# Unit 2

1. Explain simple linear regression.

# Simple Linear Regression in Machine Learning

Simple Linear Regression is a type of Regression algorithms that models the relationship between a dependent variable and a single independent variable. The relationship shown by a Simple Linear Regression model is linear or a sloped straight line, hence it is called Simple Linear Regression.

The key point in Simple Linear Regression is that the ***dependent variable must be a continuous/real value***. However, the independent variable can be measured on continuous or categorical values.

Simple Linear regression algorithm has mainly two objectives:

* + - * **Model the relationship between the two variables.** Such as the relationship between Income and expenditure, experience and Salary, etc.
      * **Forecasting new observations.** Such as Weather forecasting according to temperature, Revenue of a company according to the investments in a year, etc.

## Simple Linear Regression Model:

The Simple Linear Regression model can be represented using the below equation:

y= a0+a1x+ ε

Where,

**a0= It is the intercept of the Regression line (can be obtained putting x=0)**

**a1= It is the slope of the regression line, which tells whether the line is increasing**

**or**

**decreasing. ε = The error term. (For a good model it will be negligible**

1. Explain gradient descent for simple linear regression.

# Gradient Descent in Machine Learning

Gradient Descent is known as one of the most commonly used optimization algorithms to train machine learning models by means of minimizing errors between actual and expected results. Further, gradient descent is also used to train Neural Networks.

In mathematical terminology, Optimization algorithm refers to the task of minimizing/maximizing an objective function f(x) parameterized by x. Similarly, in machine learning, optimization is the task of minimizing the cost function parameterized by the model's parameters. The main objective of gradient descent is to minimize the convex function using iteration of parameter updates. Once these machine learning models are optimized, these models can be used as powerful tools for Artificial Intelligence and various computer science applications.

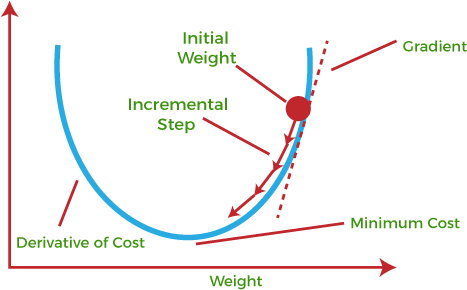
In this tutorial on Gradient Descent in Machine Learning, we will learn in detail about gradient descent, the role of cost functions specifically as a barometer within Machine Learning, types of gradient descents, learning rates, etc.

**What is Gradient Descent or** Steepest Descent

Gradient descent was initially discovered by **"Augustin-Louis Cauchy"** in mid of 18th century. ***Gradient Descent is defined as one of the most commonly used iterative optimization algorithms of machine learning to train the machine learning and deep learning models. It helps in finding the local minimum of a function.***

The best way to define the local minimum or local maximum of a function using gradient descent is as follows:

* If we move towards a negative gradient or away from the gradient of the function at the current point, it will give the **local minimum** of that function.
* Whenever we move towards a positive gradient or towards the gradient of the function at the current point, we will get the **local maximum** of that function.



This entire procedure is known as Gradient Ascent, which is also known as steepest descent. ***The main objective of using a gradient descent algorithm is to minimize the cost function using iteration.*** To achieve this goal, it performs two steps iteratively:

* Calculates the first-order derivative of the function to compute the gradient or slope of that function.
* Move away from the direction of the gradient, which means slope increased from the current point by alpha times, where Alpha is defined as Learning Rate. It is a tuning parameter in the optimization process which helps to decide the length of the steps.

#### What is Cost-function?

***The cost function is defined as the measurement of difference or error between actual values and expected values at the current position and present in the form of a single real number.*** It helps to increase and improve machine learning efficiency by providing feedback to this model so that it can minimize error and find the local or global minimum. Further, it continuously iterates along the direction of the negative gradient until the cost function approaches zero. At this steepest descent point, the model will stop learning further. Although cost function and loss function are considered synonymous, also there is a minor difference between them. The slight difference between the loss function and the cost function is about the error within the training of machine learning models, as loss function refers to the error of one training example, while a cost function calculates the average error across an entire training set.

The cost function is calculated after making a hypothesis with initial parameters and modifying these parameters using gradient descent algorithms over known data to reduce the cost function.

Hypothesis:

Parameters:

Cost function:

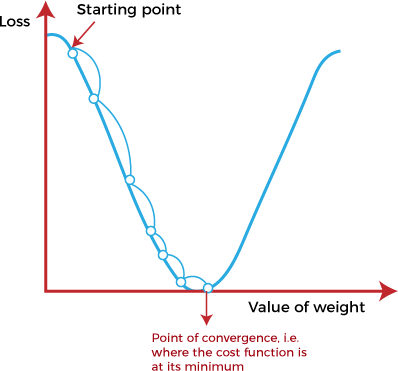
Goal:

#### How does Gradient Descent work?

Before starting the working principle of gradient descent, we should know some basic concepts to find out the slope of a line from linear regression. The equation for simple linear regression is given as:

1. Y=mX+c

Where 'm' represents the slope of the line, and 'c' represents the intercepts on the y- axis.



The starting point (shown in above fig.) is used to evaluate the performance as it is considered just as an arbitrary point. At this starting point, we will derive the first derivative or slope and then use a tangent line to calculate the steepness of this slope. Further, this slope will inform the updates to the parameters (weights and bias).

The slope becomes steeper at the starting point or arbitrary point, but whenever new parameters are generated, then steepness gradually reduces, and at the lowest point, it approaches the lowest point, which is called **a point of convergence.**

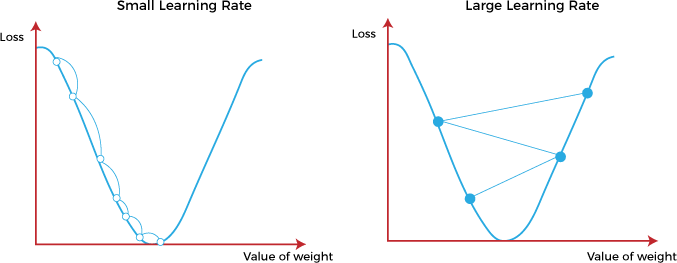
The main objective of gradient descent is to minimize the cost function or the error between expected and actual. To minimize the cost function, two data points are required:

o **Direction & Learning Rate**

These two factors are used to determine the partial derivative calculation of future iteration and allow it to the point of convergence or local minimum or global minimum. Let's discuss learning rate factors in brief;

#### Learning Rate:

It is defined as the step size taken to reach the minimum or lowest point. This is typically a small value that is evaluated and updated based on the behavior of the cost function. If the learning rate is high, it results in larger steps but also leads to risks of overshooting the minimum. At the same time, a low learning rate shows the small step sizes, which compromises overall efficiency but gives the advantage of more precision.



Types of Gradient Descent

Based on the error in various training models, the Gradient Descent learning algorithm can be divided into **Batch gradient descent, stochastic gradient descent, and mini- batch gradient descent.** Let's understand these different types of gradient descent:

#### Batch Gradient Descent:

Batch gradient descent (BGD) is used to find the error for each point in the training set and update the model after evaluating all training examples. This procedure is known as the training epoch. In simple words, it is a greedy approach where we have to sum over all examples for each update.

###### Advantages of Batch gradient descent:

* + - It produces less noise in comparison to other gradient descent.
    - It produces stable gradient descent convergence.
    - It is Computationally efficient as all resources are used for all training samples.

#### Stochastic gradient descent

Stochastic gradient descent (SGD) is a type of gradient descent that runs one training example per iteration. Or in other words, it processes a training epoch for each example within a dataset and updates each training example's parameters one at a time. As it requires only one training example at a time, hence it is easier to store in allocated memory. However, it shows some computational efficiency losses in comparison to batch gradient systems as it shows frequent updates that require more detail and speed. Further, due to frequent updates, it is also treated as a noisy gradient. However, sometimes it can be helpful in finding the global minimum and also escaping the local minimum.

###### Advantages of Stochastic gradient descent:

In Stochastic gradient descent (SGD), learning happens on every example, and it consists of a few advantages over other gradient descent.

* It is easier to allocate in desired memory.
* It is relatively fast to compute than batch gradient descent.
* It is more efficient for large datasets.

#### MiniBatch Gradient Descent:

Mini Batch gradient descent is the combination of both batch gradient descent and stochastic gradient descent. It divides the training datasets into small batch sizes then performs the updates on those batches separately. Splitting training datasets into smaller batches make a balance to maintain the computational efficiency of batch gradient descent and speed of stochastic gradient descent. Hence, we can achieve a special type of gradient descent with higher computational efficiency and less noisy gradient descent.

###### Advantages of Mini Batch gradient descent:

* + - It is easier to fit in allocated memory.
    - It is computationally efficient.
    - It produces stable gradient descent convergence.

1. What is hypothesis function for simple linear regression?

### ****Hypothesis function in Linear Regression****

As we have assumed earlier that our independent feature is the experience i.e X and the respective salary Y is the dependent variable. Let’s assume there is a linear relationship between X and Y then the salary can be predicted using:

𝑌^=𝜃1+𝜃2𝑋*Y*^=*θ*1​+*θ*2​*X*

OR

𝑦^𝑖=𝜃1+𝜃2𝑥𝑖*y*^​*i*​=*θ*1​+*θ*2​*xi*​

Here,

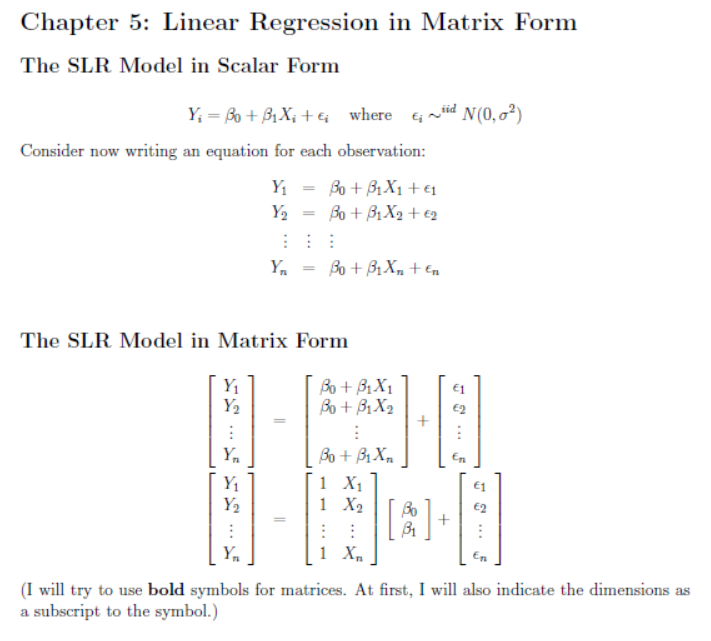
* 𝑦𝑖𝜖𝑌(𝑖=1,2,⋯,𝑛)     *yi*​*ϵY*(*i*=1,2,⋯,*n*)  are labels to data (Supervised learning)
* 𝑥𝑖𝜖𝑋(𝑖=1,2,⋯,𝑛)     *xi*​*ϵX*(*i*=1,2,⋯,*n*)  are the input independent training data (univariate – one input variable(parameter))
* 𝑦𝑖^𝜖𝑌^(𝑖=1,2,⋯,𝑛)     *yi*​^​*ϵY*^(*i*=1,2,⋯,*n*)  are the predicted values.

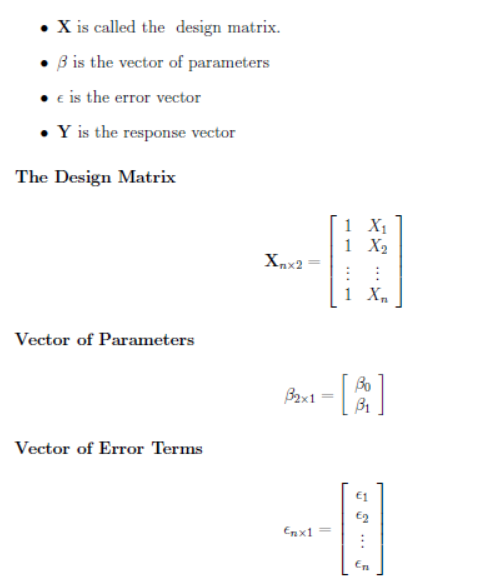
The model gets the best regression fit line by finding the best θ1 and θ2 values.

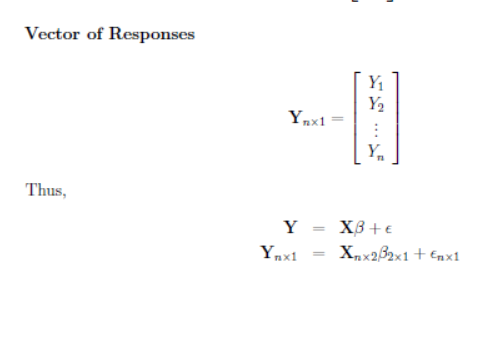
* **θ1:** intercept
* **θ2:** coefficient of x

Once we find the best θ1 and θ2 values, we get the best-fit line. So when we are finally using our model for prediction, it will predict the value of y for the input value of x.

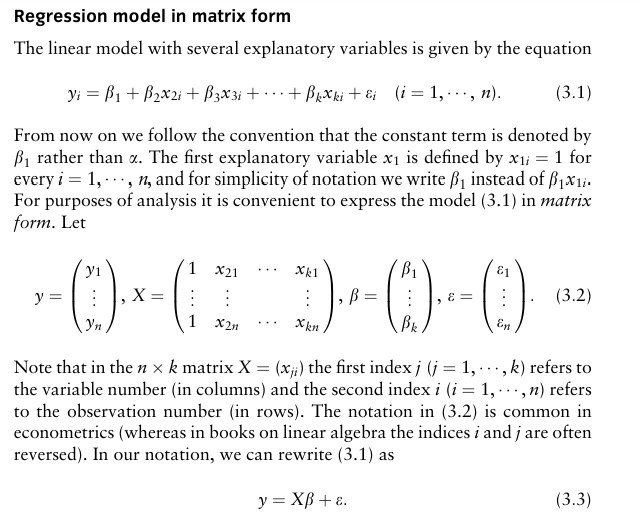
1. Explain simple regression in matrix form.

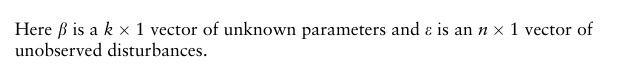






1. Explain Least Squares in Matrix Form.

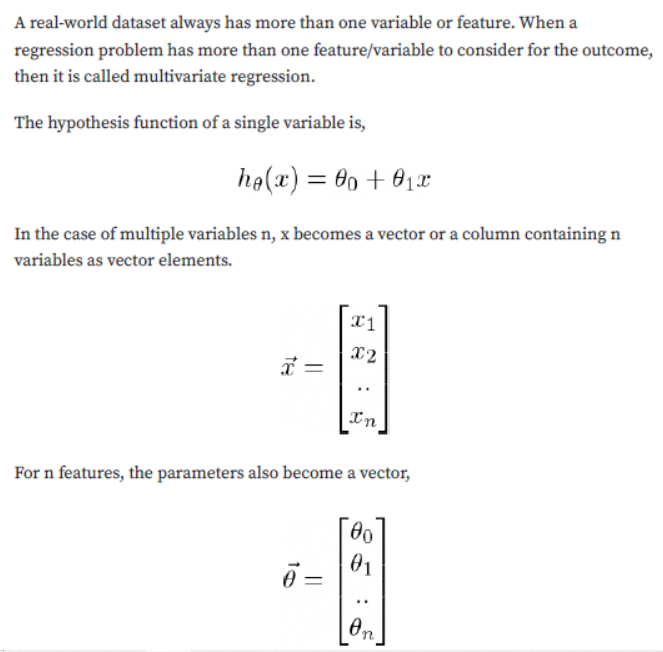


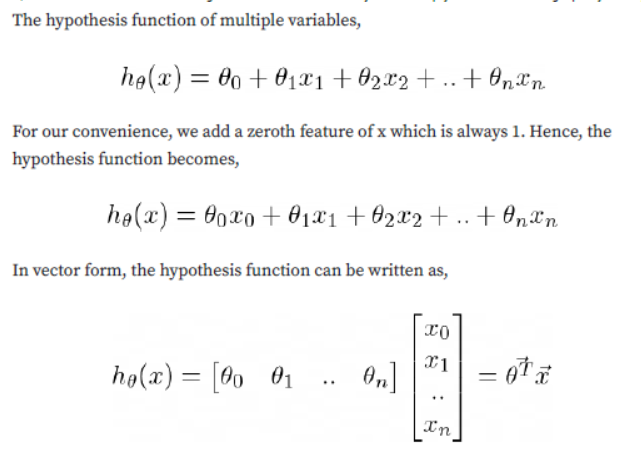


1. Explain Sampling Distribution of Estimators.
2. Using the given data set find the value y when x=10. X={1,1,2,3,4,4,5,6,6,7} Y={2.1,2.5,3.1,3.0,3.5,3.2,4.3,3.9,4.4,4.8}
3. Using the given data set find the value y when x=10. X={1,2,3,4,5,6}

Y={25,35,42,50,55}

1. Explain multivariate linear regression.





1. What is hypothesis function for multivariate linear regression?

