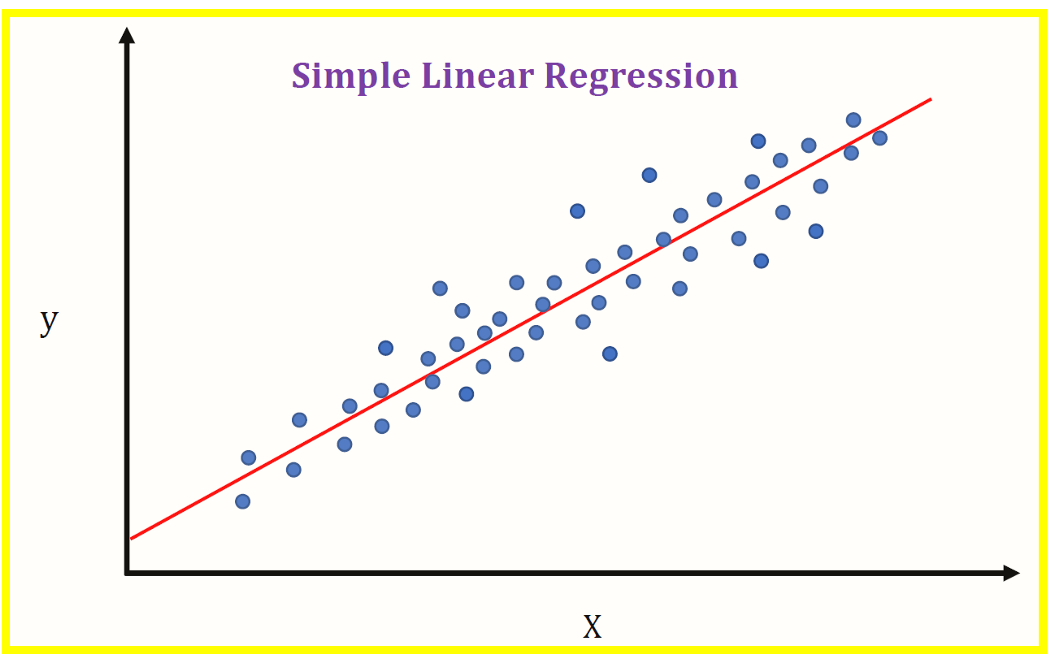
One such Algorithm is Gradient Descent.

Gradient descent is by far the most popular optimization algorithm used in machine learning.

Using gradient descent, we iteratively calculate the gradients of the loss function with respect to the parameters and keep on updating the parameters till we reach the local minima.

***LINEAR REGRESSION INTUITION***

Linear regression is a procedure used in statistics. As the term implies, it can only be used when there is a linear relationship among the variables, i.e when there is a straight-line relationship between two variables. The linear part indicates that we are using a linear approach in generalising over the data.



The types of relationships can a linear regression show are:

* Positive relationship

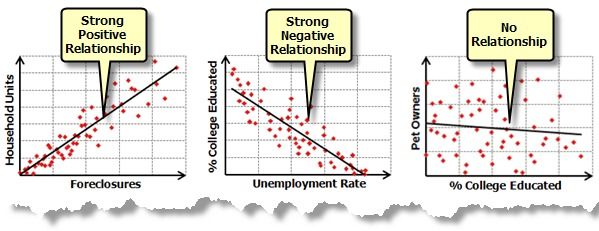
When the regression line between the two variables moves in the same direction with an upward slope, the variables are said to be in a positive relationship. This means that if the value of x (independent variable) is increased, there will be an increase in the dependent variable.

* Negative relationship

When the regression line between the two variables moves in the same direction with a downward slope, the variables are said to be in a negative relationship. This shows that if the value of the independent variable (x) is increased, there will be a decrease in the dependent variable (y).

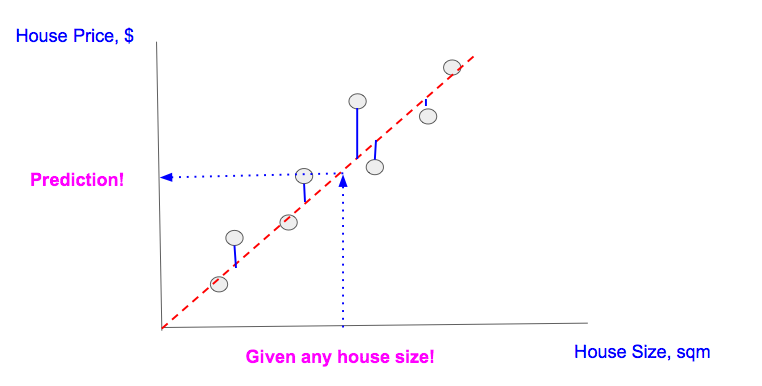
* No relationship

If the best fit line is flat (not sloped), it’s assumed that there is no relationship among the variables. There will be no change in the dependent variable (y) by increasing or decreasing the independent variable (x) value.



There are several use cases for linear regression. For instance, it can be used to predict the sales of a product, pricing, performance, or the salaries of prospected employees.

For example, suppose you want to predict the price of a house by knowing its size. You have the data which has some house prices and corresponding sizes. Charting the data and fitting a line among them will look something like this:



To generalize, you draw a straight line such that it crosses through the maximum points. Once you get that line, for house of any size you just project that data point over the line which gives you the house price. It’s used as a model for understanding the association between independent and dependent variables as well as to foresee the connection between two quantitative variables: predictor variables, which are known as independent variables, and dependent variables, which are those being predicted. For example, you want to predict the price of a house based on its Area, Garage Area, Land Contour, Utilities, etc. Here, "price" will be the dependent variable and "Area, Garage Area, Land Contour, Utilities" will be the independent variable.

## Loss functions for regression analyses:

A loss function measures how well a given machine learning model fits the specific data set. It boils down all the different under- and overestimations of the model to a single number, known as the prediction error. The bigger the difference between the prediction and the ground truth, the higher the value of the loss function. Loss functions are used automatically in the background hyperparameter optimization and when training the decision trees to compare the performance of various iterations of the model.

In the Elastic Stack, there are three different types of loss function:

* mean squared error (mse): It is the default choice when no additional information about the data set is available.
* mean squared logarithmic error (msle; a variation of mse): It is for cases where the target values are all positive with a long tail distribution (for example, prices or population).
* Pseudo-Huber Loss(huber): Use it when you want to prevent the model trying to fit the outliers instead of regular data.

The various types of loss function calculate the prediction error differently. The appropriate loss function for your use case depends on the target distribution in your data set, the problem that you want to model, the number of outliers in the data, and so on.

You can specify the loss function to be used during regression analysis when you create the data frame analytics job. The default is mean squared error (mse). If you choose msle or huber, you can also set up a parameter for the loss function. With the parameter, you can further refine the behavior of the chosen functions.

## The Cost Function of Linear Regression:

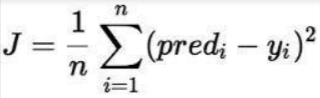
Cost function measures how a machine learning model performs.

Cost function is the calculation of the error between predicted values and actual values, represented as a single real number.

The difference between the cost function and loss function is as follows:

The cost function is the average error of n-samples in the data (for the whole training data) and the loss function is the error for individual data points (for one training example).

The cost function of a linear regression is root mean squared error or mean squared error. They are both the same; just we square it so that we don’t get negative values.



J=1/nsum(square(pred-y))J=1/nsum(square(pred –(mx+b))  
Y=mx +b

***Gradient Descent for Linear Regression***

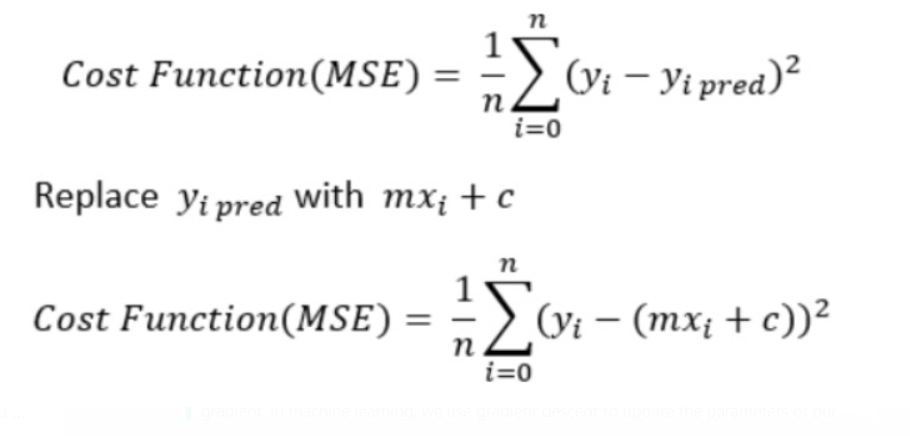
Gradient descent is an optimization algorithm used to minimize some function by iteratively moving in the direction of steepest descent as defined by the negative of the gradient. In machine learning, we use gradient descent to update the parameters of our model.

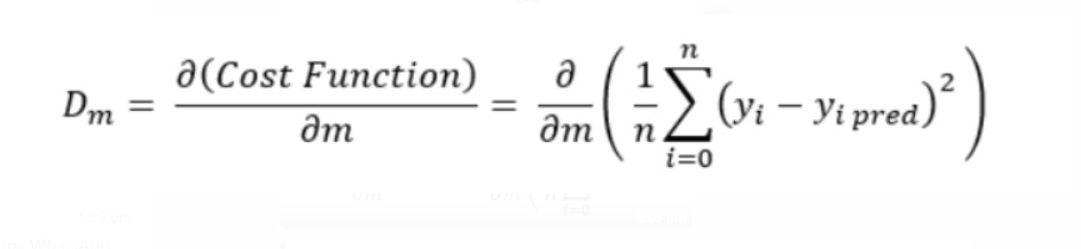
When there are more than one inputs you can use a process of optimizing values of coefficients by iteratively minimizing error of model on your training data. This is called Gradient Descent and works by starting with random values for each coefficient. The sum of squared errors are calculated for each pair of input and output variables.

A learning rate is used for each pair of input and output values. It is a scalar factor and coefficients are updated in direction towards minimizing error. The process is repeated until a minimum sum squared error is achieved or no further improvement is possible.

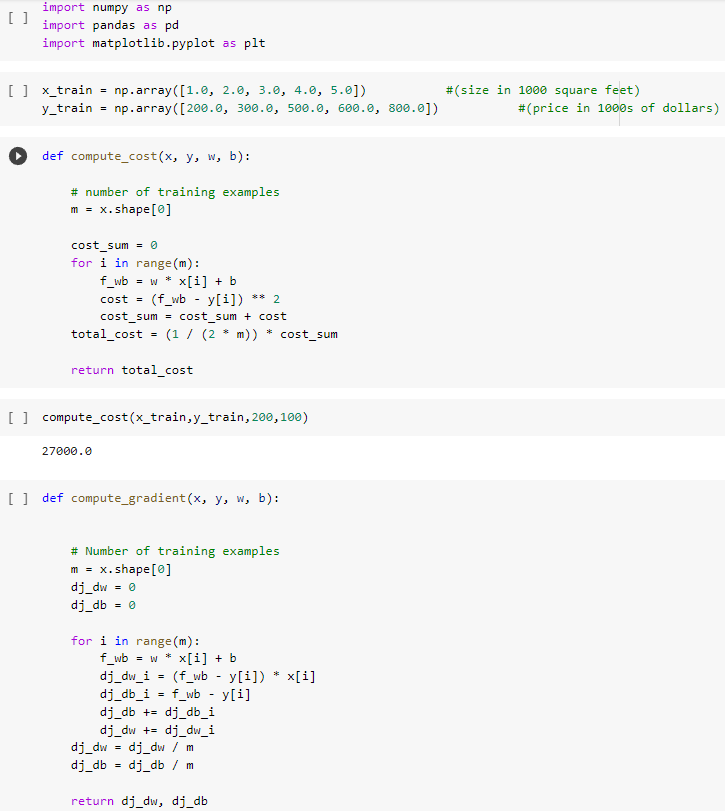
When using this method, learning rate alpha determines the size of improvement step to take on each iteration of procedure. In practise, Gradient Descent is useful when there is a large dataset either in number of rows or number of columns.

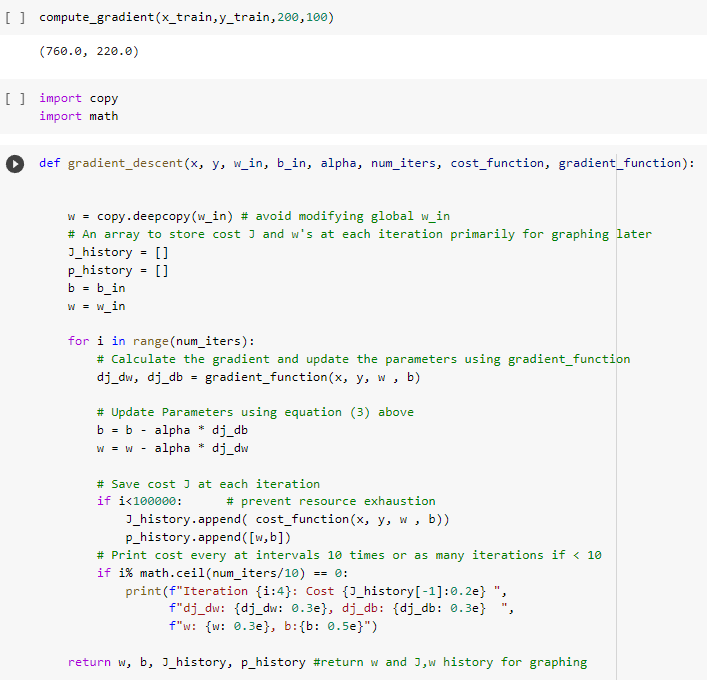
In short ,it is a minimization algorithm meant for minimizing a given activation function.

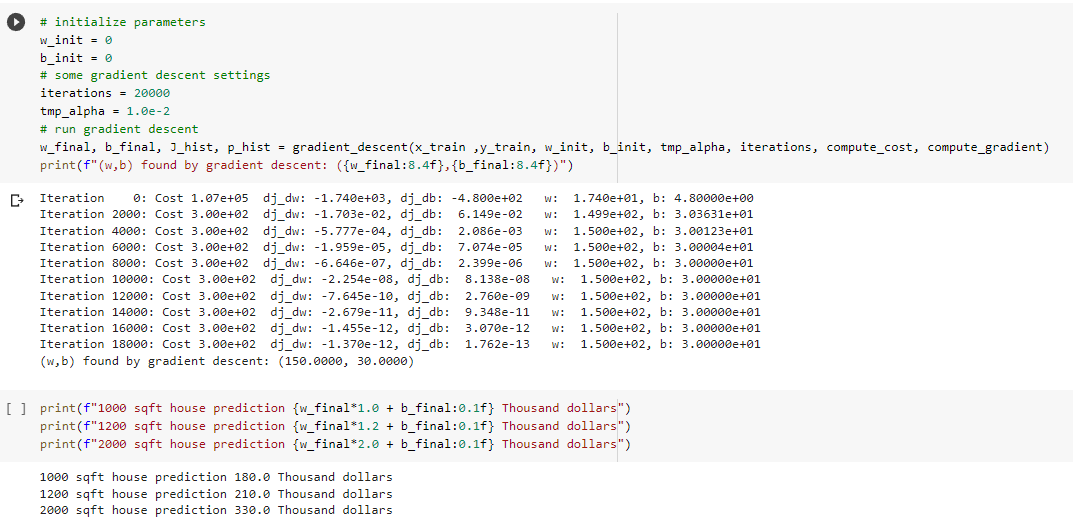


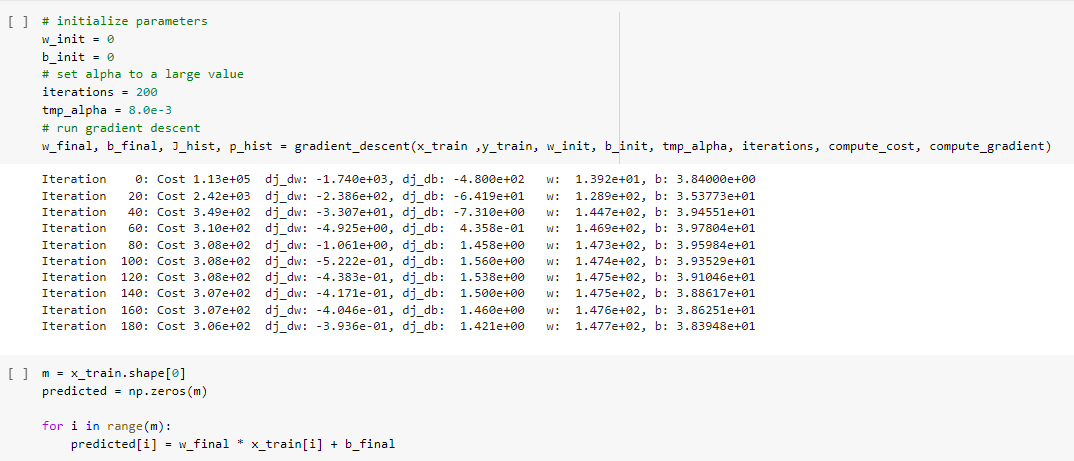


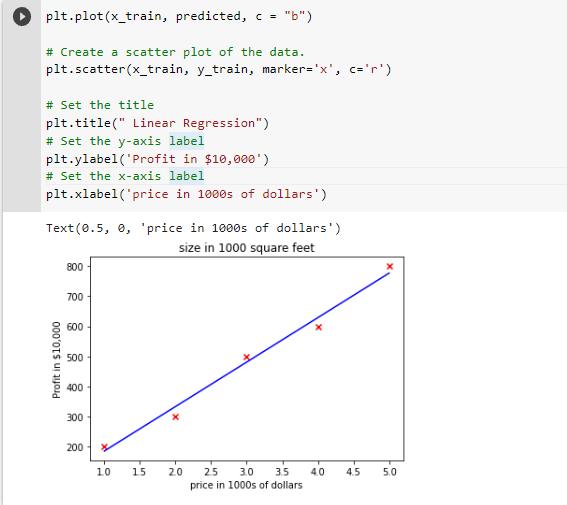
***CODE:***











***REFERENCE:***

<https://medium.com/@minions.k/gradient-descent-algo-d7b180605d7c>

<https://www.analyticsvidhya.com/blog/2021/04/gradient-descent-in-linear-regression/>