

MACHINE LEARNING

MODULE 2

By

Dr. Nancy Girdhar



THIS PRESENTATION IS ABOUT

- ❑ **Parametric and Non Parametric Learning**

- ❑ **Parametric Learning**

 - **Using Linear Regression**

 - **Gradient Descent**

 - **Linear regression with one variable**

 - **Linear Regression with multiple variables,**

 - **Features Scaling/ Selection**

- ❑ **Dimensionality Reduction**



PARAMETRIC LEARNING

- ⊙ A learning model that **summarizes data with a set of parameters of fixed size** (independent of the number of training examples) is called a parametric model.
- ⊙ No matter how much data you throw at a parametric model, **it won't change its mind about how many parameters it needs.**
- ⊙ The algorithms involve two steps:
 - ❑ Select a form for the function.
 - ❑ **Learn the coefficients** for the function from the training data.

EXAMPLES OF PARAMETRIC LEARNING

- ⊙ Logistic Regression
- ⊙ Linear Discriminant Analysis
- ⊙ Perceptron
- ⊙ Naive Bayes
- ⊙ Simple Neural Networks

BENEFITS OF PARAMETRIC MACHINE LEARNING ALGORITHMS:

- ◎ **Simpler:** These methods are easier to understand and interpret results.
- ◎ **Speed:** Parametric models are very fast to learn from data.
- ◎ **Less Data:** They do not require as much training data and can work well even if the fit to the data is not perfect.

LIMITATIONS OF PARAMETRIC MACHINE LEARNING ALGORITHMS:

- ③ **Constrained:** By choosing a functional form these methods are highly constrained to the specified form.
- ③ **Limited Complexity:** The methods are more suited to simpler problems.
- ③ **Poor Fit:** In practice the methods are unlikely to match the underlying mapping function.

NON-PARAMETRIC MACHINE LEARNING ALGORITHMS

- ◎ Non-parametric methods are good when you **have a lot of data and no prior knowledge**, and when you **don't want to worry too much about choosing just the right features**.
- ◎ Nonparametric methods seek **to best fit the training data in constructing the mapping function**, whilst maintaining some ability to generalize to unseen data. As such, they are able to fit a large number of functional forms.

EXAMPLES OF NON-PARAMETRIC MACHINE LEARNING ALGORITHMS

- ⊙ k-Nearest Neighbors
- ⊙ Decision Trees like CART and C4.5
- ⊙ Support Vector Machines

BENEFITS OF NON-PARAMETRIC MACHINE LEARNING ALGORITHMS

- ◎ **Flexibility:** Capable of fitting a large number of functional forms.
- ◎ **Power:** No assumptions (or weak assumptions) about the underlying function.
- ◎ **Performance:** Can result in higher performance models for prediction.

LIMITATIONS OF NON-PARAMETRIC MACHINE LEARNING ALGORITHMS

- ◎ **More data:** Require a lot more training data to estimate the mapping function.
- ◎ **Slower:** A lot slower to train as they often have far more parameters to train.
- ◎ **Overfitting:** More of a risk to overfit the training data and it is harder to explain why specific predictions are made.

PARAMETRIC VS NON-PARAMETRIC ALGORITHMS

	Parametric	Non-parametric
Assumed distribution	Normal	Any
Assumed variance	Homogeneous	Homogenous and Heterogeneous
Typical data	Ratio or Interval	Ordinal or Nominal
Data set relationships	Independent	Any
Usual central measure	Mean	Median
Benefits	Can draw more conclusions	Simplicity; Less affected by outliers

LINEAR REGRESSION

LINEAR REGRESSION

Reference Links

- https://realpython.com/linear-regression-in-python/?_cf_chl_captcha_tk=1fe9669d3fec714e8651f088e7ef58cb314e026-1579060931-0-ARMN3Ev9d-x19cl81yMz6o_Ef9itlQdZLbHheaRryZs_Uq4cXmcljQKpMgETH0SQ1VFRbG-NeImtJxOPR4IVSW7LXk6G3cYv1ByCI0esTzPyanWsYQ_Mo3qQ5QfL9G9rkP1ymQ_I_SkBwYzx2kvFzOR93sRiQqHlLF6giuLkGHZXzTxhK3Srh4CDK05jRXjYUKl4rOLQFkeJ_MBOsDNNcKxVO-ArW4wQD5YFtMj0FlkyR9nJoSpgynQYX9px8WGfdhfBYAHIHLgx6hrNK-MeHnRsZXyFgjtXt47M5pmCAfxut5u4_KOf2DS4cW0trVLMytBwm8AFbQDPZvRxo3C0PhESsL-gF7LMMVysYYHhhVkgq
- <https://towardsdatascience.com/a-beginners-guide-to-linear-regression-in-python-with-scikit-learn-83a8f7ae2b4f>
- <https://towardsdatascience.com/simple-and-multiple-linear-regression-in-python-c928425168f9>
- https://scikit-learn.org/stable/auto_examples/linear_model/plot_ols.html

REGRESSION MODELS

- ◎ Relationship between one **dependent variable** and **explanatory (independent) variable(s)**
- ◎ Use equation to set up relationship
 - **Numerical** Dependent Variable (Response)
 - 1 or More Numerical or Categorical Independent Variables (Explanatory)
- ◎ Used Mainly for Prediction & Estimation

CLASSIFICATION VS REGRESSION

2 KEY DIFFERENCES

CLASSIFICATION

A tree model where the target variable can take a discrete set of values.

The dependent variables are categorical.

REGRESSION

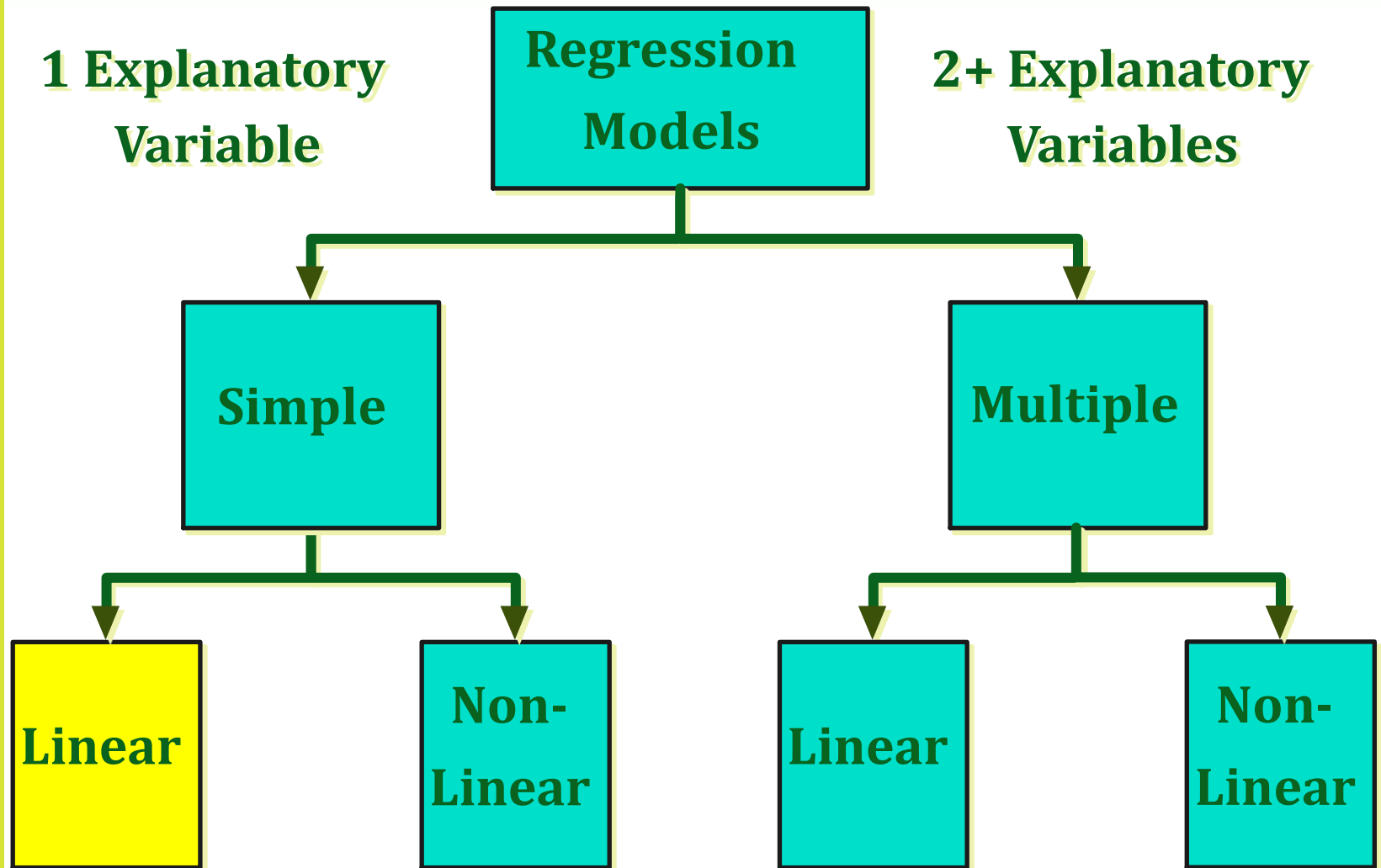
A tree model where the target variable can take continuous values typically real numbers.

The dependent variables are numerical.

REGRESSION MODELING STEPS

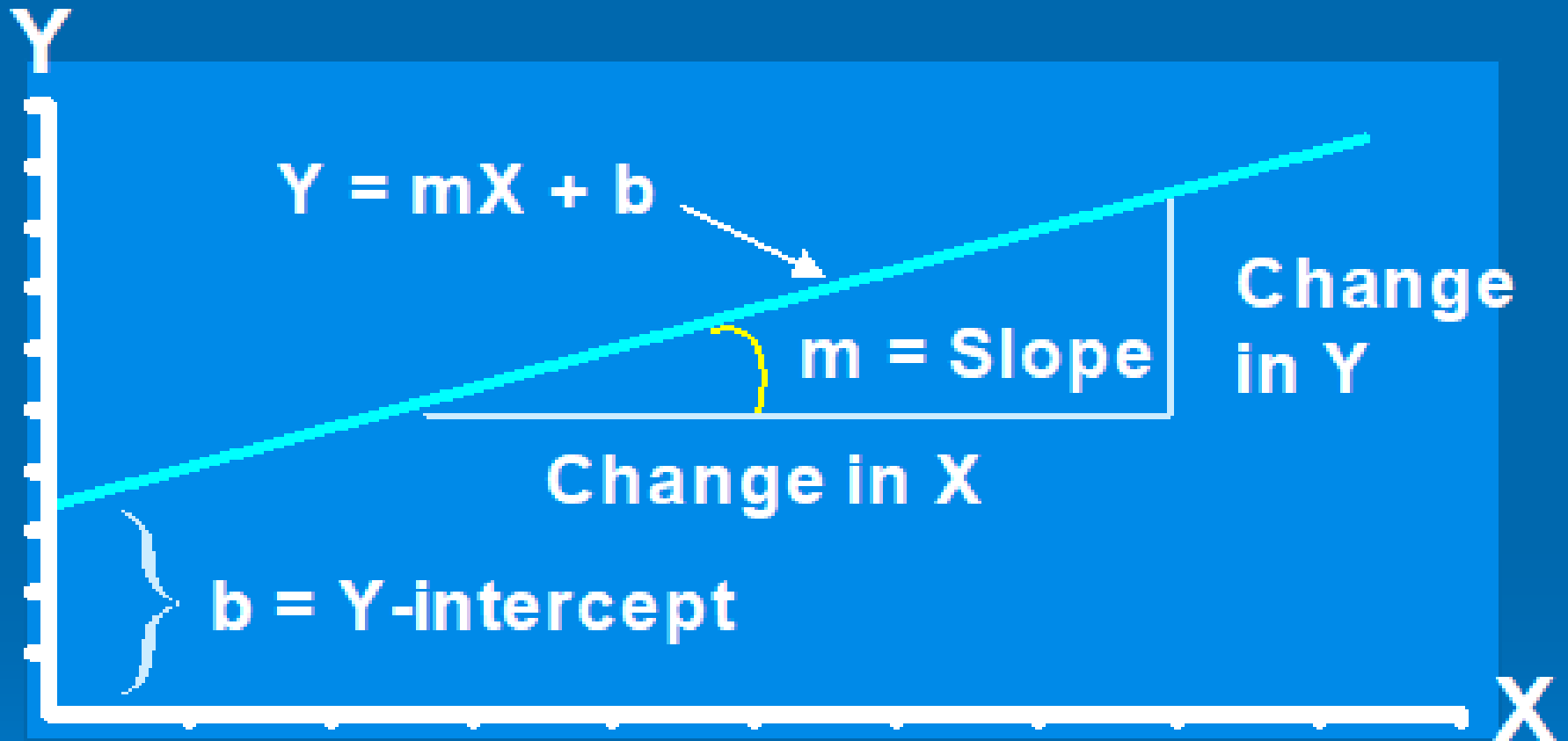
- ① 1. Hypothesize Deterministic Component
 - Estimate Unknown Parameters
- ② 2. Specify Probability Distribution of Random Error Term
 - Estimate Standard Deviation of Error
- ③ 3. Evaluate the Fitted Model
- ④ 4. Use Model for Prediction & Estimation

TYPES OF REGRESSION MODELS



LINEAR REGRESSION MODELS

LINEAR EQUATION

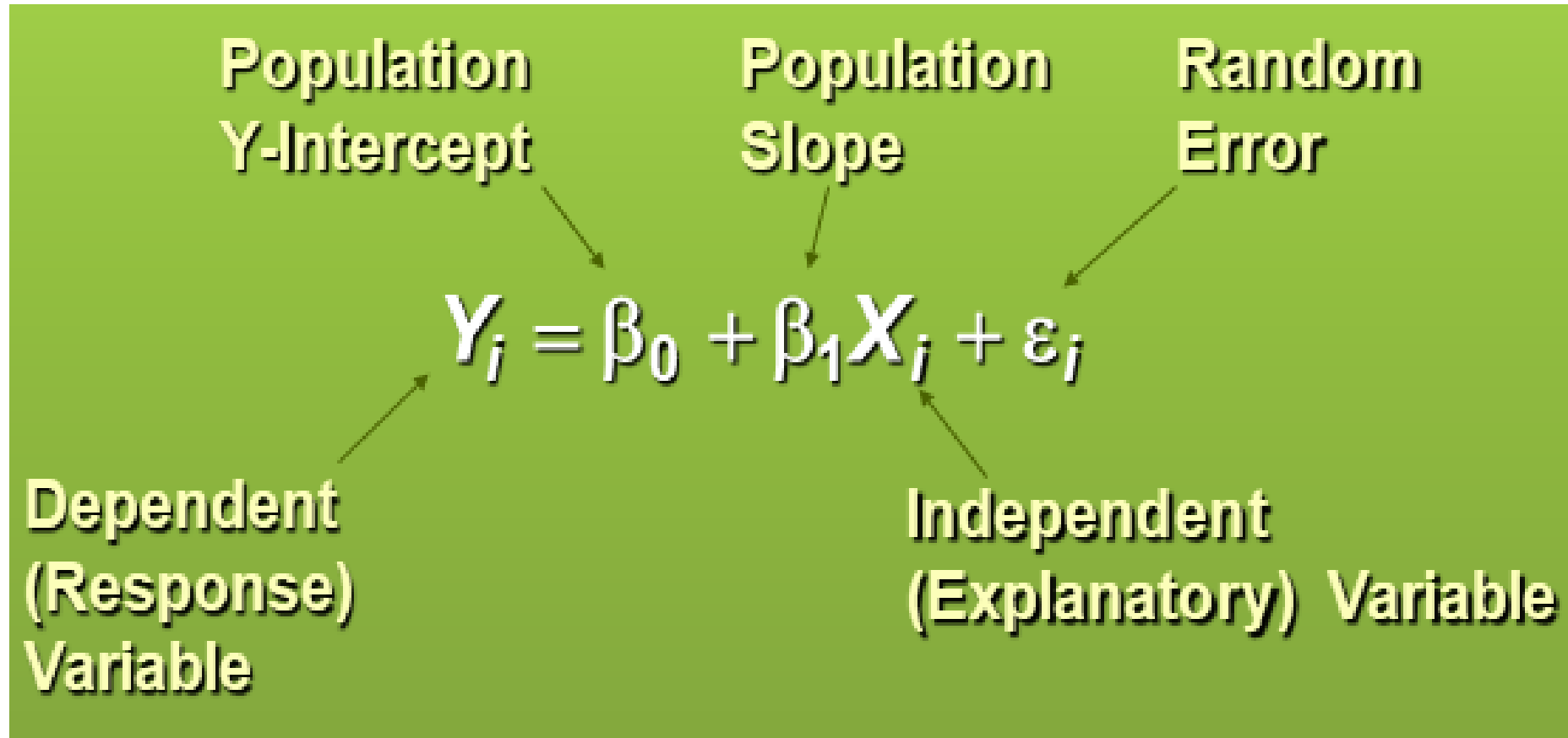


$$Y = mX + b$$

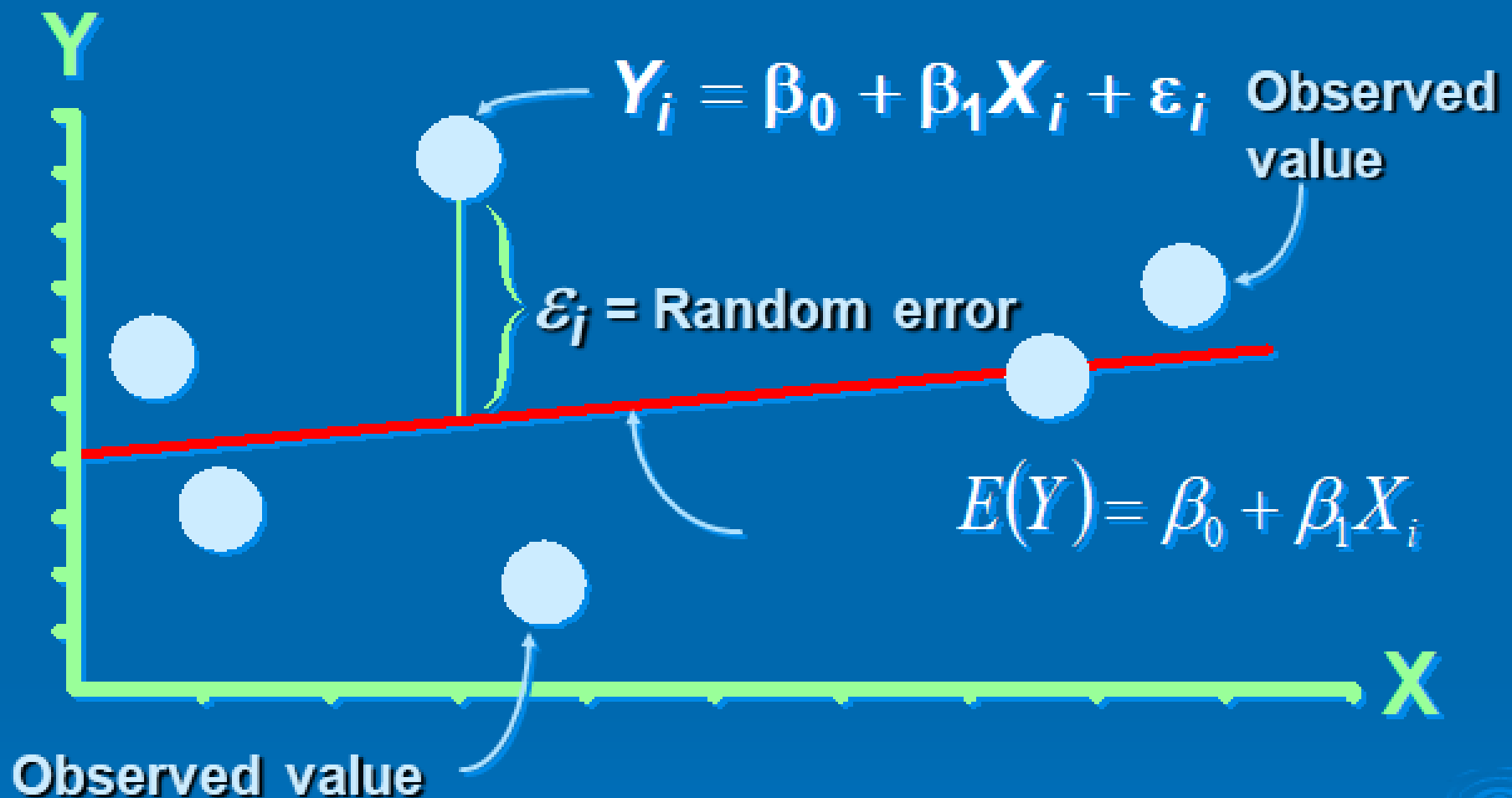
here, m = slope and b = Y-intercept

LINEAR REGRESSION MODEL

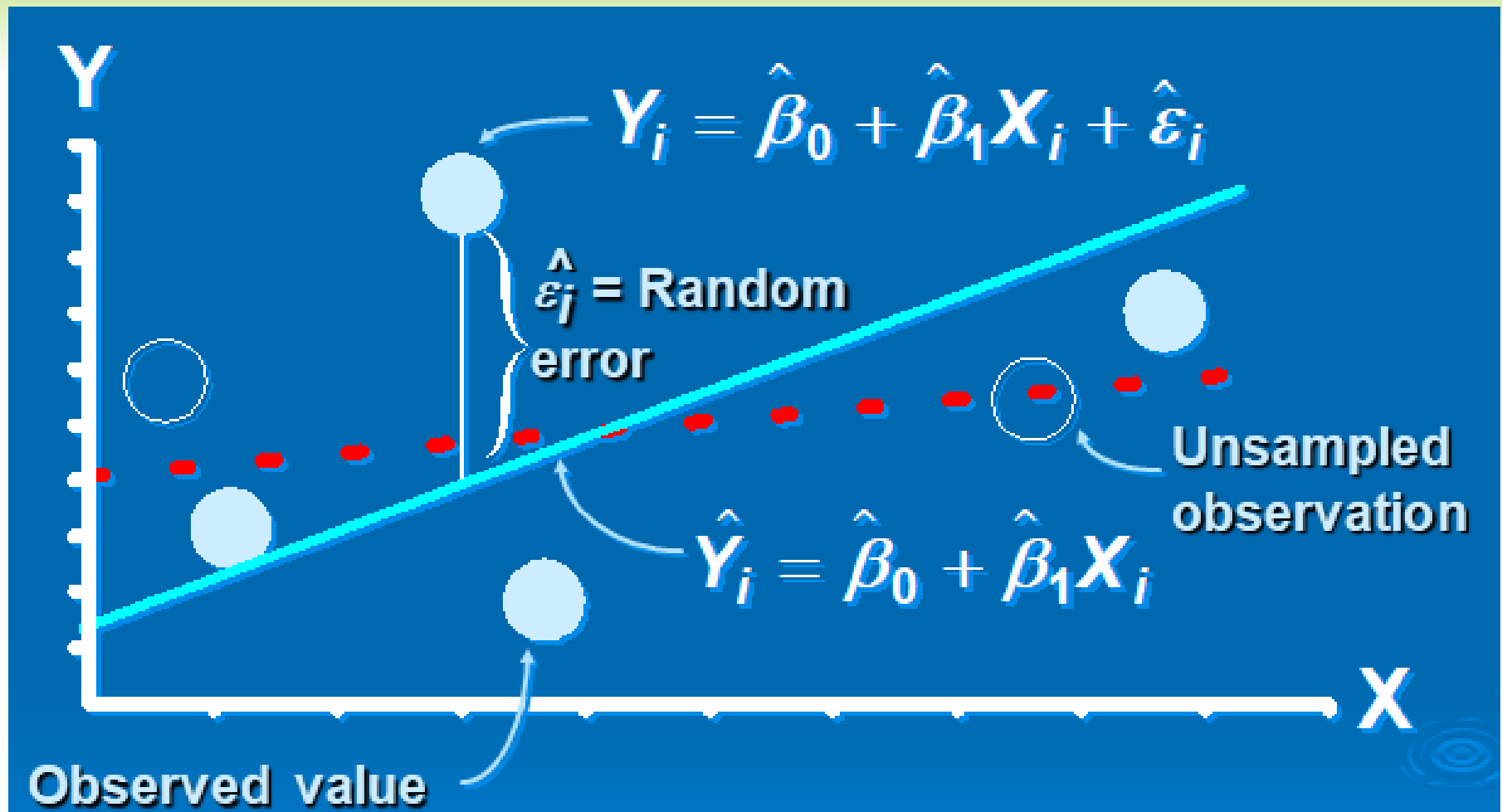
- **Relationship between Variables** is a **Linear Function**



POPULATION LINEAR REGRESSION MODEL



SAMPLE LINEAR REGRESSION MODEL



SIMPLE LINEAR REGRESSION

- Managerial **decisions** are often based on **relationship between two or more variables**.
- **Regression analysis** can be used to develop an equation showing **HOW THE VARIABLES ARE RELATED**.
- The **variable being predicted** is called **DEPENDENT variable** and denoted by **y** .
- The **variable being used to predict** the value of the dependent variable is called **INDEPENDENT variable** and is denoted by **x** .

SIMPLE LINEAR REGRESSION

- It involves **one independent variable** and **one dependent variable**.
- The **relationship** between the two variables is approximated by a **straight line**.
- Regression analysis involving **two or more independent variables** is called **MULTIPLE LINEAR REGRESSION**.

...SIMPLE LINEAR REGRESSION MODEL

The equation that describes **how y is related to x** and **an error term** is called the regression model.

The simple linear regression model is:

$$y = b_0 + b_1x + e$$

where:

b_0 and b_1 are called parameters of the model,
 e is a random variable called the error term.

SIMPLE LINEAR REGRESSION EQUATION

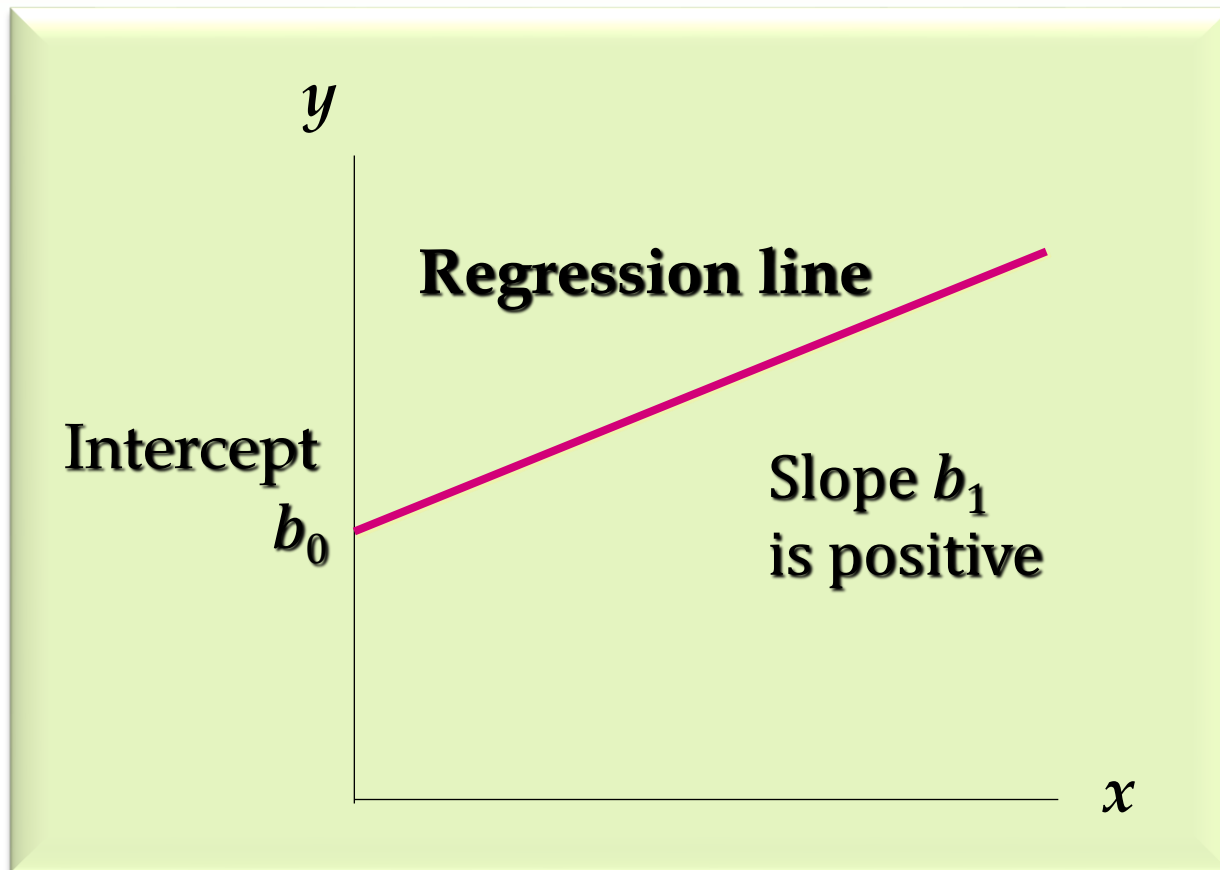
The simple linear regression equation is:

$$y = b_0 + b_1x$$

- **Graph** of the regression equation is a **straight line**.
- b_0 is the **y-intercept** of the regression line.
- b_1 is the **slope** of the regression line.
- y is the **expected value** of y for a given x value.

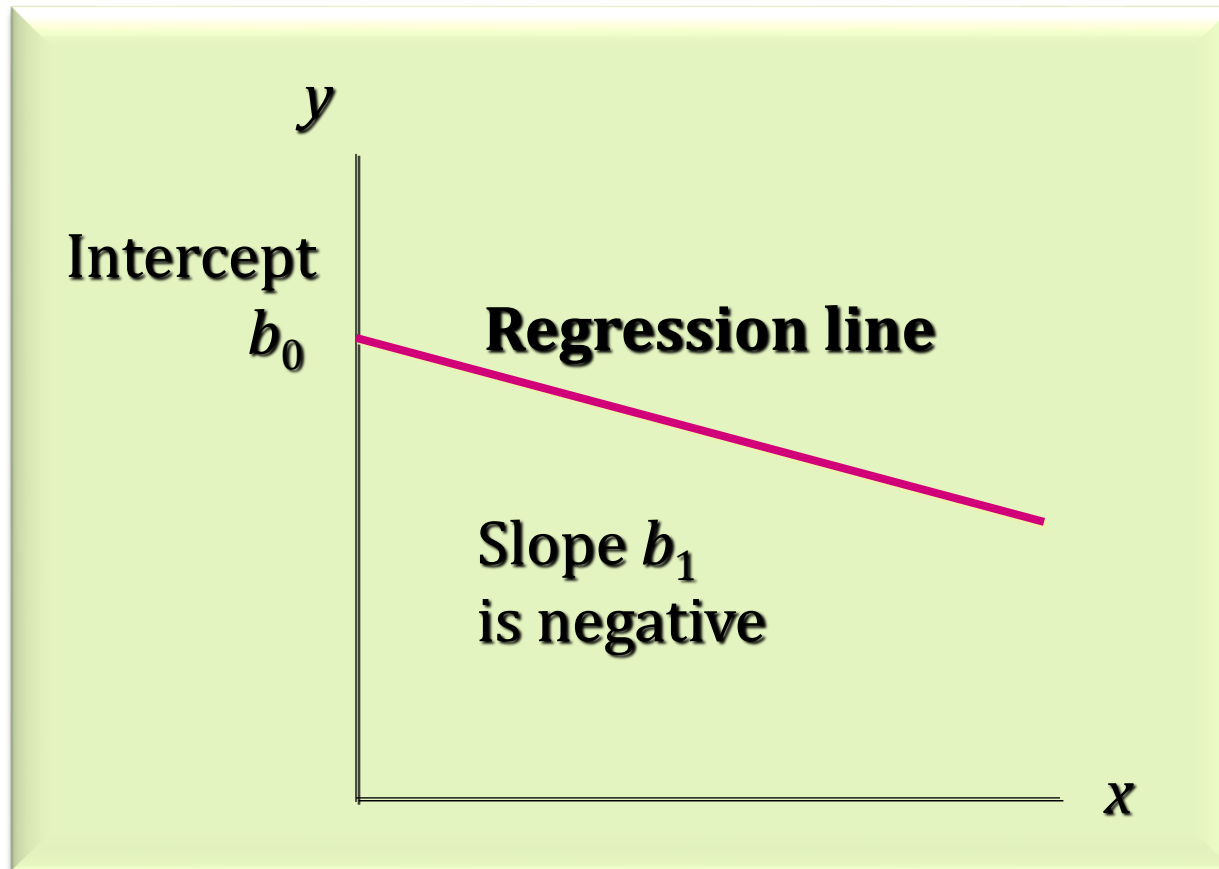
....SIMPLE LINEAR REGRESSION

■ Positive Linear Relationship



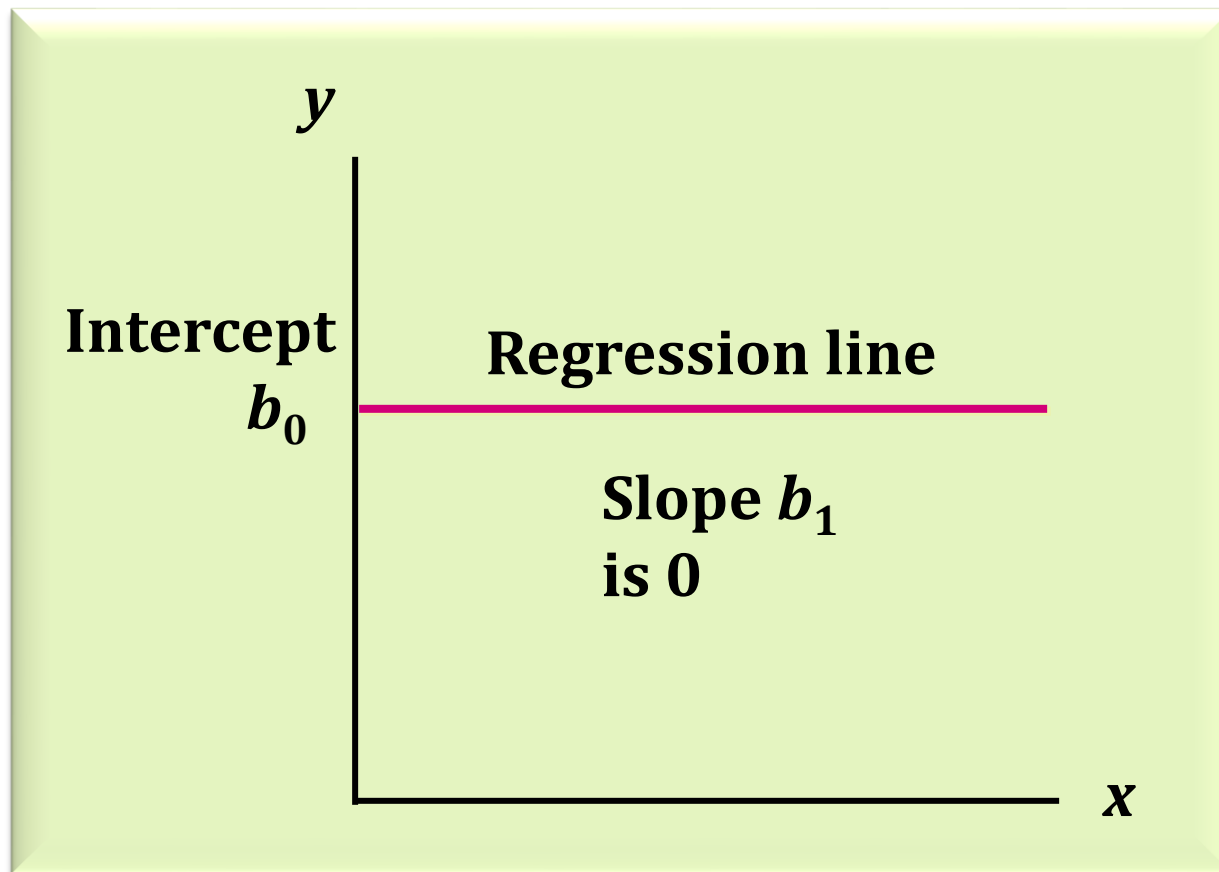
....SIMPLE LINEAR REGRESSION

- **Negative Linear Relationship**



....SIMPLE LINEAR REGRESSION

- No Relationship



...SIMPLE LINEAR REGRESSION

- Slope for the Estimated Regression Equation

$$b_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{(x_i - \bar{x})^2}$$

where:

x_i = value of independent variable for i th observation

y_i = value of dependent variable for i th observation

\bar{x} = mean value for independent variable

\bar{y} = mean value for dependent variable

- y -Intercept for the Estimated Regression Equation

$$b_0 = \bar{y} - b_1 \bar{x}$$

EXAMPLE: MARUTI AUTO SALE

Maruti Auto periodically has a special week-long sale. As part of the advertising campaign Reed runs one or more television commercials during the weekend preceding the sale. Data from a sample of 5 previous sales are shown on the next slide.

<u>Number of TV Ads (x)</u>	<u>Number of Cars Sold (y)</u>
1	14
3	24
2	18
1	17
3	27
<u> </u>	<u> </u>
$\Sigma x = 10$	$\Sigma y = 100$

Step 1: Compute

(a) $\bar{x} = 2$

(b) $\bar{y} = 20$

EXAMPLE: MARUTI AUTO SALE

Step 2: Slope for the Estimated Regression Equation

$$b_1 = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{(x_i - \bar{x})^2} = \frac{20}{4} = 5$$

Step 3: y-intercept for the Estimated Regression Equation

$$b_0 = \bar{y} - b_1\bar{x} = 20 - 5(2) = 10$$

Step 4: Estimated Regression Equation

$$y = b_0 + b_1x = 10 + 5x$$

LEAST SQUARE METHOD

- Least Squares Criterion

$$\min \sum (y_i - \hat{y}_i)^2$$

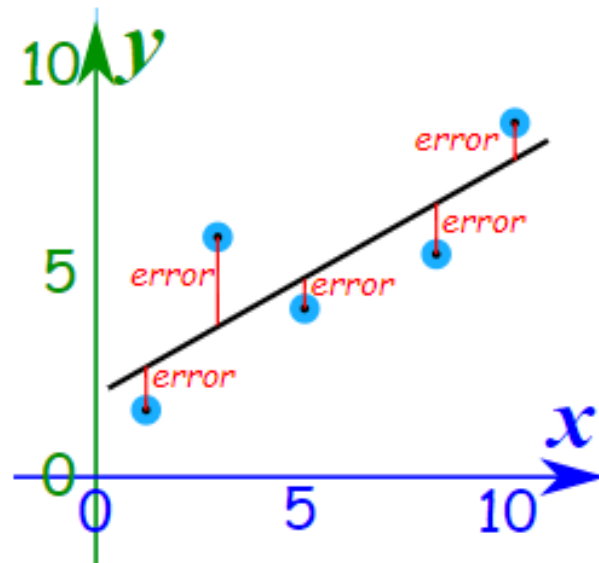
where:

y_i = **observed value** of the dependent variable
for the i th observation

\hat{y}_i = **estimated value** of the dependent variable
for the i th observation

HOW DOES IT WORK?

- ◎ It works by making the total of the **square of the errors** as small as possible (that is why it is called "least squares"):
- ◎ So, when we square each of those errors and add them all up, the total is as small as possible.



The straight line minimizes the sum of squared errors

EXAMPLE 1: LEAST SQUARE REGRESSION

x	y	x y	x ²
-1	0	0	1
0	2	0	0
1	4	4	1
2	5	10	4
$\Sigma x = 2$	$\Sigma y = 11$	$\Sigma x y = 14$	$\Sigma x^2 = 6$

$$\begin{aligned} b_1 &= (n\Sigma x y - \Sigma x \Sigma y) / (n\Sigma x^2 - (\Sigma x)^2) \\ &= (4*14 - 2*11) / (4*6 - 2^2) \\ &= 17/10 = 1.7 \end{aligned}$$

$$\begin{aligned} b_0 &= (1/n)(\Sigma y - b_1 \Sigma x) \\ &= (1/4)(11 - 1.7*2) \\ &= 1.9 \end{aligned}$$

The regression line is $y=1.7x + 1.9$

EXAMPLE 2: LEAST SQUARE REGRESSION

x	y	x y	x ²
0	2	0	0
1	3	3	1
2	5	10	4
3	4	12	9
4	6	24	16
$\Sigma x = 10$	$\Sigma y = 20$	$\Sigma x y = 49$	$\Sigma x^2 = 30$

$$\begin{aligned} b_1 &= (n\Sigma x y - \Sigma x \Sigma y) / (n\Sigma x^2 - (\Sigma x)^2) \\ &= (5*49 - 10*20) / (5*30 - 10^2) \\ &= 0.9 \end{aligned}$$

$$b_0 = (1/n)(\Sigma y - b_1 \Sigma x) = (1/5)(20 - 0.9*10) = 2.2$$

The regression line is $y=0.9x + 2.2$

COEFFICIENT OF DETERMINATION

R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.

The coefficient of determination is given by the following equation:

$$r^2 = SSR/SST$$

where:

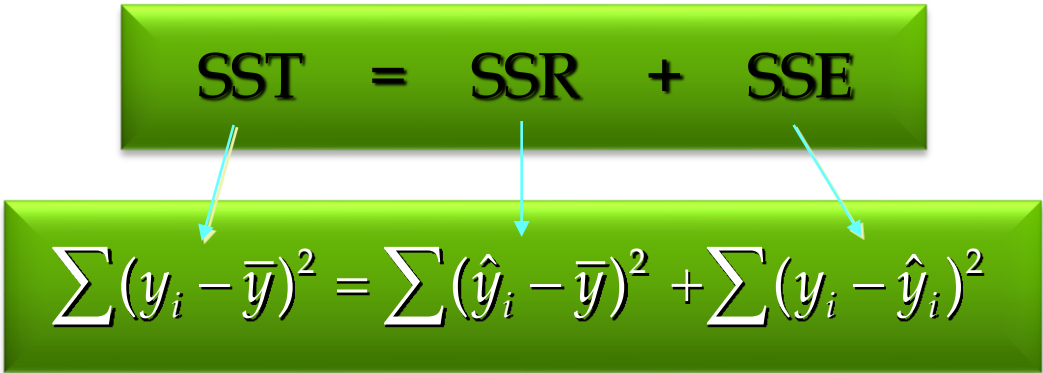
SSR = sum of squares due to regression

SST = total sum of squares

COEFFICIENT OF DETERMINATION

- Relationship Among SST, SSR, SSE

$$\text{SST} = \text{SSR} + \text{SSE}$$


$$\sum (y_i - \bar{y})^2 = \sum (\hat{y}_i - \bar{y})^2 + \sum (y_i - \hat{y}_i)^2$$

where:

SST = total sum of squares

SSR = sum of squares due to regression

SSE = sum of squares due to error

SAMPLE CORRELATION COEFFICIENT

$$r_{xy} = (\text{sign of } b_1) \sqrt{\text{Coefficient_of_determination}}$$

$$r_{xy} = (\text{sign of } b_1) \sqrt{r^2}$$

where:

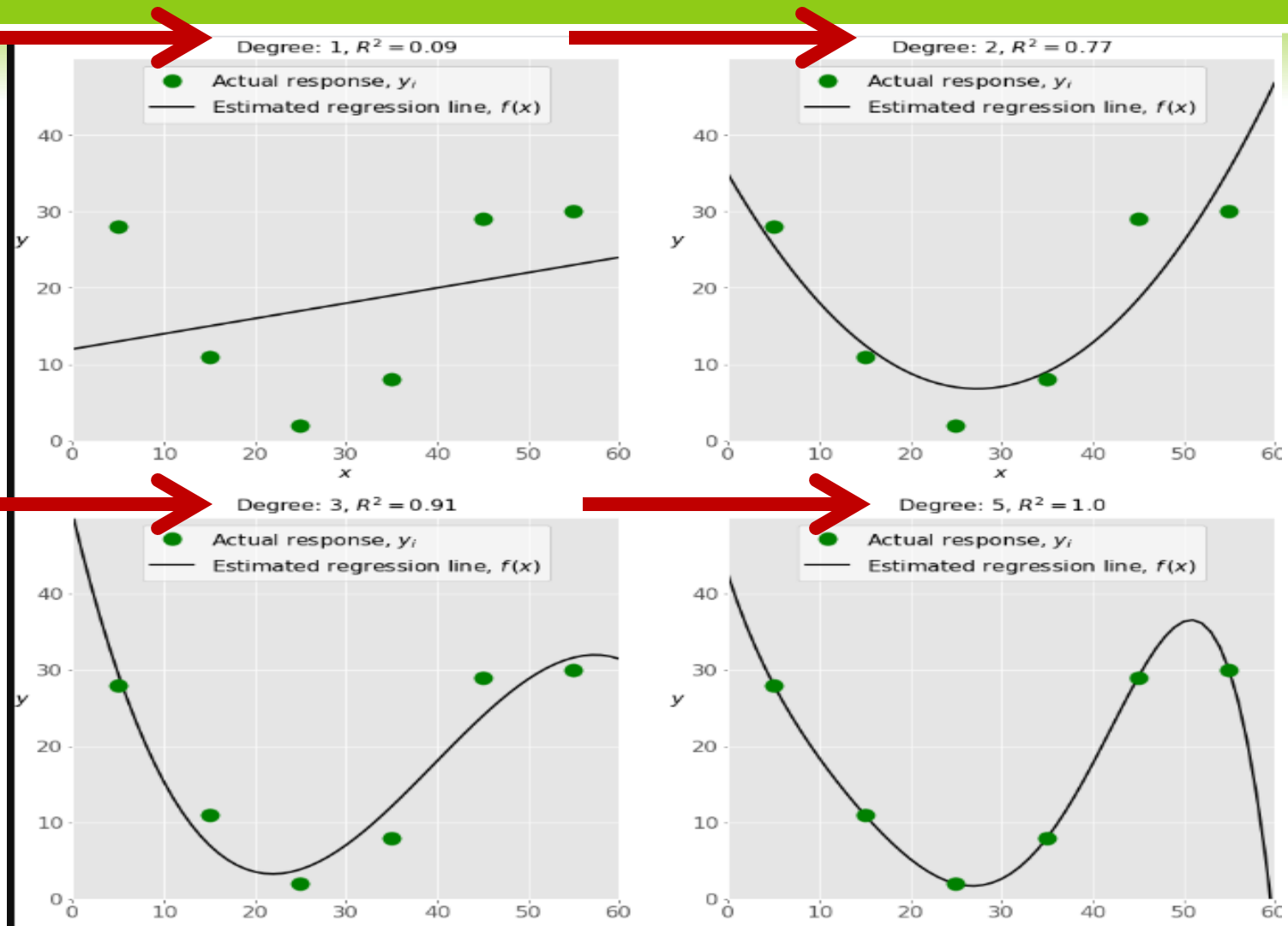
b_1 = the slope of the estimated regression
equation $\hat{y} = b_0 + b_1x$

The sign of b_1 in the equation is “+”.

$$r_{xy} = + \sqrt{0.8772}$$

$$r_{xy} = + .9366$$

...SIGNIFICANCE OF COEFFICIENT OF DETERMINATION



Observe that as the value of R^2 increases the more points lie on the curve and at $R^2 = 1$, all points are on the curve.

MULTIPLE LINEAR REGRESSION

MULTIPLE LINEAR REGRESSION

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} + \epsilon$$

where, for $i = n$ observations:

y_i = dependent variable

x_i = explanatory variables

β_0 = y-intercept (constant term)

β_p = slope coefficients for each explanatory variable

ϵ = the model's error term (also known as the residuals)

MULTI-LINEAR REGRESSION ASSUMPTIONS

- ⊙ There is a linear relationship between the dependent variables and the independent variables.
- ⊙ The independent variables are not too highly correlated with each other.
- ⊙ y_i observations are selected independently and randomly from the population.
- ⊙ Residuals should be normally distributed with a mean of 0 and variance σ .

VARIANCE

Variance measures variability from the average or mean.

The variance statistic can help determine the risk an investor assumes when purchasing a specific security.

- **Variance can be negative.**
- **A variance value of zero indicates that all values within a set of numbers are identical**

The Formula for Variance Is

$$\text{variance } \sigma^2 = \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n}$$

where:

x_i = the i^{th} data point

\bar{x} = the mean of all data points

n = the number of data points

LOGISTIC REGRESSION

LOGISTIC REGRESSION

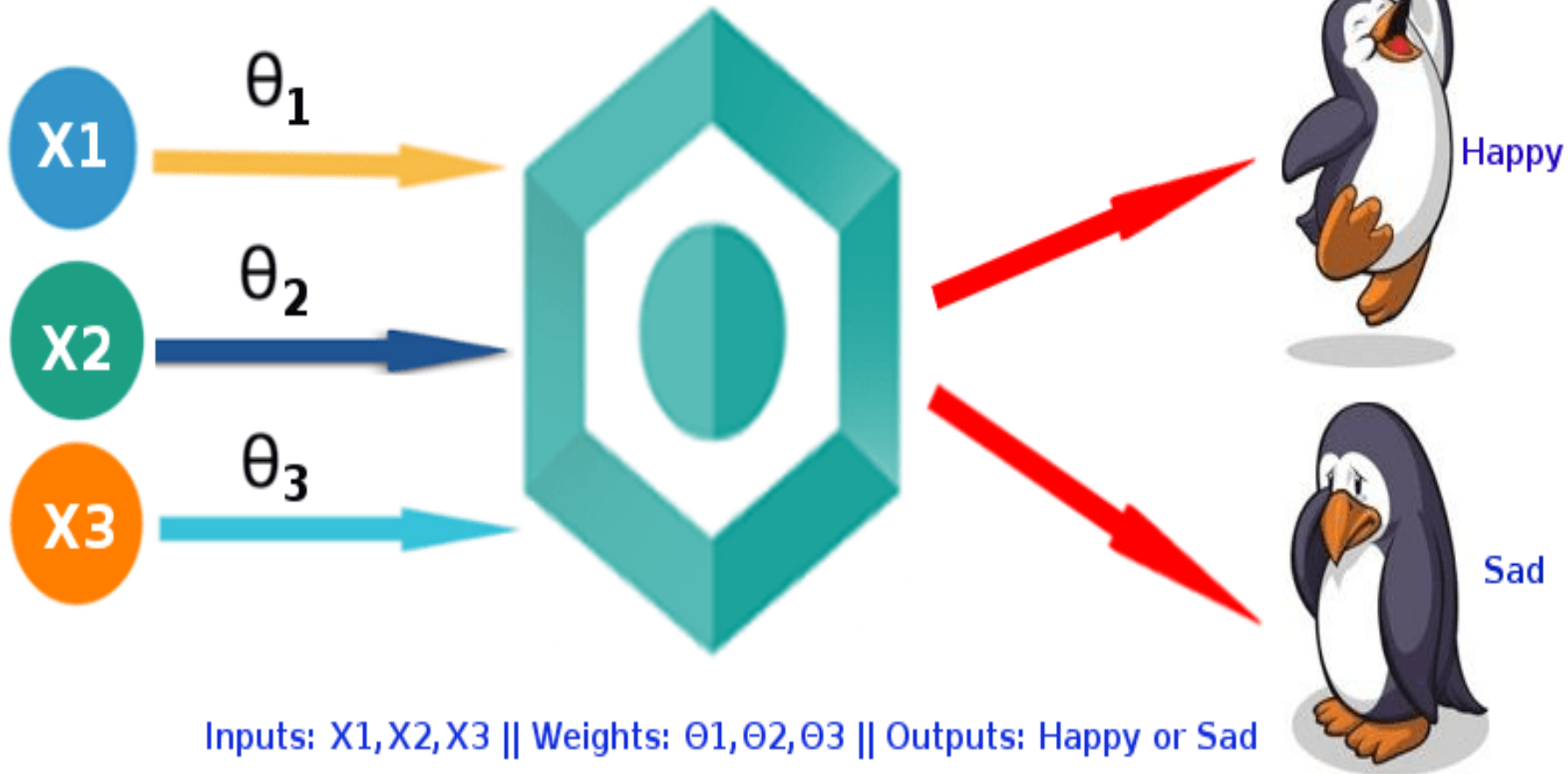
- **Supervised classification algorithm.**
- In a classification problem, the target variable(or output), y , can take only discrete values for given set of features(or inputs), X .
- We can also say that the target variable is **categorical**.

Logistic Regression

- ❖ Dependent variable is binary:
 1 (True, Success) and 0 (False, Failure)
- ❖ Goal is to find best fitting model for independent and dependent variable relationship.
- ❖ Independent variables can be continuous or binary.

GRAPHICAL REPRESENTATION: LOGISTIC REGRESSION

Logistic Regression Model



EXAMPLE: INSURANCE DATA

age	have_insurance
-----	----------------

22	0
----	---

25	0
----	---

47	1
----	---

52	0
----	---

46	1
----	---

56	1
----	---

55	0
----	---

60	1
----	---

62	1
----	---

61	1
----	---

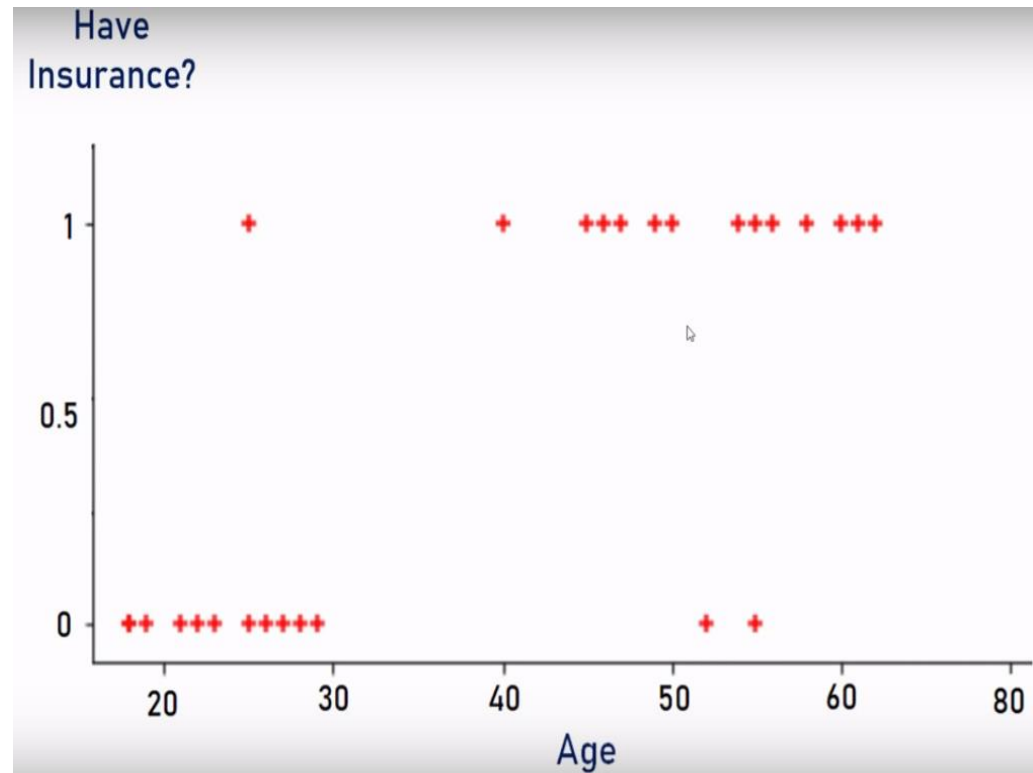
18	0
----	---

28	0
----	---

27	0
----	---

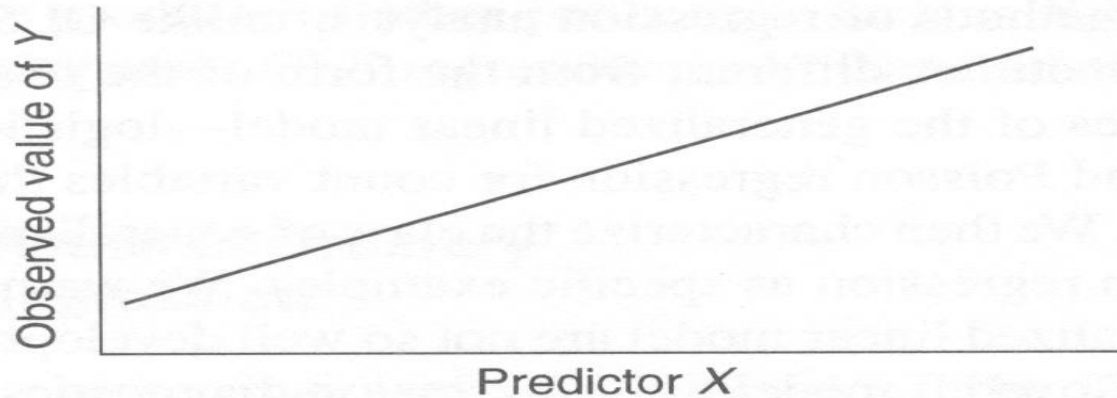
29	0
----	---

49	1
----	---

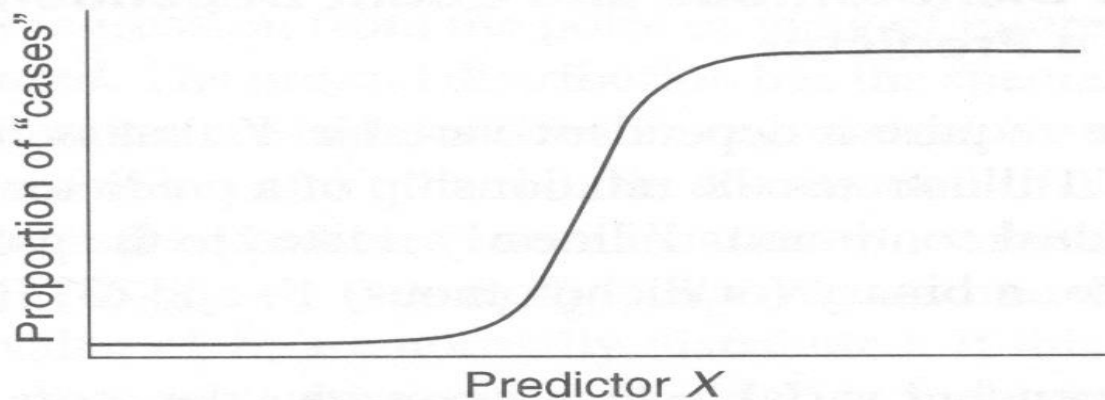


LOGISTIC REGRESSION VS LINEAR REGRESSION

(A) For a continuous outcome variable Y , the numerical value of Y at each value of X .



(B) For a binary outcome variable, the proportion of individuals who are “cases” (exhibit a particular outcome property) at each value of X .



LINEAR VS LOGISTIC REGRESSION

Linear Regression

1. Home prices
2. Weather
3. Stock price

Predicted value is
continuous

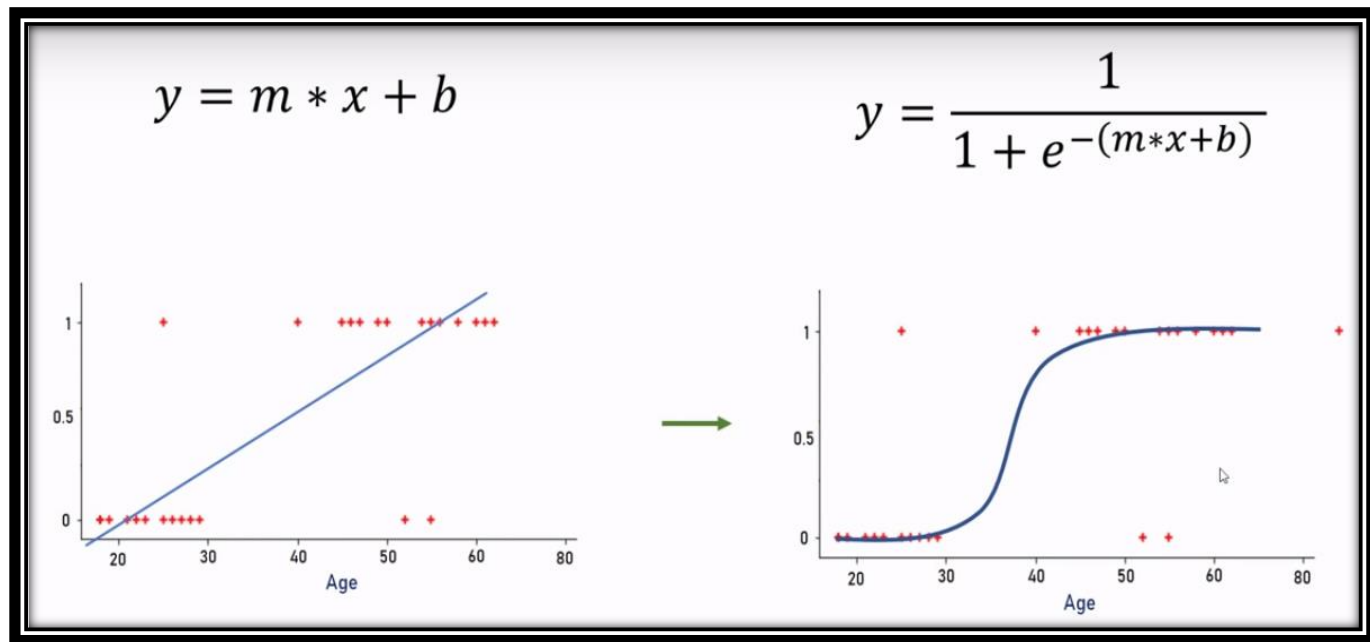
Logistic Regression

1. Email is spam or not
2. Will customer buy life insurance?
3. Which party a person is going to vote for?
 1. Democratic
 2. Republican
 3. Independent

Predicted value is
categorical

LINEAR VS LOGISTIC REGRESSION

Linear Vs Logistic



LOGISTIC REGRESSION

In linear regression, the output Y is in the same units as the target variable (the thing you are trying to predict). However, in logistic regression the output Y is in log odds.

Odds is just another way of expressing the probability of an event, $P(Event)$.

$$Odds = \frac{P(Event)}{(1 - P(Event))}$$

Sigmoid function converts input range 0 to 1

$$sigmoid(z) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + e^{-(\beta^T x_i)}}$$

e= Euler's number~2.71828

TYPES OF LOGISTIC REGRESSION

❑ **Binomial:**

- Target variable can have only 2 possible types: “0” or “1” which may represent “win” vs “loss”, “pass” vs “fail”, “dead” vs “alive”, etc.

❑ **Multinomial:**

- Target variable can have 3 or more possible types which are not ordered(i.E. Types have no quantitative significance) like “disease A” vs “disease B” vs “disease C”.

❑ **Ordinal:**

- It deals with target variables with ordered categories. For example, a test score can be categorized as:“very poor”, “poor”, “good”, “very good”. Here, each category can be given a score like 0, 1, 2, 3.

BASIC CONCEPT

The odds ratio $OR_{A,B}$, that compares the odds of events E_A and E_B (that is, Event E occurring in group A and B , respectively), is defined as the ratio between the two odds; that is

$$OR_{A \text{ vs } B} = \frac{\text{Odds}(E_A)}{\text{Odds}(E_B)} = \frac{P(E_A)}{(1 - P(E_A))} \bigg/ \frac{P(E_B)}{(1 - P(E_B))}$$

In particular, if an odds ratio is equal to one, the odds are the same for the two groups. Note that, if we define a factor with levels corresponding to groups A and B , respectively, then an odds ratio equal to one is equivalent to there being no factor-effect.

FINDING THE ODDS

- The odds in favor - the ratio of the number of ways that an outcome can occur compared to how many ways it cannot occur.

Odds in favor = Number of successes: Number of failures

- The odds against - the ratio of the number of ways that an outcome cannot occur compared to in how many ways it can occur.

Odds against = Number of failures: Number of successes

Odds in favor:

Odds in favor of a particular event are given by the ratio of Number of favorable outcomes to Number of unfavorable outcomes.

$$P(A) = \frac{\text{Number of favorable outcomes}}{\text{Number of unfavorable outcomes}}$$

FINDING THE ODDS : EXAMPLE

A jewelry box contains 5 white pearl, 2 gold rings and 6 silver rings. What are the odds of drawing a white pearl from the jewelry box?

Number of successes = 5

Number of failures = 2 + 6 = 8

Numbers of ways to draw a white pearl: number of ways to draw another jewelry.

5:8

The odds are 5:8

FINDING THE ODDS: EXAMPLE 2

Problem

Find the odds in favor of throwing a die to get “3 dots”.

Solution

Total number of outcomes in throwing a die = 6

Number of favorable outcomes = 1

Number of unfavorable outcomes = $(6 - 1) = 5$

Therefore, odds in favor of throwing a die to get “3 dots” is $1 : 5$ or $1/5$

DERIVATION OF LOGISTIC REGRESSION EQUATION

We know that

$$odds = \frac{P}{1 - P}$$

In logistic regression, **the dependent variable is a *logit***, which is the natural log of the odds, that is,

$$\log(odds) = \text{logit}(P) = \ln\left(\frac{P}{1 - P}\right) \quad (1)$$

So a *logit* is a log of odds and **odds are a function of P, the probability of a 1**. The log odds (*logit*) is assumed to be linearly related to X . So, in logistic regression, we find

$$\text{logit}(P) = b_0 + b_1X \quad (2)$$

,

...DERIVATION

From eq (1) and eq (2)

$$\ln \frac{P}{1-P} = b_0 + b_1 X$$

Take antilog on both sides

$$\frac{P}{1-P} = e^{(b_0+b_1X)}$$

$$P = \frac{e^{(b_0+b_1X)}}{1 + e^{(b_0+b_1X)}}$$

or

$$P = \frac{1}{1 + e^{-(b_0+b_1X)}}$$

The regression coefficient for logistic regression is calculated using **maximum likelihood estimation**

EXAMPLE OF LOGISTIC REGRESSION

Person	2 nd Heart Attack	Treatment of Anger	Trait Anxiety
1	1	1	70
2	1	1	80
3	1	1	50
4	1	0	60
5	1	0	40
6	1	0	65
7	1	0	75
8	1	0	80
9	1	0	70
10	1	0	60
11	0	1	65
12	0	1	50
13	0	1	45
14	0	1	35
15	0	1	40
16	0	1	50
17	0	0	55
18	0	0	45
19	0	0	50
20	0	0	60

Step 1:

	Anger Treatment		
Heart Attack	Yes (1)	No (0)	Total
Yes (1)	3 (a)	7 (b)	10 (a+b)
No (0)	6 (c)	4 (d)	10 (c+d)
Total	9 (a+c)	11 (b+d)	20 (a+b+c+d)

...EXAMPLE

Step 2:

$$P_0 = P(HA = 1 | Anger = 0) = \frac{e^{(b_0)}}{(1 + e^{(b_0)})} = \frac{P}{1 - P} = \frac{7/20}{11/20}$$

According to maximum likelihood

$$\frac{7}{4} = e^{(b_0)}$$

Take natural log on both sides

$$\ln e^{(b_0)} = \ln \frac{7}{4}$$

$$b_0 = 0.559$$

Step 3:

$$P_1 = P(HA = 1 | Anger = 1) = \frac{e^{(b_0 + b_1 X)}}{(1 + e^{(b_0 + b_1 X)})} = \frac{P}{1 - P} = \frac{3/20}{9/20}$$

$$\frac{3}{6} = e^{(b_0 + b_1 X)}$$

...EXAMPLE

Put $b_0 = 0.559$

$$\frac{3}{6} = e^{(b_0 + b_1 X)}$$

$$\frac{3}{6} = e^{(0.559 + b_1 X)}$$

Take natural log on both sides

$$\ln(0.5) = 0.559 + b_1 X$$

Since, $X=1$

$$\ln(0.5) = 0.559 + b_1$$

$$-0.693 = 0.559 + b_1$$

$$b_1 = -1.25276$$

Logistic Regression Equation:

$$y = 0.559 + (-1.25276)X$$

...EXAMPLE

Now we can compute the odds of having a heart attack for the treatment group and the no treatment group.

For the treatment group, the odds are $3/6 = 1/2$.

The odds for the no treatment group are $7/4$ or 1.75 .

$$\begin{aligned} \text{Odd Ratio}(\text{treatment vs non treatment})(OR) &= \frac{a/c}{b/d} \\ &= \frac{ad}{bc} = \frac{1.75}{0.5} = 3.5 \end{aligned}$$

$$\text{Now, } 1/1 + e^{-b} = OR$$

Take \ln on both sides (natural log)

$$b = \ln OR$$

$$b = 1.2528$$

GRADIENT DESCENT

GRADIENT DESCENT

Gradient Descent

Gradient descent is an **optimization algorithm** used to **find the values of parameters (coefficients)** of a function f **that minimizes a cost function** (cost).

Gradient descent is best used **when the parameters cannot be calculated analytically** (e.g. using linear algebra) and must be searched for by an optimization algorithm.

GRADIENT DESCENT

Let say y is the actual value **$y = \text{actual value}$**

Let say \hat{y} is the predicted value **$\hat{y} = \text{predicted value} = mx + b$**

Now, ***error = predicted value – actual value***

We have to minimize the error to get the correct predicted value. To do so, we have to minimize the following equation

$$\text{Cost} = \sum_{i=1}^n (\hat{y}_i - y_i)^2$$

The above equation is known as Cost function or in general the Loss function

...GRADIENT DESCENT

$$Cost = \sum_{i=1}^n (\hat{y}_i - y_i)^2 \quad // \text{Cost Function}$$

Or we can say

We want to know in what way we can made change in value of m in $y = mx + b$ in order to make the error less.

$$m = m + \Delta m$$

$$b = b + \Delta b$$

So, our objective is to find m and b values such that *cost function is minimum* i.e the error is minimum and to do that we need to define the derivative of the cost function

...GRADIENT DESCENT

Let say the cost function is J which is a function of m and b . Hence we can write it as $J_{(m,b)}$.

Step 1:

$$J_{(m,b)} = \text{Error}^2 \quad // \text{ Cost} = \text{Error}^2$$

Step 2:

Apply Power Rule and Chain Rule

Why Chain Rule?

Since, Cost function depends on Error and Error depends on m and b **i.e, J is a function of Error and Error is a function on m and b .**

...GRADIENT DESCENT

To find in which direction the minimum is and how big or small the step size should be, we find the derivative of the cost function.

To find the derivatives , we are going to apply

- **Power Rule**

If

$$y = x^n$$

Then

$$\frac{\partial y}{\partial x} = nx^{n-1}$$

- **Chain Rule**

If

$$y = x^2 \text{ and } x = z^2,$$

Then

$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial z} * \frac{\partial z}{\partial x}$$

...GRADIENT DESCENT

We know that $\text{Error} = \hat{y} - y$

Also, $\text{Error} = mx + b - y$

To find Δm *// How error changes wrt to m*

$$\Delta m = \frac{\partial \text{Error}}{\partial m} = x$$

To find Δb *// How error changes wrt to b*

$$\Delta b = \frac{\partial \text{Error}}{\partial b} = 1$$

...GRADIENT DESCENT

Can be written as

$$Cost = Error^2$$

To find how J changes when m changes, we need to find $\frac{\partial J}{\partial m}$

Apply power and Chain Rule

$$\frac{\partial J}{\partial m} = 2 * Error * \frac{\partial Error}{\partial m} * Learning_Rate$$

$$\frac{\partial J}{\partial m} = 2 * Error * x * Learning_Rate$$

Learning Rate is a tuning parameter used to determine the step size at each iteration while moving towards minimum of a loss function.

Can be written as

$$\frac{\partial J}{\partial m} = Error * x * Learning_Rate$$

$$m = m + Error * x * Learning_Rate$$

...GRADIENT DESCENT

Similarly

$$Cost = Error^2$$

To find how J changes when m changes, we need to find $\frac{\partial J}{\partial m}$

Apply power and Chain Rule

$$\frac{\partial J}{\partial b} = 2 * Error * \frac{\partial Error}{\partial b} * Learning_Rate$$

$$\frac{\partial J}{\partial b} = 2 * Error * 1 * Learning_Rate$$

Can be written as

$$\frac{\partial J}{\partial b} = Error * 1 * Learning_Rate$$

$$b = b + error * Learning_Rate$$

...GRADIENT DESCENT FUNCTION

```
Function GradientDescent()  
{  
    var learning_rate=0.05  
    for (var i=1=0; i<data.length; i++)  
        var x = data[i].x;  
        var y = data[y].y;  
        var y_pred = m * x + b;  
        var error = y_pred - y;  
        m = m + error * x * learning_rate;  
        b = b + error * learning_rate;  
}
```

Linear Regression

Hypothesis: $h_{\theta}(x) = \theta_0 + \theta_1 x$

Parameters: θ_0, θ_1

Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$

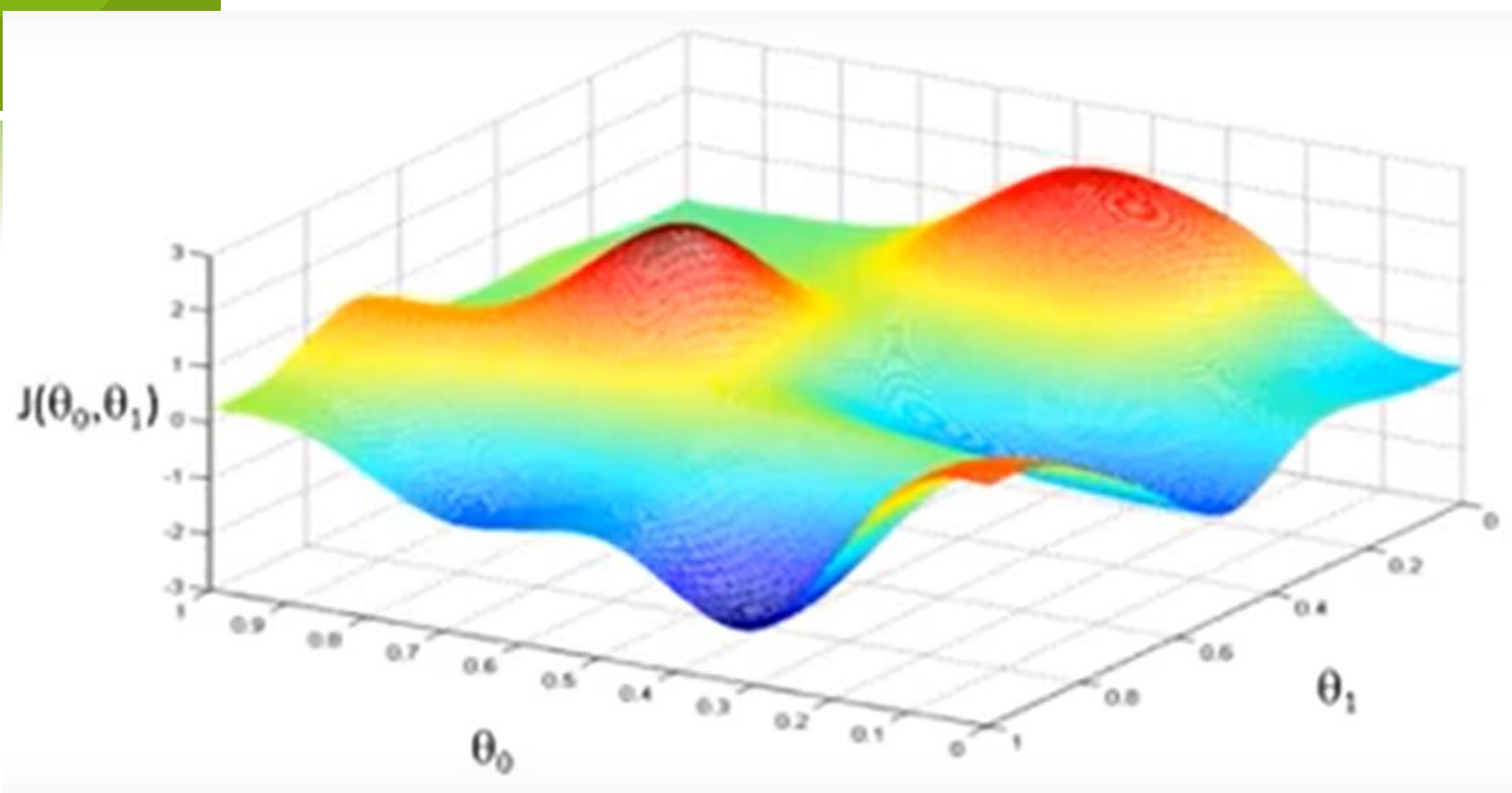
Goal: $\underset{\theta_0, \theta_1}{\text{minimize}} J(\theta_0, \theta_1)$

Have some function $J(\theta_0, \theta_1)$

Want $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Outline:

- Start with some θ_0, θ_1
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$
until we hopefully end up at a minimum



Gradient descent algorithm

repeat until convergence {
 $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ (for $j = 0$ and $j = 1$)
}

Learning rate

Correct: Simultaneous update

```
temp0 :=  $\theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$   
temp1 :=  $\theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$   
 $\theta_0 :=$  temp0  
 $\theta_1 :=$  temp1
```

Incorrect:

```
temp0 :=  $\theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$   
 $\theta_0 :=$  temp0  
temp1 :=  $\theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$   
 $\theta_1 :=$  temp1
```

Note: $:=$ is Assignment operator

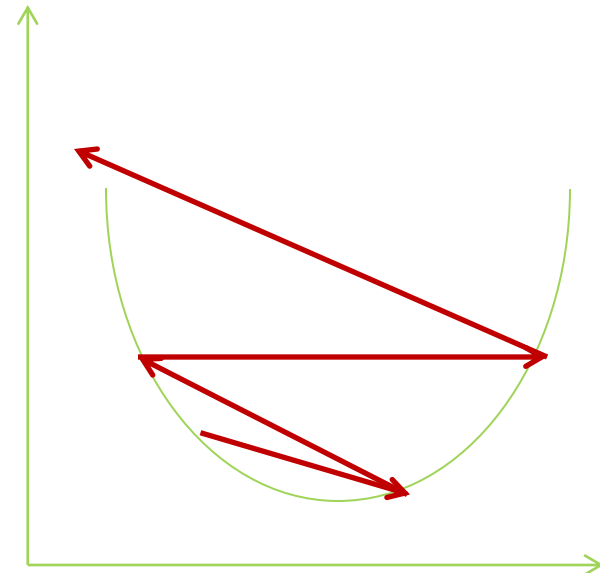
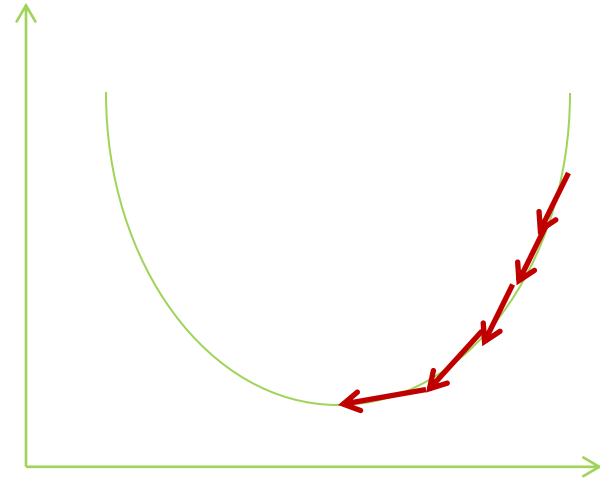
Learning Parameter (α)

Lets consider only 1 parameter, θ_1

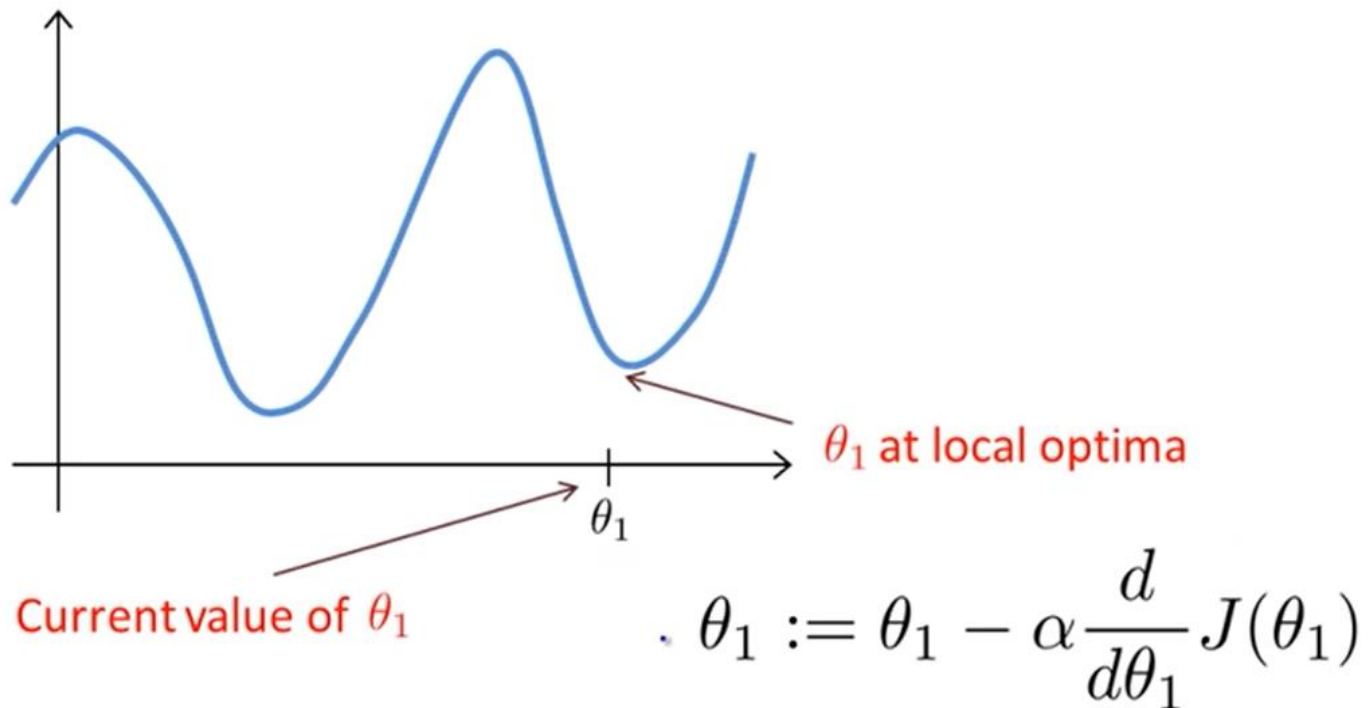
$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

If α is too small, gradient descent can be slow.

If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.



Problem of Local Minima



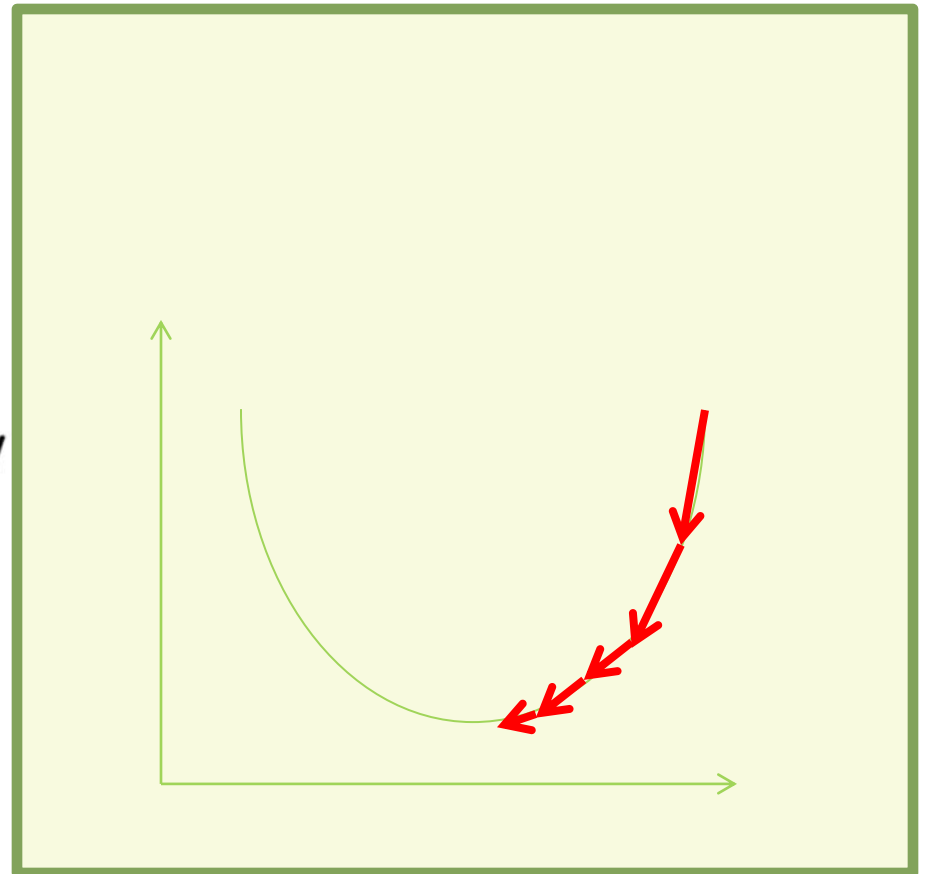
Derivative will be 0 at local minima

$\theta_1 := \theta_1$

Gradient descent can converge to a local minimum, even with the learning rate α fixed.

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

As we approach a local minimum, gradient descent will automatically take smaller steps. So, no need to decrease α over time.



GRADIENT DESCENT VS LINEAR REGRESSION

Gradient descent algorithm

repeat until convergence {
 $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$
 (for $j = 1$ and $j = 0$)
}

Linear Regression Model

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

BATCH GRADIENT DESCENT

- ◎ The goal of all supervised machine learning algorithms is to best estimate a target function (f) that maps input data (X) onto output variables (Y).
- ◎ Some machine learning algorithms have coefficients that characterize the algorithms estimate for the target function (f).
- ◎ Different algorithms have different representations and different coefficients, but many of them require a process of optimization to find the set of coefficients that result in the best estimate of the target function.
- ◎ Common examples of algorithms with coefficients that can be optimized using gradient descent are Linear Regression and Logistic Regression.
- ◎ The cost is calculated for a machine learning algorithm over the entire training dataset for each iteration of the gradient descent algorithm.
- ◎ **One iteration of the algorithm is called one batch and this form of gradient descent is referred to as batch gradient descent.**
- ◎ Batch gradient descent is the most common form of gradient descent described in machine learning.

STOCHASTIC GRADIENT DESCENT

- ⊙ Gradient descent can be slow to run on very large datasets.
- ⊙ In situations when you have large amounts of data, you can use a variation of gradient descent called stochastic gradient descent.
- ⊙ In this variation, the gradient descent procedure described above is run but the update to the coefficients is performed for each training instance, rather than at the end of the batch of instances.

TIPS FOR GRADIENT DESCENT

- ◎ **Plot Cost versus Time:** Collect and plot the cost values calculated by the algorithm each iteration. The expectation for a well performing gradient descent run is a decrease in cost each iteration. If it does not decrease, try reducing your learning rate.
- ◎ **Learning Rate:** The learning rate value is a small real value such as 0.1, 0.001 or 0.0001. Try different values for your problem and see which works best.
- ◎ **Rescale Inputs:** The algorithm will reach the minimum cost faster if the shape of the cost function is not skewed and distorted. You can achieve this by rescaling all of the input variables (X) to the same range, such as between 0 and 1.
- ◎ **Few Passes:** Stochastic gradient descent often does not need more than 1-to-10 passes through the training dataset to converge on good or good enough coefficients.
- ◎ **Plot Mean Cost:** The updates for each training dataset instance can result in a noisy plot of cost over time when using stochastic gradient descent. Taking the average over 10, 100, or 1000 updates can give you a better idea of the learning trend for the algorithm.

DIMENSIONALITY REDUCTION AND FEATURE SELECTION

DIMENSIONALITY REDUCTION

- ❑ In machine learning classification problems, there are often too many factors on the basis of which the final classification is done.
- ❑ These factors are basically variables called features.
- ❑ The higher the number of features, the harder it gets to visualize the training set and then work on it.
- ❑ Sometimes, most of these features are correlated, and hence redundant.
- ❑ Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables.

DIMENSIONALITY REDUCTION

□ Example

❖ Email: Spam or not Spam

- Features: whether or not the e-mail has a generic title, the content of the e-mail, whether the e-mail uses a template, etc.

❖ Classification:

- Dataset Features: Temperature, Humidity, Rainfall etc..

Dimensionality reduction has two main components:

□ **Feature selection :**

- Find a subset of the original set of variables, or features, to get a smaller subset which can be used to model the problem.

□ **Feature extraction/scaling:**

- This reduces the data in a high dimensional space to a lower dimension space, i.e. a space with lesser no. of dimensions.

FEATURE SELECTION

- ⊙ Higher dimensional data sets increase the time complexity and also the space required will be more.
- ⊙ Also, all the features in the dataset might not be useful.
- ⊙ Some may contribute no information at all, while some may contribute similar information as the other features.
- ⊙ Selecting the optimal set of features will help us hence reduce the space and time complexity as well as increase the accuracy or purity of classification (or regression) and clustering (or association) for supervised and unsupervised learning respectively.

FEATURE SELECTION

- ❑ Also called Variable Selection and Attribute Selection.
- ❑ Task of selecting a subset of relevant features for use in model construction
- ❑ It is used to identify and remove unneeded, irrelevant and redundant attributes from data.
- ❑ It has Four approaches
 - Wrapper method
 - Filter Method
 - Embedded Method
 - Hybrid Method

FEATURE SELECTION APPROACHES

❑ Wrapper method

- It considers the selection of a set of features as a search problem, where different combinations are prepared, evaluated and compared to other combinations.
- A predictive model is used to evaluate a combination of features and assign a score based on model accuracy.
- Eg: Best Fit Search and Random Hill Climbing

❑ Filter Method

- Applies a statistical measure to assign a scoring to each feature.
- Features are ranked by the score and either selected or rejected from the dataset.
- Eg: Correlation Coefficient Scores, Chi Square Test

FEATURE SELECTION APPROACHES

❑ Embedded method

- The applied learning algorithms determine the specificity of this approach
- It selects the features during the process of training the data set.

❑ Hybrid Method

- Both filter and wrapper-based methods are used in hybrid approach.
- This approach first selects the possible optimal feature set which is further tested by the wrapper approach.
- It hence uses the advantages of both filter and wrapper-based approach.

PARAMETERS FOR FEATURE SELECTION

The parameters are classified based on two factors:

- ❑ The Similarity of information contributed by the features :
 - ⊙ Correlation
- ❑ Quantum of information contributed by the features :
 - ⊙ Entropy
 - ⊙ Mutual Information

PARAMETERS FOR FEATURE SELECTION

❑ Correlation

- ❖ Features may be correlated and may have some correlation factor.
- ❖ If f1 and f2 are two correlated features of a data set, then the classifying or regression model including both f1 and f2 will give the same as the predictive model compared to the scenario where either f1 or f2 was included in the dataset.
- ❖ This is because both f1 and f2 are correlated Only one representative can be considered
- ❖ Method of finding Correlation: Pearson's correlation coefficient(ρ)

$$\rho(X, Y) = \frac{Cov(X, Y)}{\sigma_X \sigma_Y}$$

where ,

cov(X, Y) - covariance

$\sigma(X)$ - standard deviation of X

$\sigma(Y)$ - standard deviation of Y

PARAMETERS FOR FEATURE SELECTION

□ Entropy

- It is the measure of the average information content.
- The higher the entropy, the higher is the information contribution by that feature. Entropy (H) can be formulated as:

$$H(x) = E[I(X)] = E[-\ln P(X)]$$

Where, X - discrete random variable X
P(X) - probability mass function
E - expected value operator,
I - information content of X.
I(X) - a random variable.

- To calculate Entropy of feature f_i : Exclude f_i and calculate entropy for rest of the features.
- If Entropy is low, then the information by feature f_i is high.
- Entropy is mostly used for Unsupervised Learning

PARAMETERS FOR FEATURE SELECTION

❑ Mutual Information

- Amount of uncertainty in X due to the knowledge of Y.

- It is calculated as:

$$I(X, Y) = \sum_{y=Y} \sum_{x=X} P(x, y) \log\left(\frac{P(x, y)}{P(x)P(y)}\right)$$

where

$p(x, y)$ - joint probability function of X and Y,

$p(x)$ - marginal probability distribution function of X

$p(y)$ - marginal probability distribution function of Y

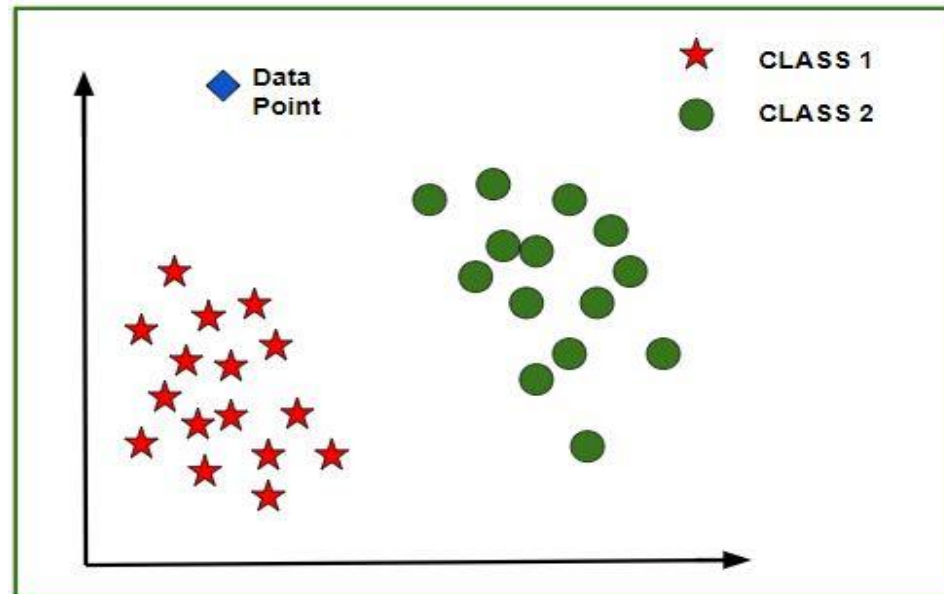
- Calculated to know the amount of information shared about the class by a feature.
- Mostly used for dimensionality reduction in Supervised Learning

FEATURE SCALING

- ① Feature Scaling is a technique to standardize the independent features present in the data in a fixed range.
- ② It is performed during the data pre-processing.
- ③ Feature Scaling Algorithms will scale Age, Salary, BHK in fixed range say $[-1, 1]$ or $[0, 1]$.
- ④ And then no feature can dominate other.
- ⑤ If feature scaling is not done, then a machine learning algorithm tends to weigh greater values, higher and consider smaller values as the lower values, regardless of the unit of the values.

WITHOUT FEATURE SCALING

- ◎ Eg: Classes: Yes or No
 - Prediction of the class of new data point:
 - The model calculates the distance of this data point from the centroid of each class group.
 - Finally this data point will belong to that class, which will have a minimum centroid distance from it.



WITHOUT FEATURE SCALING: DISTANCE MEASURE

- **Euclidean Distance** : It is the square-root of the sum of squares of differences between the coordinates (feature values – Age, Salary, BHK Apartment) of data point and centroid of each class. This formula is given by Pythagorean theorem.

$$d(x, y) = \sqrt[r]{\sum_{k=1}^n (x_k - y_k)^r}$$

where x is Data Point value, y is Centroid value and k is no. of feature values,
Example: given data set has k = 3

- **Manhattan Distance** : It is calculated as the sum of absolute differences between the coordinates (feature values) of data point and centroid of each class.

$$d(x, y) = \sum_{k=1}^n |x_k - y_k|$$

- **Minkowski Distance** : It is a generalization of above two methods. As shown in the figure, different values can be used for finding r.

$$d(x, y) = \sqrt[r]{\sum_{k=1}^n (x_k - y_k)^r}$$

FEATURE SCALING

◎ **Example:**

- If an algorithm is not using feature scaling method then it can consider the value 3000 meter to be greater than 5 km but that's actually not true and in this case, the algorithm will give wrong predictions.
- So, we use Feature Scaling to bring all values to same magnitudes and thus, tackle this issue.

TECHNIQUES FOR FEATURE SCALING

- **Min-Max Normalisation:** This technique re-scales a feature or observation value with distribution value between 0 and 1.

$$X_{\text{new}} = \frac{X_i - \min(X)}{\max(x) - \min(X)}$$

- **Standardisation:** It is a very effective technique which re-scales a feature value so that it has distribution with 0 mean value and variance equals to 1.

$$X_{\text{new}} = \frac{X_i - X_{\text{mean}}}{\text{Standard Deviation}}$$

METHODS OF DIMENSIONALITY REDUCTION

- ◎ Dimensionality reduction may be both linear or non-linear, depending upon the method used.
- ◎ The various methods used for dimensionality reduction include:
 - Principal Component Analysis (PCA)
 - Linear Discriminant Analysis (LDA)
 - Generalized Discriminant Analysis (GDA)
- Most Common Linear Method:
 - Principal Component Analysis (PCA)

PRINCIPLE COMPONENT ANALYSIS

PRINCIPLE COMPONENT ANALYSIS

- ◎ This method was introduced by Karl Pearson.
- ◎ Principal Component Analysis (PCA) is a dimension-reduction tool that can be used to reduce a large set of variables to a small set that still contains most of the information in the large set.
- ◎ It is a mathematical procedure that transforms a number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called principal components.
- ◎ The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible.

GIST OF PCA

$$X_{n*m} = \begin{bmatrix} & \end{bmatrix}$$

X=Original data

n= number of samples

m=number of measurements/ attributes

Eigen decomposition of covariance matrix

$$C_{m*m} = X^T X \rightarrow W$$

where, W= Eigen Vector Matrix

$$T_{n*m} = XW$$

- T=Truncated matrix , also called as Scores which gives the transformed way for looking at the data.
- X= Original data
- W= Eigen Vector Matrix Columns of w is called loadings or Eigen Vectors. Each Column of W is a Principal Component. W is an ordered matrix by the value of greater eigen value (λ)

...GIST OF PCA

$W_{m \times r}$ = Select first r columns (i.e first r Principal Components) of W .

$$T_{n \times r} = X_{n \times m} W_{m \times r}$$

$T=r$ -dimensional representation of same data, where r are the best selected principal components (Principal Component Space).

Note: PCA is NOT actually changing the data in anyway. Its is just the different way of looking at the data.

PCA IMPLEMENTATION

- ◎ Step 1: Normalize the data

Eg: Try to produces a dataset whose mean is zero

- ◎ Step 2: Calculate the covariance matrix

	x	y
x	Cov(x,x)	Cov(x,y)
y	Cov(y,x)	Cov(y,y)

- ◎ Step 3: Calculate the eigenvalues and eigenvectors

- ◎ Step 4: Choosing components and forming a feature vector:

- Order the eigenvalues from largest to smallest and select n principle components from the top
- Form a Feature Vector: only those eigenvectors which will be considered

- Step 5: Forming Principal Components:

- $NewData = FeatureVector^T \times ScaledData^T$
- $NewData$ = Matrix consisting of the principal components

PCA EXAMPLE

- ◎ **Step 1:** Find the mean value of each attribute.

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

- Mean Values:

$$x' = 1.81$$

$$y' = 1.91$$

PCA EXAMPLE

- ◎ **Step 2:** Subtract the mean to make the data pass through the origin.

$(x' - x)$	$(y' - y)$
-0.69	-0.49
1.31	1.21
-0.39	-0.99
-0.09	-0.29
-1.29	-1.09
-0.49	-0.79
-0.19	0.31
0.81	0.81
0.31	0.31
0.71	1.01

The normalized data will have mean=0

Step 3: Find the Co-Variance Matrix: It shows how two variable vary together

$$Co - Variance = \frac{(x - x')(y - y')}{n - 1}$$

PCA EXAMPLE

The Co-variance matrix for example is:

$$\begin{pmatrix} 0.616 & 0.6154 \\ 0.615 & 0.716 \end{pmatrix}$$

Since, the non-diagonal elements in covariance matrix are positive, Thus, x and y variable increase together in one direction.

Step 4: Calculate Eigen Vales and Eigen Vectors for covariance matrix"

$$\text{eigen values} = \begin{pmatrix} 0.4908 \\ 1.25402 \end{pmatrix} \quad \text{eigen vectors} \begin{pmatrix} -0.735 & -0.678 \\ 0.677 & -0.731 \end{pmatrix}$$

The most important (principle) Eigen vector would have the direction in which the variables strongly correlate.

PCA EXAMPLE

- **Step 5:** The Eigen vectors with highest Eigen value will be selected for PCA.
- Now we can ignore the other dimensions,
- For n dimensions of data $\rightarrow n$ Eigen vectors \rightarrow select p Eigen vectors.
- For dimensionality reduction $p < n$
$$\text{Final data} = \text{FeatureVector}^T \times \text{ScaledData}^T$$
- Final data is the final dataset, with data items in columns, and dimensions along rows.
- Example data has 2 dimensions so data was in terms of x and y . Now the data will be in the terms of eigen vectors.

PROS AND CONS OF DIMENSIONALITY REDUCTION

Advantages of Dimensionality Reduction

- ◎ It helps in data compression, and hence reduced storage space.
- ◎ It reduces computation time.
- ◎ It also helps remove redundant features, if any.

Disadvantages of Dimensionality Reduction

- ◎ It may lead to some amount of data loss.
- ◎ PCA tends to find linear correlations between variables, which is sometimes undesirable.
- ◎ PCA fails in cases where mean and covariance are not enough to define datasets.
- ◎ We may not know how many principal components to keep- in practice, some thumb rules are applied.

REGULARIZATION

UNDER-FITTING AND OVER-FITTING

Example:

- ⊙ We have 10 students in a classroom.
- ⊙ We intend to **train a model based on their past score to predict their future score.**
- ⊙ There are 5 females and 5 males in the class.
- ⊙ The average score of females is 60 whereas that of males is 80.
- ⊙ The overall average of the class is 70.

.... UNDER-FITTING AND OVER-FITTING

Ways of making prediction

- ⊙ Predict the score as 70 for the entire class
- ⊙ Predict score of males = 80 and females = 60. This a simplistic model which might give a better estimate than the first one.
- ⊙ We can use the roll number of students to make a prediction and say that every student will exactly score same marks as last time. Now, this is unlikely to be true and we have reached such granular level that we can go seriously wrong.

.... UNDER-FITTING AND OVER-FITTING

◎ Under Fit

Predict the score as 70 for the entire class

◎ Optimum Fit

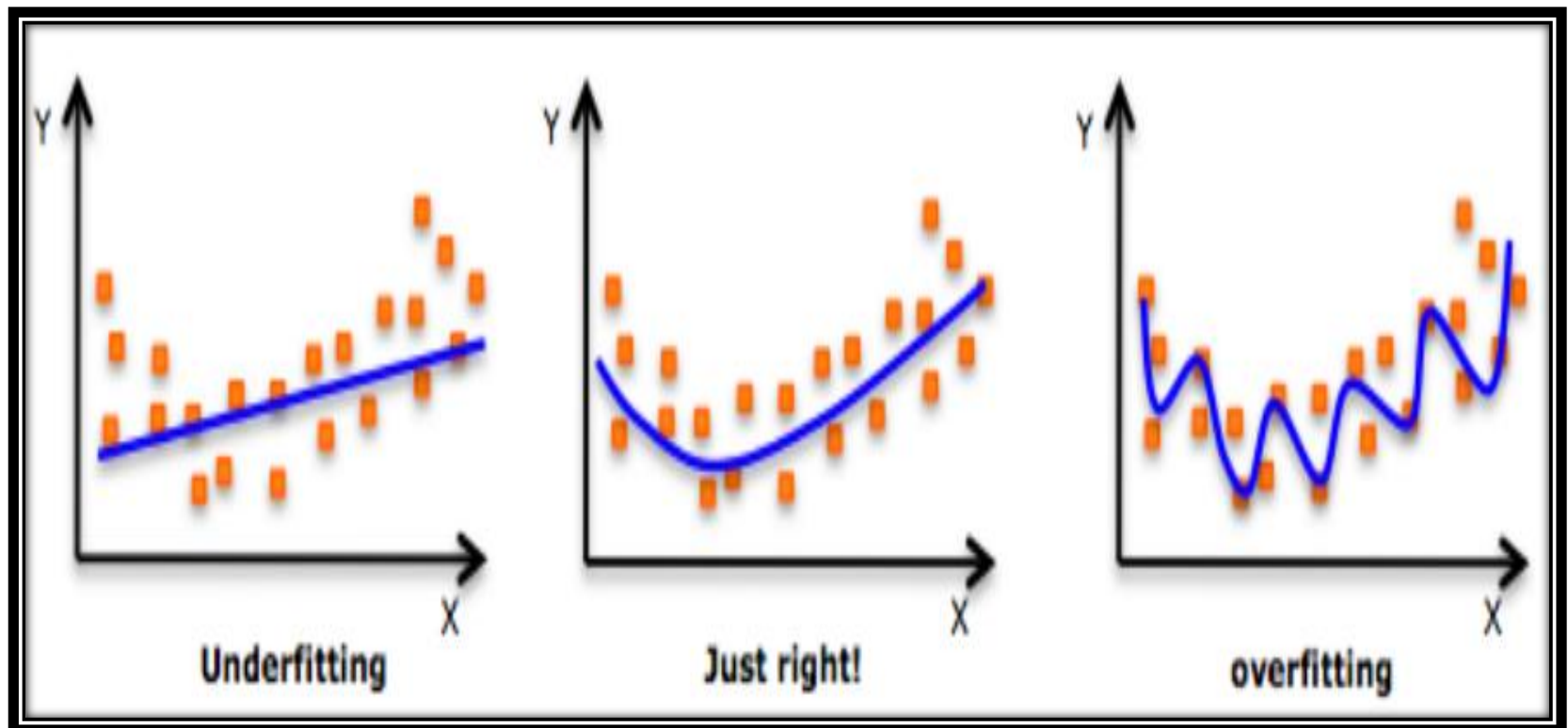
Predict score of males = 80 and females = 60. This a simplistic model which might give a better estimate than the first one.

◎ Over Fit

We can use the roll number of students to make a prediction and say that every student will exactly score same marks as last time. Now, this is unlikely to be true and we have reached such granular level that we can go seriously wrong.

.... UNDER-FITTING AND OVER-FITTING

- ◎ In the world of analytics, where we try to fit a curve to every pattern, Over-fitting and under-fitting are biggest concerns.



.... UNDER-FITTING AND OVER-FITTING

- ◎ Over-fitting is caused by a hypothesis function that fits **the available data** but does **not generalize well to predict new data**.
- ◎ An over-fitted model performs well on training data but fails to generalize.
- ◎ Goal of our machine learning algorithm is **to learn the data patterns and ignore the noise in the data set**.

.... UNDER-FITTING AND OVER-FITTING

METHODS TO AVOID OVER-FITTING:

- ◎ **Early Stopping** : Early stopping rules provide guidance as to **how many iterations can be run** before the learner begins to overfit.
- ◎ **Cross-Validation** : we **train our model using the subset of the data-set** and then evaluate using the complementary subset of the data-set.
- ◎ **Pruning** : It simply **removes the nodes which add little predictive power for the problem** in hand.
- ◎ **Regularization** : It introduces a **cost term for bringing in more features** with the objective function. It tries to push the coefficients for many variables to zero and hence reduce cost term.

OVER-FITTING MORE FORMALLY

- ⊙ Assume that the data is drawn from **some fixed, unknown probability distribution**
- ⊙ Every hypothesis has a "**true**" **error** $J^*(h)$, which is the expected error when data is drawn from the distribution.
- ⊙ Because we do **not have all the data**, we **measure the error on the training set** $J_D(h)$
- ⊙ Suppose we **compare hypotheses** h_1 and h_2 on the training set, and $J_D(h_1) < J_D(h_2)$.
- ⊙ If h_2 is "truly" better, i.e. $J^*(h_2) < J^*(h_1)$, our algorithm is **over-fitting**.

REGULARIZATION

Regularization can be motivated as a technique to **improve the generalizability** of a learned model.

The **goal of this learning problem** is to **find a function** that fits or predicts the outcome (label) **that minimizes the expected error over all possible inputs and labels.**

Regularization is a technique **used for tuning the function by adding an additional penalty term in the error function.** The additional term controls the excessively fluctuating function such that the coefficients don't take extreme values.

REGULARIZATION

Types of Regularization:

- **Ridge Regression (L2-norm)**
- **Lasso Regression (L1-norm)**

REGULARIZATION

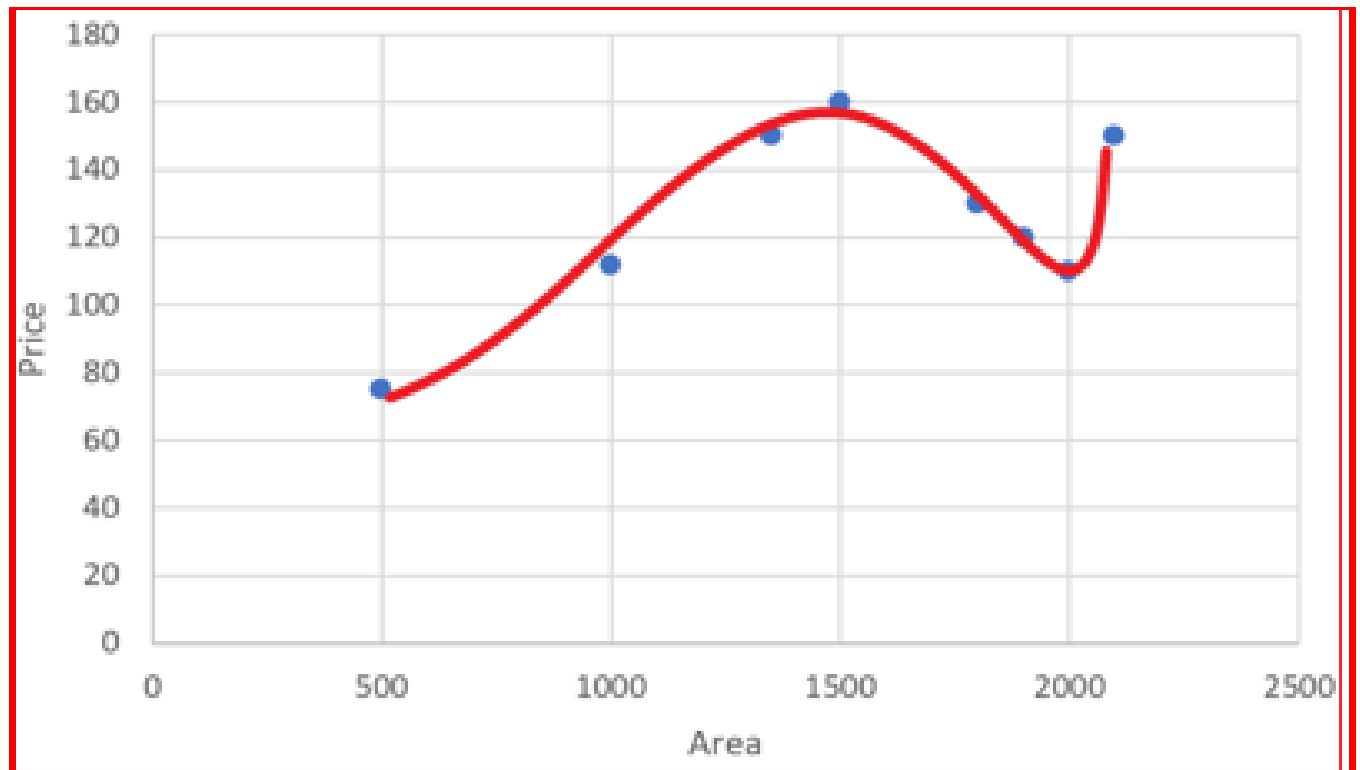
- ⊙ Regularization is a technique to **discourage the complexity of the model**.
- ⊙ It does this by **penalizing the loss function**.
- ⊙ This helps to solve the over-fitting problem.
- ⊙ **Loss function is the sum of squared difference between the actual value and the predicted value.**

$$L(x, y) = \sum_{i=1}^n (y_i - f(x_i))^2$$

$$f(x_i) = h_{\theta}x = \theta_0 + \theta_1x_1 + \theta_2x_2^2 + \theta_3x_3^3 + \theta_4x_4^4$$

...REGULARIZATION

- ⊙ As the degree of the input features increases the model becomes complex and **tries to fit all the data points as shown below**



...REGULARIZATION

- ⊙ When we penalize the weights θ_3 and θ_4 and make them **too small, very close to zero**.
- ⊙ It makes those **terms negligible and helps simplify the model**.

$$f(x_i) = h_{\theta}x = \theta_0 + \theta_1x_1 + \theta_2x_2^2 + \theta_3x_3^3 + \theta_4x_4^4$$

$$f(x_i) = h_{\theta}x = \theta_0 + \theta_1x_1 + \theta_2x_2^2$$

Note: Regularization works on assumption that smaller weights generate simpler model and thus helps avoid overfitting.

...REGULARIZATION

What if the input variables have an impact on the output?

- ◎ To ensure we take into account the input variables, **we penalize all the weights by making them small.**
- ◎ This also makes the model simpler and less prone to over-fitting.

$$L(x, y) = \sum_{i=1}^n (y_i - h_{\theta}(x_i))^2$$

where $h_{\theta}x_i = \theta_0 + \theta_1x_1 + \theta_2x_2^2 + \theta_3x_3^3 + \theta_4x_4^4$

$$L(x, y) \equiv \sum_{i=1}^n (y_i - h_{\theta}(x_i))^2 + \lambda \sum_{i=1}^n \theta_i^2$$

Loss function with
regularization
term

..... REGULARIZATION

- ◎ We have **added the regularization term to the sum of squared differences** between the actual value and predicted value.
- ◎ Regularization term **keeps the weights small making the model simpler and avoiding over-fitting.**
- ◎ λ is the penalty term or **regularization parameter which determines how much to penalizes the weights.**
- ◎ When λ **is zero then the regularization term becomes zero.** We are back to the original Loss function.

$$L(x, y) = \sum_{i=1}^n (y_i - h_{\theta}(x_i))^2 + 0$$

when $\lambda=0$

..... REGULARIZATION

- ⊙ When λ is large, we penalizes the weights and they become close to zero.
- ⊙ This results is a very simple model having a high bias or is under-fitting.

$$h_{\theta}x = \theta_0 + \theta_1 \times c_1 + \theta_2 \times c_2^2 + \theta_3 \times c_3^3 + \theta_4 \times c_4^4$$

What is the right value for λ ?

- ⊙ It is somewhere in between **0 and a large value**. we need to find an optimal value of λ so that the generalization error is small.
- ⊙ A simple approach would be **try different values of λ on a subsample of data**, understand variability of the loss function and then use it on the entire dataset

L1 REGULARIZATION OR LASSO OR L1 NORM

- ◎ In L1 norm we **shrink the parameters to zero.**
- ◎ When input features have **weights closer to zero that leads to sparse L1 norm.**
- ◎ In Sparse solution **majority of the input features have zero weights and very few features have non zero weights.**
- ◎ L1 regularization **does feature selection.**
- ◎ It does this by **assigning insignificant input features with zero weight and useful features with a non zero weight.**

L1 REGULARIZATION OR LASSO OR L1 NORM

- ◎ It does this by **assigning insignificant input features with zero weight and useful features with a non zero weight.**
- ◎ In L1 regularization **we penalize the absolute value of the weights.** L1 regularization term is highlighted in the red box.
- ◎ Lasso produces a model that is **simple, interpretable** and **contains a subset of input features.**

$$L(x, y) \equiv \sum_{i=1}^n (y_i - h_{\theta}(x_i))^2 + \lambda \sum_{i=1}^n |\theta_i|$$

L2 REGULARIZATION OR RIDGE OR L2-NORM

- ◎ In L2 regularization, **regularization term is the sum of square of all feature weights** as shown below in the equation.
- ◎ L2 regularization **forces the weights to be small but does not make them zero** and does non sparse solution.
- ◎ L2 is **not robust to outliers** as **square terms blows up the error differences of the outliers** and the regularization term tries to fix it by penalizing the weights

L2 REGULARIZATION OR RIDGE OR L2-NORM

- ◎ Ridge regression performs better **when all the input features influence the output and all with weights are of roughly equal size**

$$L(x, y) \equiv \sum_{i=1}^n (y_i - h_{\theta}(x_i))^2 + \lambda \sum_{i=1}^n \theta_i^2$$

L1 NORM VS L2 NORM

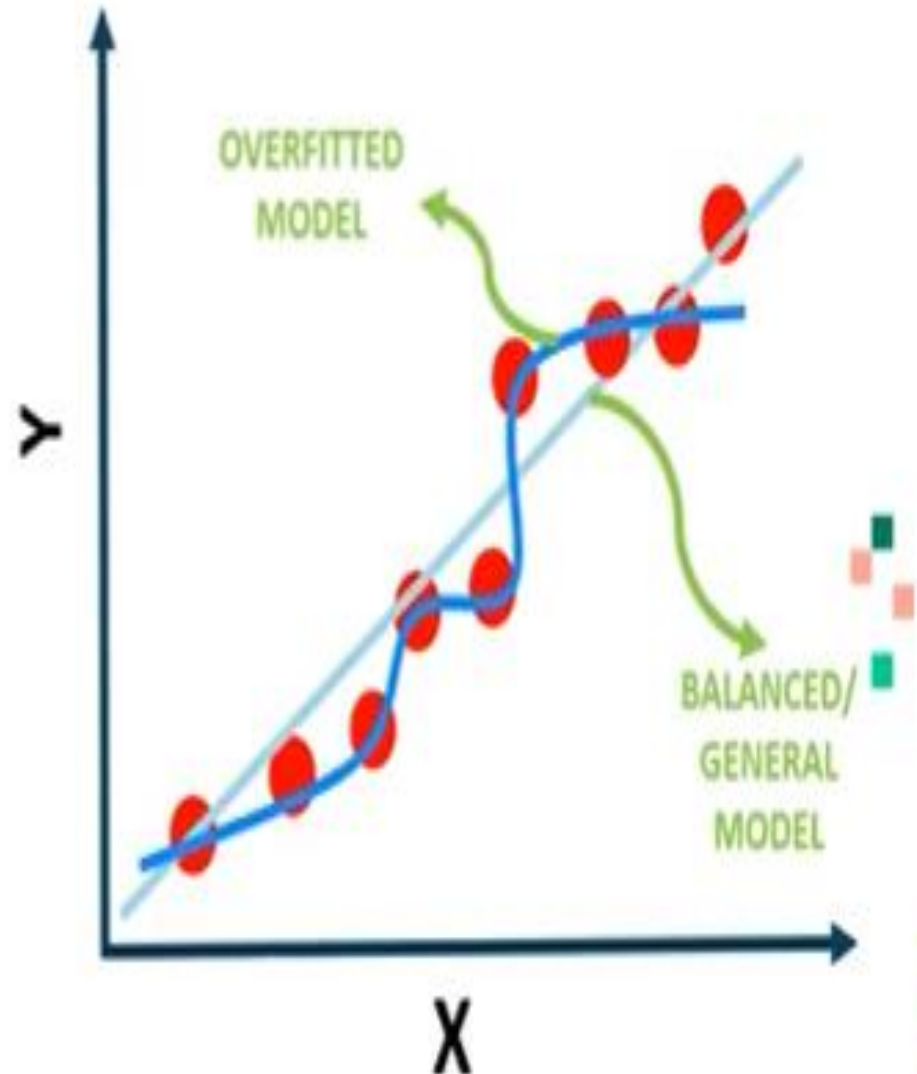
L1 Norm	L2 Norm
L1 penalizes sum of absolute value of weights.	L2 regularization penalizes sum of square weights.
L1 has a sparse solution	L2 has a non sparse solution
L1 has multiple solutions	L2 has one solution
L1 has built in feature selection	L2 has no feature selection
L1 is robust to outliers	L2 is not robust to outliers
L1 generates model that are simple and interpretable but cannot learn complex patterns	L2 regularization is able to learn complex data patterns

Observation: L2 gives better prediction when output variable is a function of all input features

RIDGE REGRESSION OR L2 REGRESSION OR L2-NORM

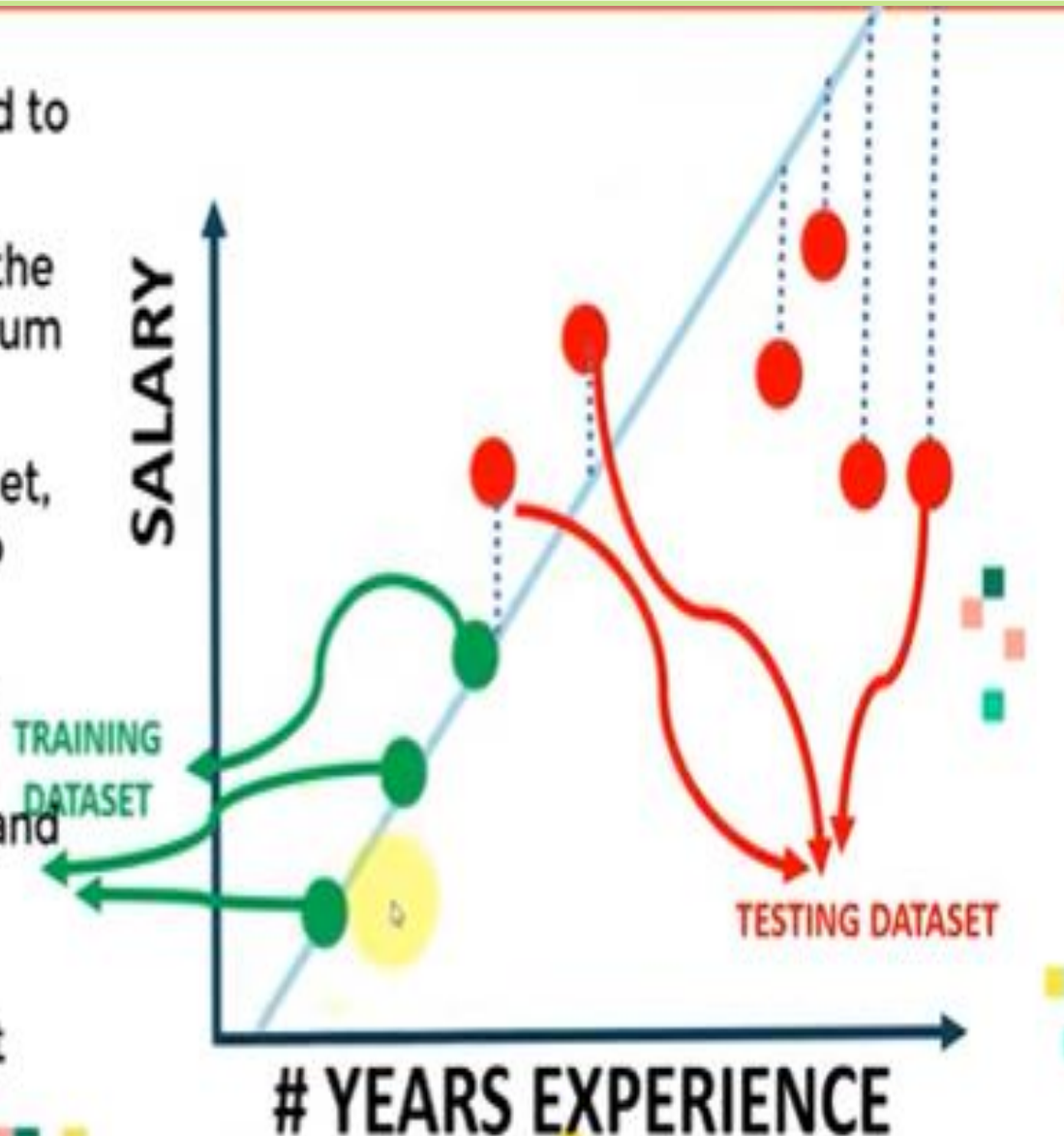
RIDGE REGRESSION (L2 REGRESSION): INTUITION

- Ridge regression advantage is to avoid overfitting.
- Our ultimate model is the one that could generalize patterns; i.e.: works best on the training and testing dataset
- Overfitting occurs when the trained model performs well on the training data and performs poorly on the testing datasets
- Ridge regression works by applying a penalizing term (reducing the weights and biases) to overcome overfitting.



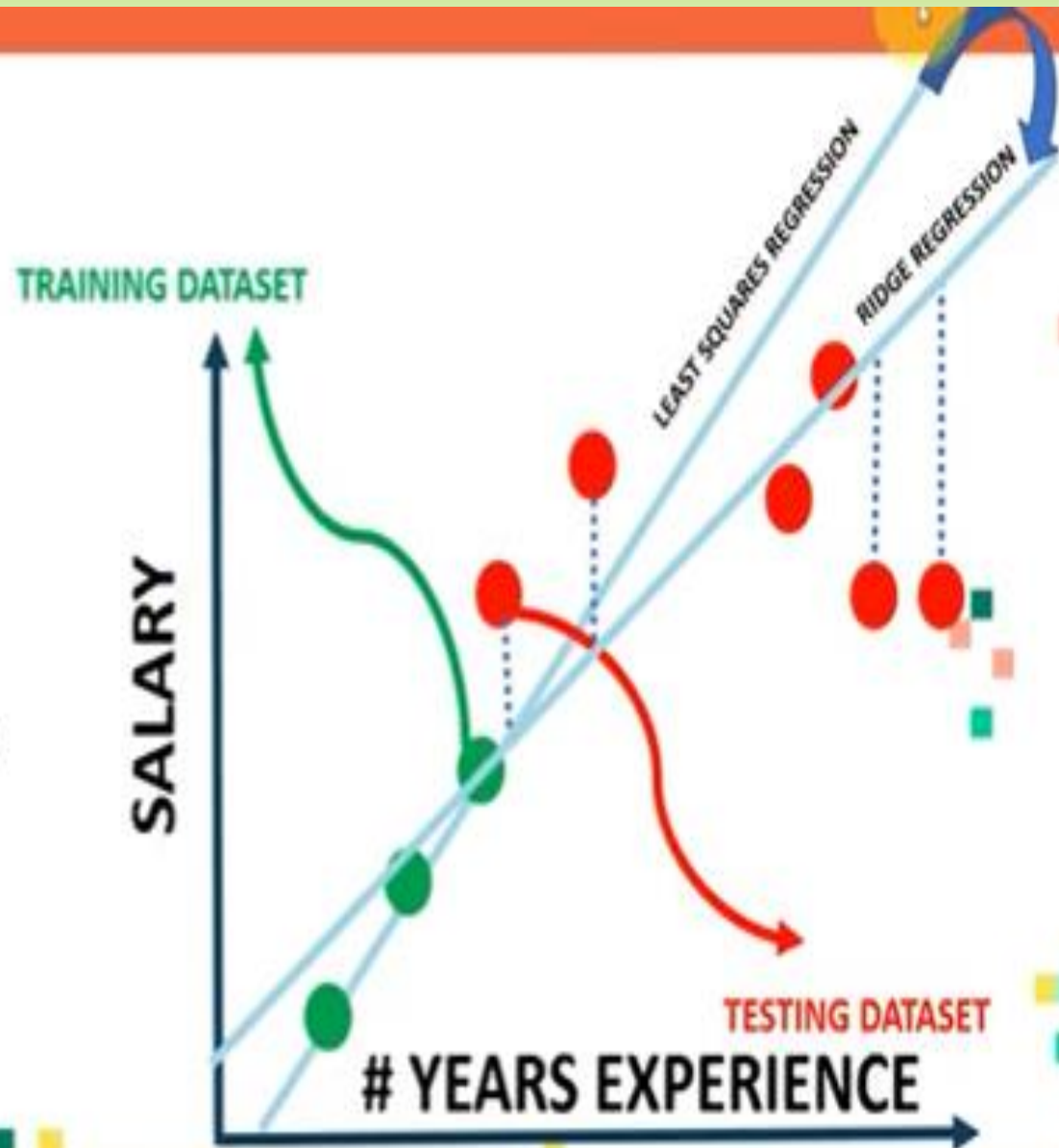
RIDGE REGRESSION (L2 REGRESSION): INTUITION

- Least sum of squares is applied to obtain the best fit line
- Since the line passes through the 3 training dataset points, the sum of squared residuals = 0
- However, for the testing dataset, the sum of residuals is large so the line has a high variance.
- Variance means that there is a difference in fit (or variability) between the training dataset and the testing dataset.
- This regression model is overfitting the training dataset



RIDGE REGRESSION (L2 REGRESSION): INTUITION

- Ridge regression works by attempting at increasing the bias to improve variance (generalization capability)
- This works by changing the slope of the line
- The model performance might be little poor on the training set but it will perform consistently well on both the training and testing datasets.



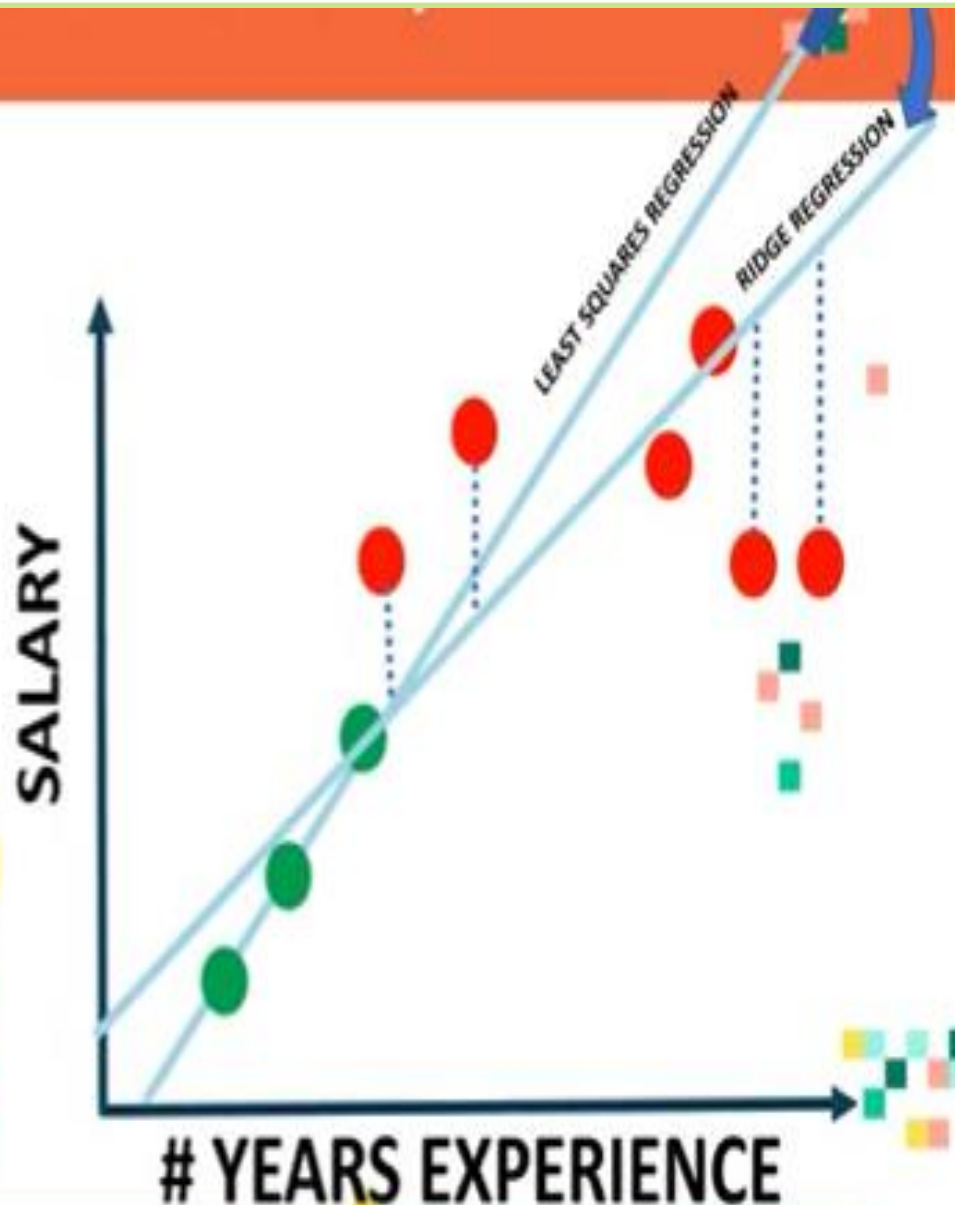
RIDGE REGRESSION (L2 REGRESSION): MATH

- Slope has been reduced with ridge regression penalty and therefore the model becomes less sensitive to changes in the independent variable (#Years of experience)

PENALTY TERM

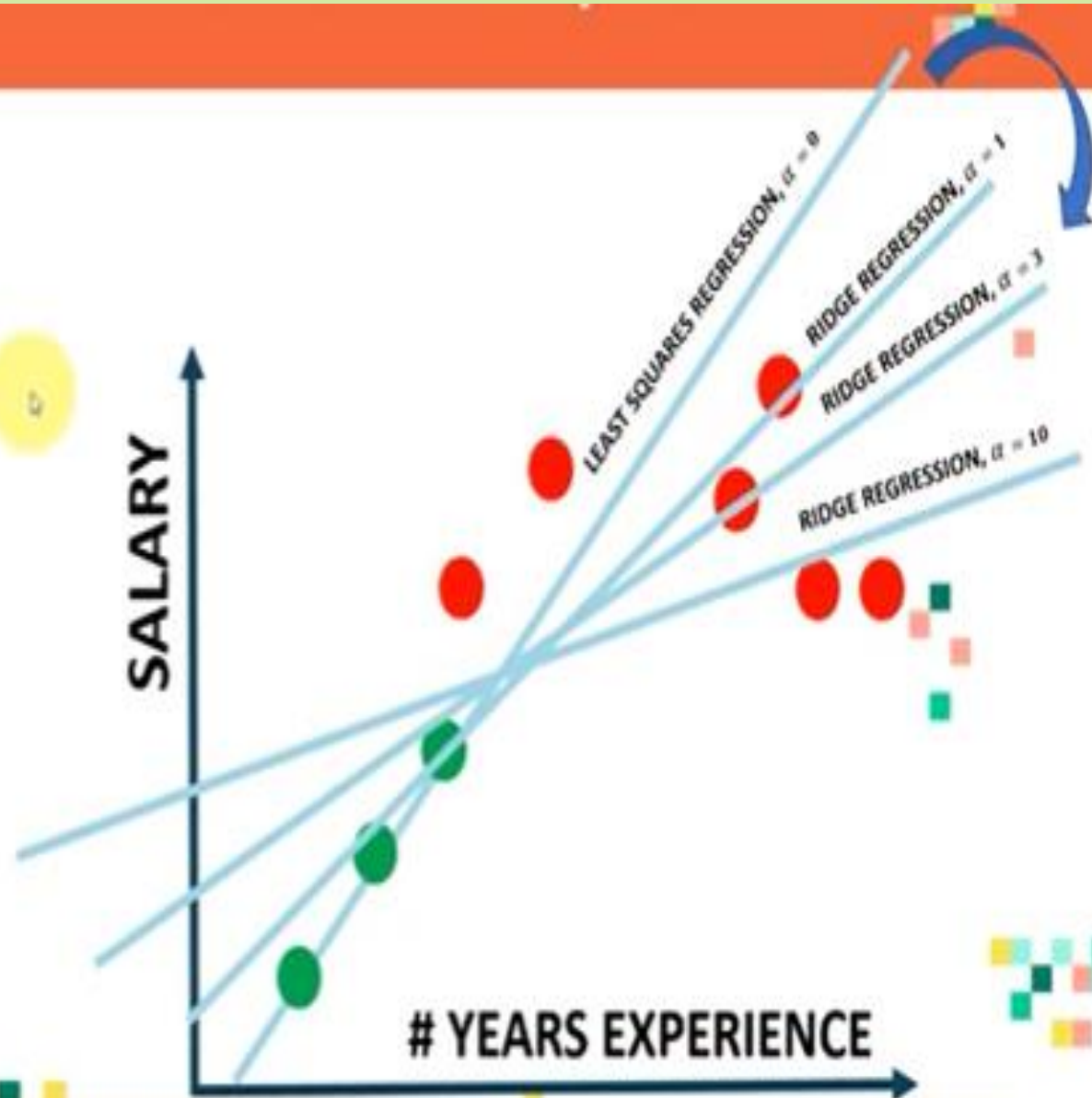
Least Squares Regression:
 $\text{Min}(\text{sum of the squared residuals})$

Ridge Regression:
 $\text{Min}(\text{sum of squared residuals} + \alpha * \text{slope}^2)$



RIDGE REGRESSION (L2 REGRESSION): ALPHA EFFECT

- As Alpha increases, the slope of the regression line is reduced and becomes more horizontal.
- As Alpha increases, the model becomes less sensitive to the variations of the independent variable (# Years of experience)



LASSO REGRESSION OR L1 REGRESSION OR L1-NORM

LASSO REGRESSION (L1 REGRESSION): MATH

- Lasso Regression is similar to Ridge regression
- It works by introducing a bias term but instead of squaring the slope, the absolute value of the slope is added as a penalty term

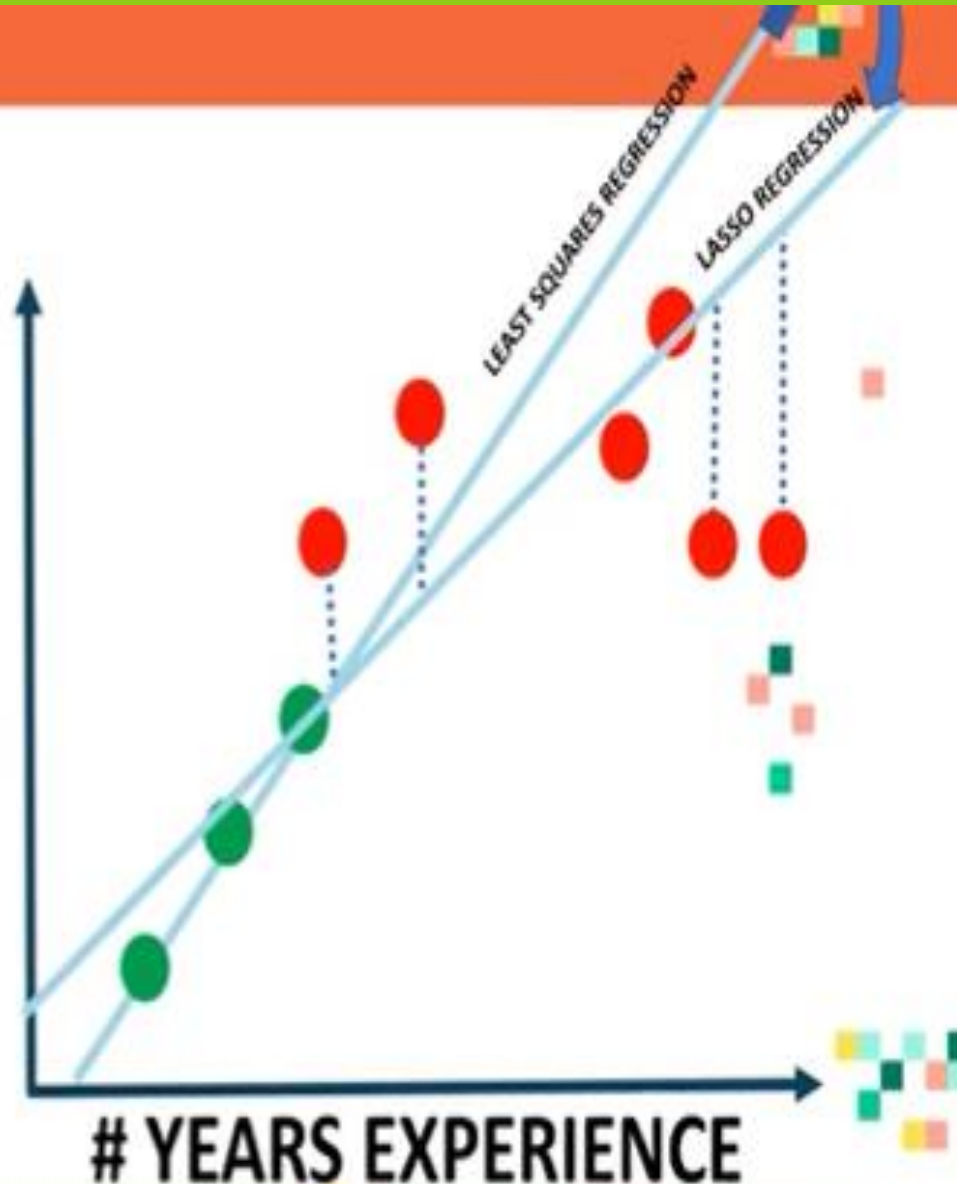
Least Squares Regression:
 $\text{Min}(\text{sum of the squared residuals})$

Lasso Regression:
 $\text{Min}(\text{sum of squared residuals} + \alpha * |\text{slope}|)$

PENALTY TERM

SALARY

YEARS EXPERIENCE



LASSO REGRESSION (L1 REGRESSION): EFFECT OF ALPHA

- The effect of Alpha on Lasso regression is similar to its effect on ridge regression
- As Alpha increases, the slope of the regression line is reduced and becomes more horizontal.
- As Alpha increases, the model becomes less sensitive to the variations of the independent variable (# Years of experience)



LASSO REGRESSION (L1 REGRESSION)

- Lasso regression helps reduce overfitting and it is particularly useful for feature selection
- Lasso regression can be useful if we have several independent variables that are useless
- Ridge regression can reduce the slope close to zero (but not exactly zero) but Lasso regression can reduce the slope to be exactly equal to zero.

Least Squares Regression:

Min(sum of the squared residuals)

Ridge Regression:

*Min(sum of squared residuals + $\alpha * \text{slope}^2$)*

Lasso Regression:

*Min(sum of squared residuals + $\alpha * |\text{slope}|$)*