Machine Learning Algorithms

Machine Learning Algorithm can be broadly classified into three types:

1. **Supervised Learning Algorithms**
2. **Unsupervised Learning Algorithms**
3. **Reinforcement Learning algorithm**

# **Regression analysis in Machine Learning**

## What is regression?

Regression analysis is a statistical method to model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables. More specifically, Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed

## Terminologies Related to the Regression Analysis:

* **Dependent Variable:** The main factor in Regression analysis which we want to predict or understand is called the dependent variable. It is also called **target variable**.
* **Independent Variable:** The factors which affect the dependent variables or which are used to predict the values of the dependent variables are called independent variable, also called as a **predictor**.
* **Outliers:** Outlier is an observation which contains either very low value or very high value in comparison to other observed values. An outlier may hamper the result, so it should be avoided.
* **Multicollinearity:** If the independent variables are highly correlated with each other than other variables, then such condition is called Multicollinearity. It should not be present in the dataset, because it creates problem while ranking the most affecting variable.
* **Underfitting and Overfitting:** If our algorithm works well with the training dataset but not well with test dataset, then such problem is called **Overfitting**. And if our algorithm does not perform well even with training dataset, then such problem is called **underfitting**.

## Types of Regression

### Linear Regression

* Linear regression is a statistical regression method which is used for predictive analysis.
* It is one of the very simple and easy algorithms which works on regression and shows the relationship between the continuous variables.
* It is used for solving the regression problem in machine learning.
* Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), hence called linear regression.
* If there is only one input variable (x), then such linear regression is called **simple linear regression**. And if there is more than one input variable, then such linear regression is called **multiple linear regression**.

**Example**

Area of a plot and price of a plot below.

|  |  |  |
| --- | --- | --- |
| **area(x)** | **price(y)(Actual)** | **Predicted Y** |
| 2,600 | 5,50,000.00 | 5,33,670.00 |
| 3,000 | 5,65,000.00 | 5,87,986.00 |
| 3,200 | 6,10,000.00 | 6,15,144.00 |
| 3,600 | 6,80,000.00 | 6,69,460.00 |
| 4,000 | 7,25,000.00 | 7,23,776.00 |

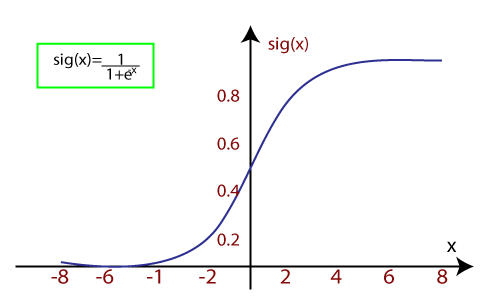
### Logistic Regression:

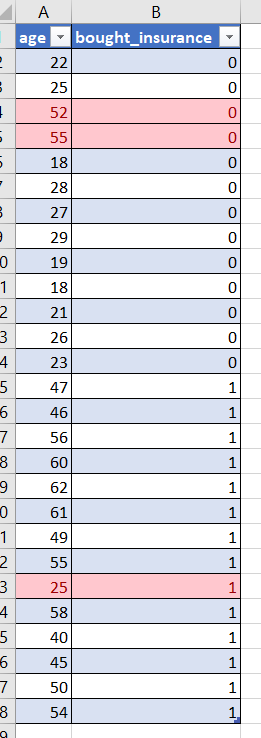
* Logistic regression is another supervised learning algorithm which is used to solve the classification problems. In **classification problems**, we have dependent variables in a binary or discrete format such as 0 or 1.
* Logistic regression algorithm works with the categorical variable such as 0 or 1, Yes or No, True or False, Spam or not spam, etc.
* It is a predictive analysis algorithm which works on the concept of probability.
* Logistic regression is a type of regression, but it is different from the linear regression algorithm in the term how they are used.
* Logistic regression uses **sigmoid function** or logistic function which is a complex cost function. This sigmoid function is used to model the data in logistic regression. The function can be represented as:

Regression Analysis in Machine learning

* f(x)= Output between the 0 and 1 value.
* x= input to the function
* e= base of natural logarithm.

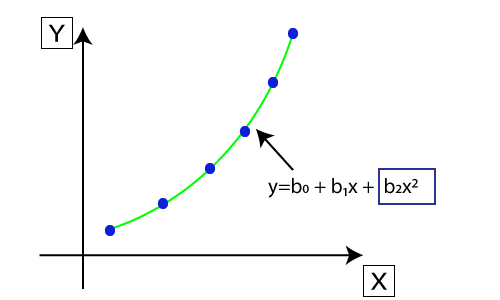
When we provide the input values (data) to the function, it gives the S-curve as follows:





### Polynomial Regression:

* Polynomial Regression is a type of regression which models the **non-linear dataset** using a linear model.
* It is similar to multiple linear regression, but it fits a non-linear curve between the value of x and corresponding conditional values of y.
* Suppose there is a dataset which consists of datapoints which are present in a non-linear fashion, so for such case, linear regression will not best fit to those datapoints. To cover such datapoints, we need Polynomial regression.
* I**n Polynomial regression, the original features are transformed into polynomial features of given degree and then modelled using a linear model.** Which means the datapoints are best fitted using a polynomial line.



* The equation for polynomial regression also derived from linear regression equation that means Linear regression equation Y= b0+ b1x, is transformed into Polynomial regression equation Y= b0+b1x+ b2x2+ b3x3+.....+ bnxn.

# **Gradient Descent**

## What is a gradient?

**A gradient** measure **how much the output of a function changes if you change the inputs a little bit.**

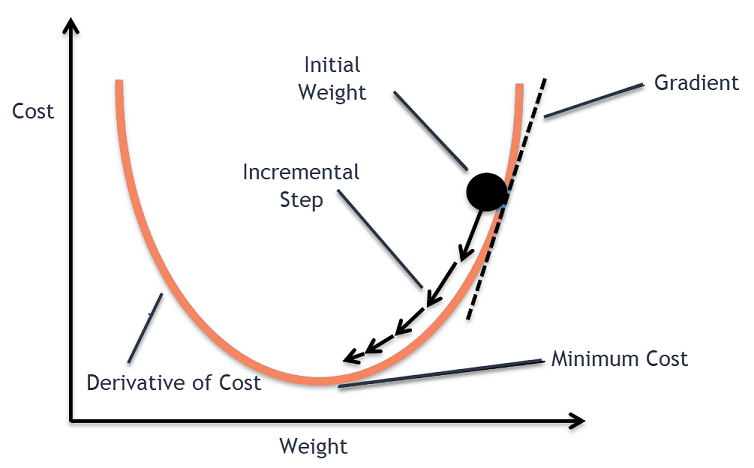
Before going deeper into GD, we need to understand what is a cost function?

## Cost Function

It is a **function** that measures the performance of a model for any given data. [Cost Function](https://www.analyticsvidhya.com/blog/2021/02/cost-function-is-no-rocket-science/) quantifies the error between predicted values and expected values and presents it in the form of a single real number.

Gradient descent is an optimization algorithm used in machine learning to minimize the cost function by iteratively adjusting parameters in the direction of the negative gradient, aiming to find the optimal set of parameters.

The cost function represents the discrepancy between the predicted output of the model and the actual output. The goal of gradient descent is to find the set of parameters that minimizes this discrepancy and improves the model’s performance.



## How Does Gradient Descent Work?

1. Gradient descent is an optimization algorithm used to minimize the cost function of a model.
2. The cost function measures how well the model fits the training data and is defined based on the difference between the predicted and actual values.
3. The gradient of the cost function is the derivative with respect to the model’s parameters and points in the direction of the steepest ascent.
   1. Here derivative means slope
4. The algorithm starts with an initial set of parameters and updates them in small steps to minimize the cost function.
5. In each iteration of the algorithm, the gradient of the cost function with respect to each parameter is computed.
6. The gradient tells us the direction of the steepest ascent, and by moving in the opposite direction, we can find the direction of the steepest descent.
7. The size of the step is controlled by the learning rate, which determines how quickly the algorithm moves towards the minimum.
8. The process is repeated until the cost function converges to a minimum, indicating that the model has reached the optimal set of parameters.
9. There are different variations of gradient descent, including batch gradient descent, stochastic gradient descent, and mini-batch gradient descent, each with its own advantages and limitations.
10. Efficient implementation of gradient descent is essential for achieving good performance in machine learning tasks. The choice of the learning rate and the number of iterations can significantly impact the performance of the algorithm.

### **Batch Gradient Descent**

Batch gradient descent updates the model’s parameters using the gradient of the entire training set. It calculates the average gradient of the cost function for all the training examples and updates the parameters in the opposite direction. Batch gradient descent guarantees convergence to the global minimum, but can be computationally expensive and slow for large datasets.

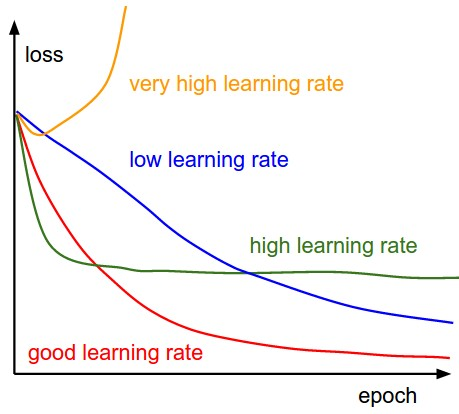
### **Stochastic Gradient Descent**

Stochastic gradient descent updates the model’s parameters using the gradient of one training example at a time. It randomly selects a training example, computes the gradient of the cost function for that example, and updates the parameters in the opposite direction. Stochastic gradient descent is computationally efficient and can converge faster than batch gradient descent.

SGD is preferred usually when there are clusters within the data set.

### **Mini-Batch Gradient Descent**

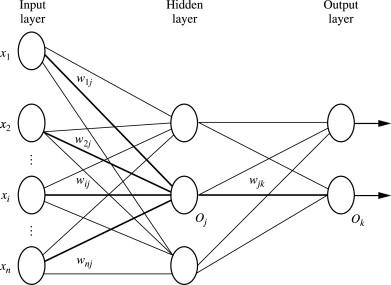
Mini-batch gradient descent updates the model’s parameters using the gradient of a small subset of the training set, known as a mini-batch. It calculates the average gradient of the cost function for the mini-batch and updates the parameters in the opposite direction.



# **Back propagation**

Backpropagation is an algorithm that backpropagates the errors from the output nodes to the input nodes. Therefore, it is simply referred to as the backward propagation of errors.

The backpropagation algorithm works by computing the gradient of the loss function with respect to each weight via the chain rule, computing the gradient layer by layer, and iterating backward from the last layer to avoid redundant computation of intermediate terms in the chain rule.



**Step 1:** Inputs X, arrive through the preconnected path.

**Step 2:** The input is modeled using true weights W. Weights are usually chosen randomly.

**Step 3:**Calculate the output of each neuron from the input layer to the hidden layer to the output layer.

**Step 4:** Calculate the error in the outputs

Backpropagation Error= Actual Output – Desired Output

**Step 5:** From the output layer, go back to the hidden layer to adjust the weights to reduce the error.

**Step 6:** Repeat the process until the desired output is achieved.

# **Loss function**

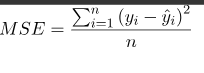
A loss function is a mathematical function that quantifies the difference between predicted and actual values in a machine learning model. It measures the model's performance and guides the optimization process by providing feedback on how well it fits the data

2 main classes of Loss Function:

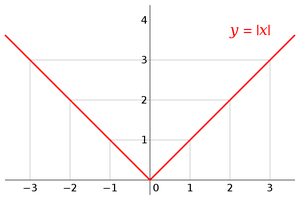
* Regression Models: predict continuous values.
* Classification Models: predict the output from a set of finite categorical values.

## **Regression Losses**

### **Mean Squared Error (MSE) | L2 Loss**

* It is the Mean of Square of Residuals for all the datapoints in the dataset. Residuals is the difference between the actual and the predicted prediction by the model.
* Squaring of residuals is done to convert negative values to positive values. The normal error can be both negative and positive. If some positive and negative numbers are summed up, the sum maybe 0.
* Squaring also gives more weightage to larger errors. When the cost function is far away from its minimal value, squaring the error will penalize the model more and thus helping in reaching the minimal value faster.

### **Mean absolute Error ( MAE)**

* It is the Mean of Absolute of Residuals for all the datapoints in the dataset. Residuals is the difference between the actual and the predicted prediction by the model.
* The absolute of residuals is done to convert negative values to positive values.
* Mean is taken to make the loss function independent of number of datapoints in the training set.
* One advantage of MAE is that is robust to outliers.

## **Classification Losses**

### **Binary classification**

* Binary Classification is a problem where we have to segregate our observations in any of the two labels on the basis of the features
* Suppose you have some images now you have to put each of them in a stack one for Dogs and the other for the Cats. Here you are solving a binary classification problem.



**Loss Function = Actual value – Predicted value**

**Here comes the concept of entropy?**

Entropy is defined as the randomness or measuring the disorder of the information being processed in Machine Learning. Further, in other words, we can say that entropy is the machine learning metric that measures the unpredictability or impurity in the system

Higher entropy (randomness) 🡪 Higher loss 🡪 lower quality prediction

### **What is Binary Cross Entropy Or Logs Loss?**

Binary Cross Entropy is a loss function used in machine learning and deep learning to measure the difference between predicted binary outcomes and actual binary labels.

Also called as logs loss. Binary cross entropy compares each of the predicted probabilities to actual class output which can be either 0 or 1. It then calculates the score that penalizes the probabilities based on the distance from the expected value.

**Definition 🡪 Binary Cross Entropy is the negative average of the log of corrected predicted probabilities.**

