➢ What is Linear Regression?

It is a supervised learning technique used to **predict** a continuous values by modeling a relationship between independent variables(X) and dependent variable(Y) using straight line.

- In simple words it finds the **best straight line** that explain Y changes with X

Equation of Linear Regression :

$$y = mx+b$$

or more general,

$$y = teta0+teta1(x1)+teta2(x2)+ \quad + tetan(xn)$$

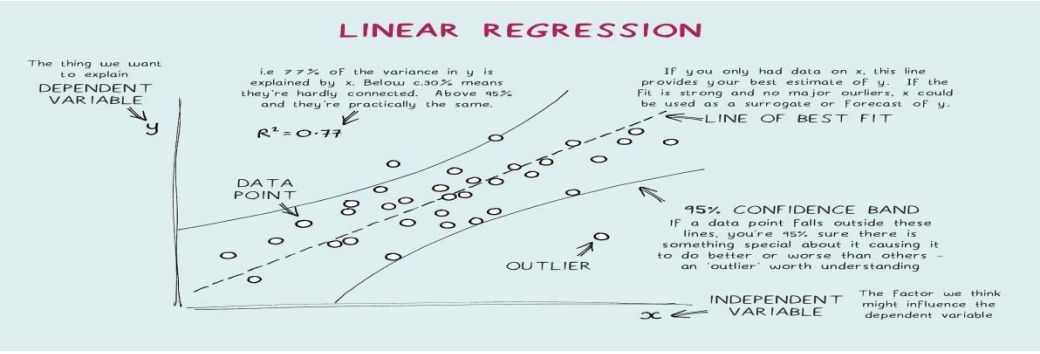
where,

y = dependent variable

x = independent variable

teta0 = y intercept

teta = coefficients(weights)



➢ Types of Linear Regression :

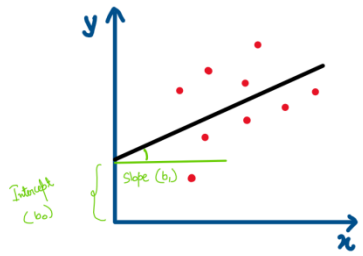
- Simple linear regression
- Multiple linear regression
- Polynomial linear regression
- Logistic linear regression

➔ Simple Linear Regression :

In which one independent and one dependent variable

Example :

Predicting salary(y) based on experience(x).



➔ Multiple Linear Regression:

In which one dependent and two or more independent variables.

❖ Hypothesis :

In linear regression hypothesis/prediction is mean that equation we mentioned in the above slide.
In simple linear regression it is,

$$y = mx + b$$

where,

y = prediction/hyp

x = input feature

b = y intercept

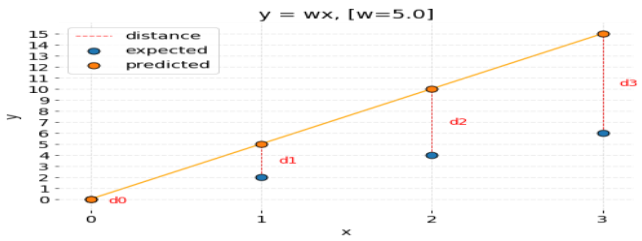
m = slope

```python
# function for prediction
def prediction(exp,teta1,teta2):
    m = len(exp)
    prd = np.zeros(m)
    for i in range(m):
        prd[i] = teta1 + teta2*exp[i]
    return prd
```

❖ Cost function/(Mean Square Error) :

- It tells how bad our model is.
- It calculate the error between actual(y) and predicted(y).
- When smaller cost, the better model.

Mean Squared Error (MSE)= 1/n*sum(yi-yprdi)^2



Now here we can see in the graph that the orange points show the prediction. And the actual data is blue points now the difference between these orange and blue points is error we add all these errors it give the complete error of our prediction line.

Python code,

```python
# cost function
def cost_function(y,prd):
    m = y.shape[0]
    loss = np.sum(np.square(y-prd))
    return loss/(2*m)
```

➢ What is optimization :

It means that improving the model by minimizing the error.
We start with random values of m and b, and keep adjusting them using gradient descent until the model is good.

❖ Gradient Descent :

- It is a common optimization technique.
- Gradient Descent is an iterative algorithm.
- It gradually changes the slope and intercept to reduce the cost.
- Uses derivatives to find the direction of minimum cost.

Repeat until convergence{

M(new) = M(old) – alpha * (derivative of cost with respect to M)

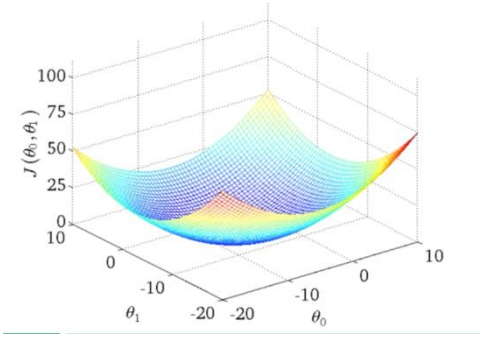
B(new) = B(old) – alpha * (derivative of cost with respect to B)

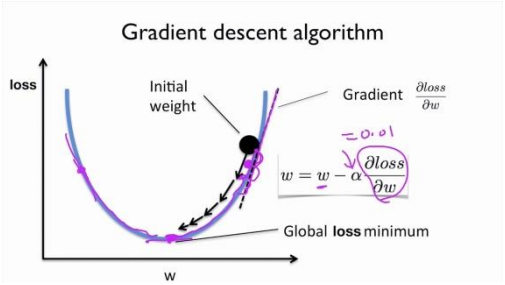
}

Where ,

Alpha = learning rate

We use partial derivative





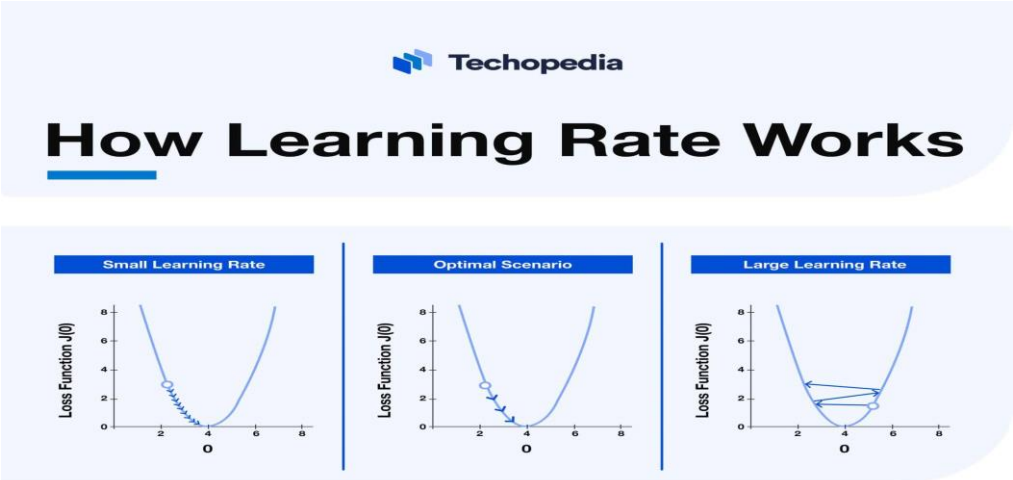
```python
# gradient descent
def gradient_descent(x,y,teta,gdnt,lr):
    m = 50
    cost_hist = []
    for i in range(1000):
        pred = prediction(x,teta)
        gdnt = gradient(x,pred,y)
        teta = teta - lr*(gdnt/m)
        cost = cost_function(y,prd)
        cost_hist.append(cost)
    return teta,cost_hist,pred
```

➔What is Learning rate :

- it controls how fast or slow our model is learns.
- it is like controlling the speed of moving car towards the destination(minimum cost).
- In simple words the steps towards optimal position.

Example : 0.01,0.001,0.1

- When the learning rate value is high then we can not achive to our goal.
- And the value is very low then also we ca not reached because the steps are very small.
- The above values are best and scientifically values for learning rate.
- See the given graphs.



- Types of Gradient Descent :

1. Batch Gradient Descent.
2. Stochastic Gradient Descent.
3. Mini Batch Gradient Descent.

- What is Batch Gradient Descent ?

This Gradient is the normal gradient we use mostly.
This computed the entire dataset.

Advantages :

Stable convergence.
Accurate Gradients.

Disadvantages :

Slow if the dataset is large.
Needs more memory.

- In this we take derivative of the whole in each epoch.

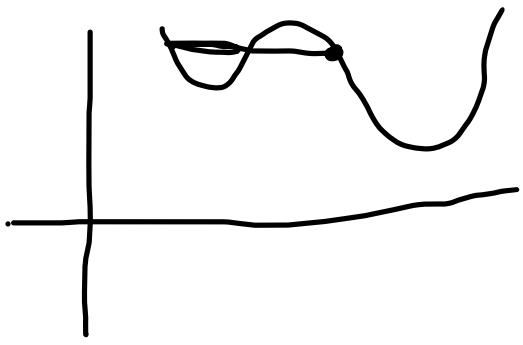
- Stochastic Gradient Descent :

Uses a single data point (example) per update.

helps in non convex functions

Advantages :

Very faster updates.
Can escape local minima.
Useful for large dataset.



Disadvantages :

Noisy may not converges smoothly.
More iterations needed.

- In this we pick a random row in each iteration and updates weights.

➔What is Polynomial Regression :

Polynomial Regression is a form of **linear regression** where the relationship between the independent variable x and the dependent variable y is modeled as an **nth-degree polynomial**.

It is used when the data shows a **non-linear relationship** but can be approximated by a polynomial curve.

When the scatter plot of our data shows a curve trend, the linear regression fails.

It fits a curved line to the data .

It helps in modeling relationships like quadratic, cubic or higher degrees.

Example :

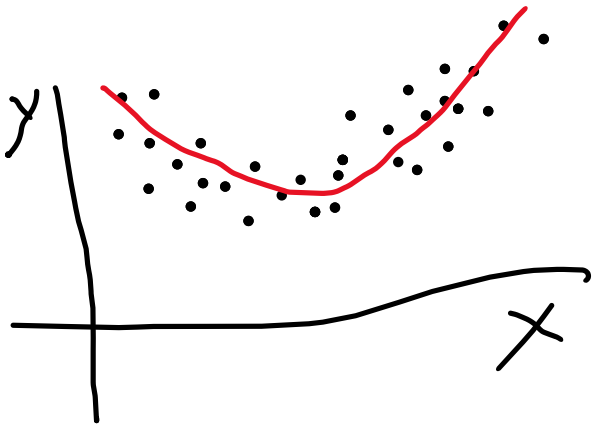
in example the data is

in non linear form it is

not represent on straight

line so that is why we use

polynomial regression,



in which we add polynomial features to independent variable.

General equation :

$$y = \text{teta0} + \text{teta1}*X1 + \text{teta2}*X1^2 + \quad + \text{tetan}*X^n$$

❖ Bias Variance Trade-off :

Bias :

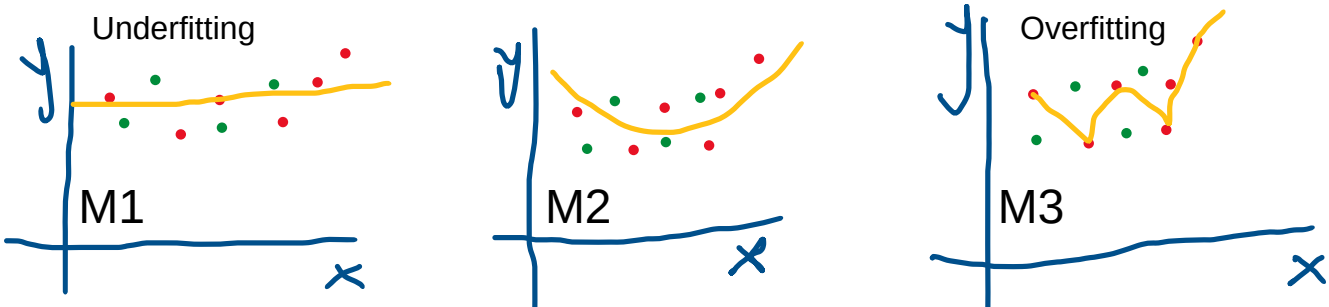
The inability of a machine learning model to truly captured the relationship in the training data.

Variance :

The difference between the training set error and testing set error as known as variance.

Let, training error = 10, and testing error = 50

Variance is = 40



High bias
Low variance
variance

low variance

low bias
high

- ### Overfitting :

When the accuracy of machine learning model on training data is higher than the testing data.

Great performance on training data.

Less performance on testing data.

In the above model M3 is overfitted.

- Underfitting :

  Which does not captured relationship in training data.
  In the above models M1 is underfitted.

- The best model is M2 which has not really good performance or not really bad performance on both dataset is known as bias variance trade-off.

➔ What is Regularization ?

It is a technique use to reduce overfitting problems.

It has the following types.

- Ridge Regression.
- Lasso Regression.
- ElasticNet Regression.

Ridge Regression (L2):

It add a regularization term to loss function.

In Ridge regression the coefficient reached very close to zero.

The high weights reduce more efficiently as compare to low weights.

The general equation become like,

L = sum(y-y^) + lamda(||W||)**2

python code.

```
y_predicted = np.dot(X, self.weights) + self.bias

# Gradient calculation with Ridge penalty
dw = (1/n_samples) * (np.dot(X.T, (y_predicted - y)) + self.alpha * self.weights)
db = (1/n_samples) * np.sum(y_predicted - y)

# Update rule
self.weights -= self.learning_rate * dw
self.bias -= self.learning_rate * db
```

Lasso Regression (L1) :

It is also regularization technique it works like ridge regression but in this the weights can be reached to zero.

It is used in feature selection.

It remove the unnecessary feature columns which has not high effect on output.

Equation,

L = sum(y – y^) + lamda*(magnitude sum of W)

ElasticNet Rgeression :

It is the combination of Ridge and Lasso regression.

In this if have many feature columns and we have no idea about that columns that they are important or not then we use ENT regression.

Equation,

L = sum(y – y^) + a||W||**2 + b||W||

Where,

a = ridge term

b = lasso term