Chapter-Two

Supervised Learning

Compiled By: Alemisa Endebu(MSc)

Outline

❑**Introduction:** What is Supervised Learning?

❑ **Linear model**

- **Regression**
- Understand the operation regression
  - *Linear regression*
  - *Polynomial regression*
  - *Regularization techniques*
  - *Understand the metrics used to evaluate regression*
  - *A case study in regression*

❑**Classification**

❑Understand the operation of classifiers

- Logistic regression
- KNN
- Naïve Bayes
- Decision trees
- Random forest
- Support vector machines

❑A case study in classification.

❑Understand the metrics used to evaluate classifiers.

What is Supervised Learning?

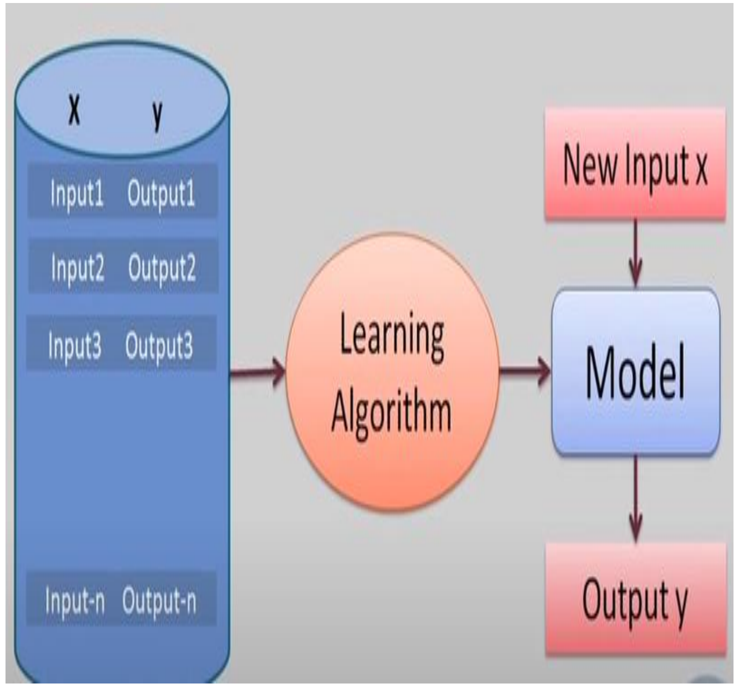
❖Supervised learning is a type of ML algorithm that learns from ***labeled data***.

❖**X, Y** (Pre-classified training examples)

❖Given A set of input features **X1,…, Xn. , What is the best label for Y?**

❖A set of training examples where the values for the ***input features*** and ***the target features*** are given for each example.

❖A new example is where only the values for the input features are given.



❖**Labeled data** consists of **input** and **output pairs**, where the input is the data that the algorithm is trying to learn from, and the ***output is the desired result.***

❖So, let us look at this picture which gives us a schematic diagram of ***Supervised learning***.

Ex: Supervised Learning Application

- ❖ Supervised learning algorithms are used in a wide variety of applications, including:
  - ❑ ***Spam filtering:*** To identify and classify spam emails.
  - ❑ ***Recommendation systems:*** To recommend products, movies, and other content to users based on their past behavior.
  - ❑ ***Fraud detection:*** to identify fraudulent transactions.
  - ❑ **Medical diagnosis:** to develop medical diagnostic tools that can help doctors diagnose diseases more accurately.
  - ❑ ***Image recognition:*** to develop image recognition systems that can identify objects in images and videos.
  - ❑ **Predict the price of a house**

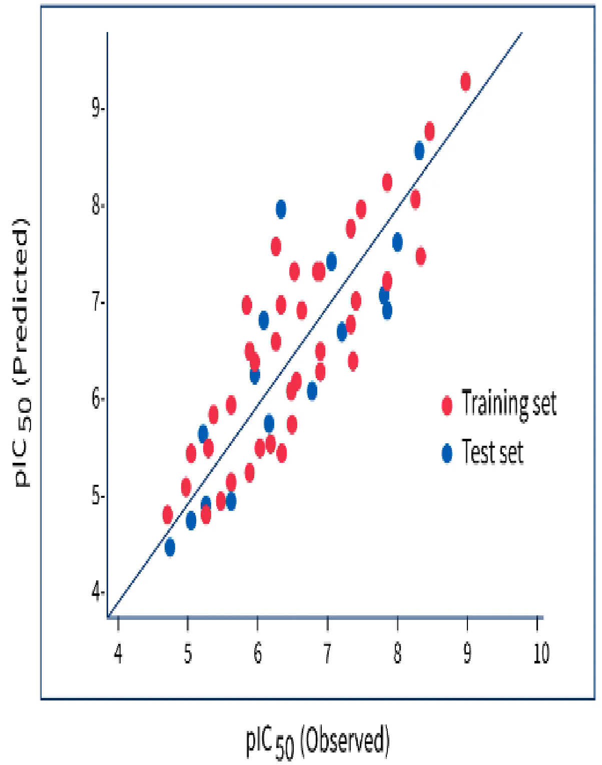
Linear model

- ❖The **linear model** is one of the most simple models in ML.
- ❖It assumes that the data is linearly separable and tries to learn the weight of each feature ***as shown in the figure***.
- ❖**Linear models** predict the target variable using a linear function of the input features.
- ❖ Mathematically, it can be written as

$$y = a_0 + a_1 \mathbf{x1} + a_2 \mathbf{x2} + a_3 \mathbf{x3} + \mathbf{\ldots} + \mathbf{an * xn}$$

where:

- ◦ **y** is the output variable
- ◦ $\mathbf{a_0}$ is the bias term
- ◦ **x1, x2, ..., xn** are the input variables
- ◦ **a1, a2, ..., an** are the weights
- ◦ The ***weights are learned*** during the training process, and they represent the importance of each input variable in ***predicting the output variable.***



❖**Linear models** can be used for both classification and regression tasks.

Regression

❖ What is Regression Machine Learning?

- ❖ ***Regression*** is a supervised learning technique that is used to predict ***continuous values.***
- ❖In Regression, we plot a graph between the variables that **best fit** the given data points.
- ❖The machine learning model can deliver ***predictions*** regarding the data.
- ❖ In naïve words, *"Regression shows a line or curve that passes through all the data points on a target-predictor graph in such a way that the vertical distance between the data points and the regression line is minimum."*
- ❖In a ***regression problem,***
  - ✓we are trying to predict results within a ***continuous output*****,** meaning that we are trying to map input variables to some ***continuous function.***
- ❖Let **X, Y** (Pre-classified training examples)
  - ✓Regression when Y is ***continuous.***

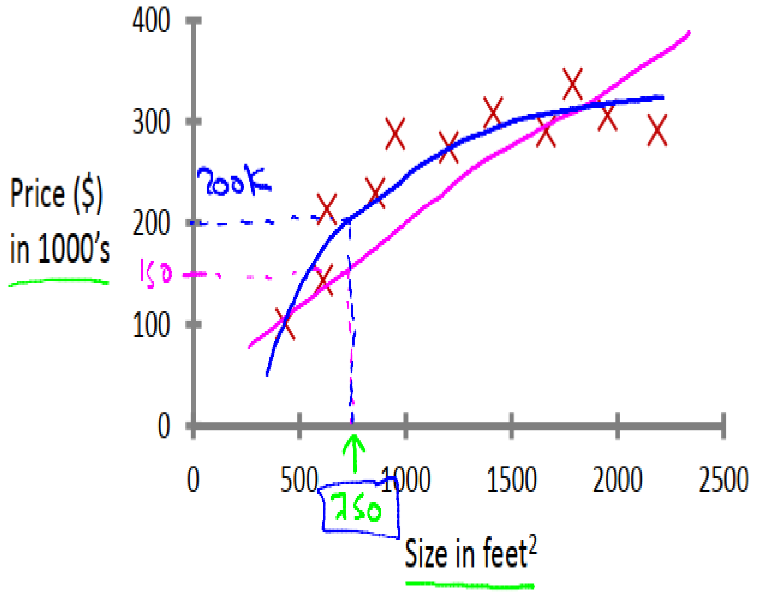
Regression: Ex#1- Predict the price of a house

- ❖ **Regression algorithms** are used to predict the numerical value of a new data point.
- ❖ **For example, used to predict the price of a house.**

❖*Let's say you want to predict housing prices.*

❑ *A while back a student collect data sets from the city of Portland Oregon and let's say plotted the data set and it looks like this.*

- *As shown in the figure.*



✓ **Here, the horizontal axis represents, the size of different houses in square feet.**

✓ **The vertical axis represents the price of different houses in thousands of dollars.**

Regression: Ex#1- Predict the price of a house…

❖So given this data, let's say you have a friend who owns a house that is say ***750 square feet***, and they are hoping to sell the house and they want to know how much they can get the house.

❖**So, How can learning algorithms help you?**

- ❑One thing a learning algorithm might want to do is ***put a straight line through the data,*** and also ***fit a straight line to the data***.
- ❑Based on that we predict the house price. But maybe this isn't the only learning algorithm you can use and there might be a better one.
- ❑For example, instead of ***fitting a straight line*** to the data we might decide that it's better to ***fit quadratic functions*** or a ***second-order polynomial*** to this data.

Understand the operation regression

❖ There are many different regression algorithms available, but some of the most common include:

- **Linear regression:**
  - ✓ is the simplest type of regression algorithm.
  - ✓ It assumes that the relationship between the ***input*** and ***output*** variables is linear.
- **Polynomial regression:**
  - ✓ is a generalization of linear regression that can be used to model non-linear relationships between the **input** and **output** variables.
- **Regularization techniques:**
  - ✓ is a technique used to reduce errors by fitting the function appropriately on the given training set and avoiding overfitting.

i. Linear regression

❖Our first learning algorithm will be linear regression.

❖**Linear regression** is a type of ML algorithm more specifically a supervised ***ML*** algorithm.

❖**Linear regression** is a quiet and simple statistical regression method used for predictive analysis and shows the relationship between the ***continuous variables***.

❖It learns from the ***labeled datasets*** and maps the data points to the most optimized linear functions which can be used for *prediction on new datasets*.

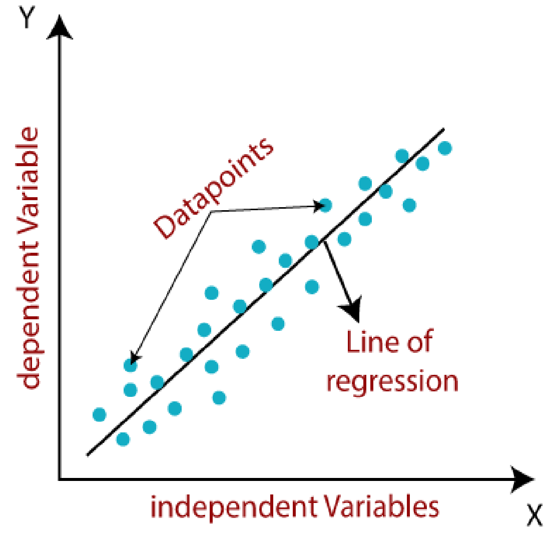
❖**Linear regression** graph shows the linear relationship between the ***independent variable (X-axis)*** and the ***dependent variable (Y-axis).***

❖ The **linear regression model** gives a sloped straight line describing the relationship within the variables.

❖ When the value of x ***(independent variable)*** increases, the value of y ***(dependent variable)*** is likewise increasing.

❖ The **blue line** is referred to as the *best-fit straight line.*

❖ Based on the given data points, we try to plot a line that models the points the best.



Linear regression...

❖When the number of the ***independent feature***, is ***1*** then it is known as ***Univariate Linear regression***, and

❖When the number of ***independent features is*** more than 1, it is known as ***multivariate linear regression.***

❖Mathematically, we can represent a ***Univariate Linear regression*** as:

$$y= a_0+a_1x$$

Here,

- ❑ **Y**= Dependent Variable (Target Variable)
- ❑ **X**= Independent Variable (predictor Variable)
- ❑ **a0**= intercept of the line (Gives an additional degree of freedom)
- ❑ **a1** = Linear regression coefficient (scale factor to each input value).

:

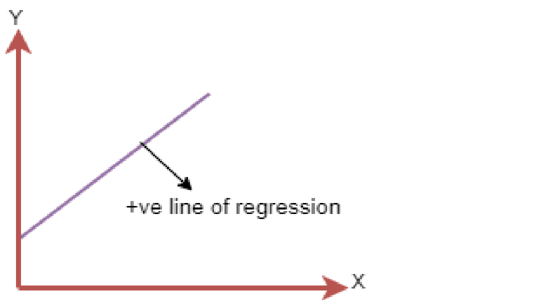
❖ The values for the x and y variables are training datasets for Linear Regression model representation.

Need of a linear regression

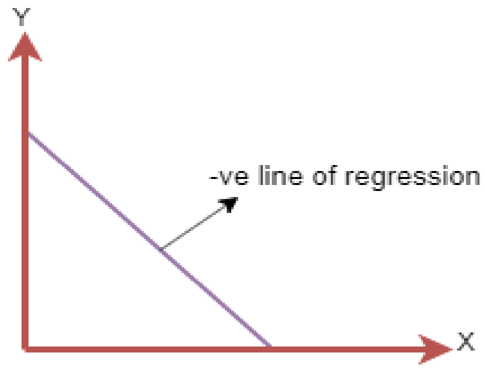
- ❖As mentioned in the previous slide, Linear regression estimates the relationship between a ***dependent variable*** and an ***independent variable***.
- ❖***Let's understand this with a scenario:***
    - ❑*Let's say we want to estimate the salary of an employee based on years of experience. You have the recent company data, which indicates the relationship between experience and salary. Here year of experience is an independent variable, and the salary of an employee is a dependent variable, as the salary of an employee is dependent on the experience of an employee. Using this insight, we can predict the future salary of the employee based on current & and past information.*

Linear Regression Line

- ❖A **linear line** showing the relationship between the ***dependent*** and ***independent variables*** is called a **regression line**.
- ❖A **regression line** can show two types of relationship:
  - ❑**Positive Linear Relationship:**
    - ✓If the dependent variable increases on the Y-axis and independent variable increases on X-axis .
  - ❑**Negative Linear Relationship:**
    - ✓If the dependent variable decreases on the Y-axis and independent variable increases on the X-axis.



The line equation will be: **$Y= a_0+a_1x$**



The line of equation will be: **$Y= -a_0+a_1x$**

- ❖ *The goal of the* ***linear regression algorithm*** *is to get the best values for* ***a0*** *and* ***a1*** *to find the best-fit line.*
- ❖ *The best-fit line should have the* ***least error*** *means the error between predicted values and actual values should be minimized.*

Types of Linear Regression

❖Linear regression can be further divided into two types of algorithms.

i. Simple Linear Regression:

- ❑i.e. If a single independent variable is used to predict the value of a numerical ***dependent variable***
- ❑ it is also called ***Linear regression With One variable***

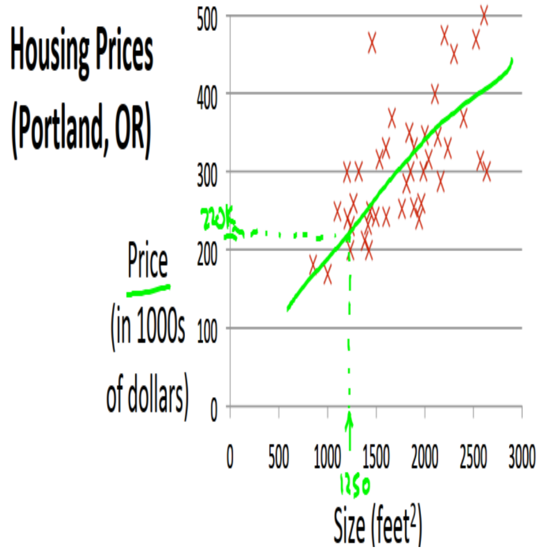
ii. Multiple Linear regression:

- ❑i.e. If more than one independent variable is used to predict the value of a numerical ***dependent variable***.
- ❑ it is also called ***Linear regression With Multiple-variable***

i. Linear regression With One variable

- Linear Regression with one variable is also called "**univariate linear regression**.
- It is used when you want to predict a **single output value** from ***a single input value***.
- That means you only have one **x as input(attribute)** and **one y as output**.

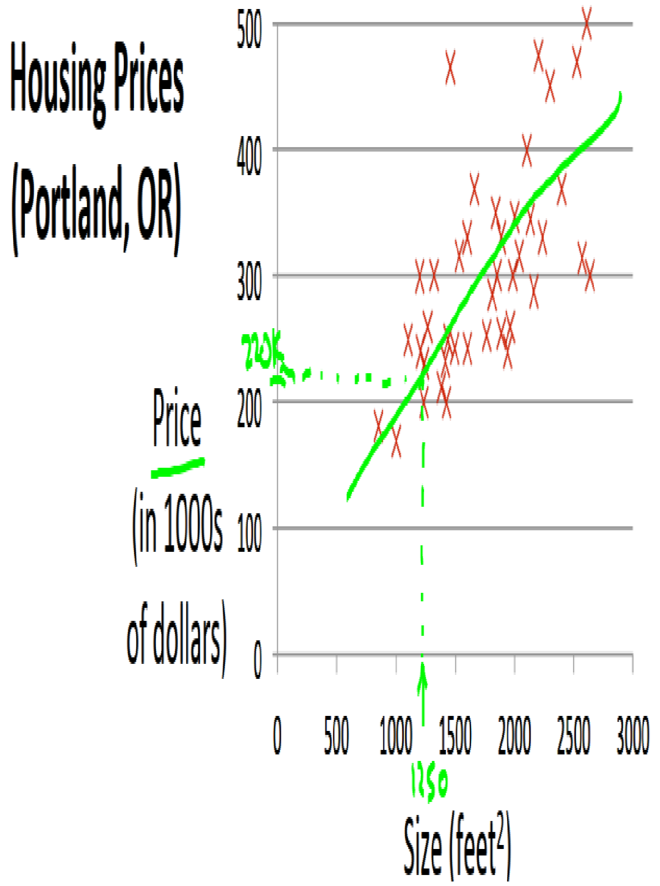
- So, in this section, let we will see:
  - **What does the model look like?**
  - **What the overall process of supervised learning looks like.**
- *Let's use some motivating examples for predicting housing prices. We're going to use a dataset of housing prices from the city of Portland, Oregon. And here we gonna plot our data set of a number of houses that were different sizes that were sold for a range of different prices.* ***As shown fig:***



- ❑ Let's say that given this data set, you have a friend who's trying to sell a house, let's see if a friend's house is a size of **1,250 square feet** and you want to tell them how much they might be able to sell the house for?

Linear regression With One variable....

- ❖ Well, one thing you could do is ***fit a model.***
- ❖ Maybe fit a straight line to the data. As shown fig
- ❖ Looks something like that and based on that, maybe you could tell your friend that let's say maybe he can sell the house for around **$220,000.**
- ❖ So this is an example of a supervised learning algorithm because we're given the "***right answer***" for each of our examples. **Namely,** we're told
  - ❑ *What is the **actual house?***
  - ❑ *What was the actual price of each of the houses in our data set?*
- ❖ Moreover, this is an example of a ***regression problem***
- ❖ where the term regression refers to the fact that we are predicting a ***real-valued output*** namely the price.



Linear regression With One variable

Ex: *Predicting housing prices.*

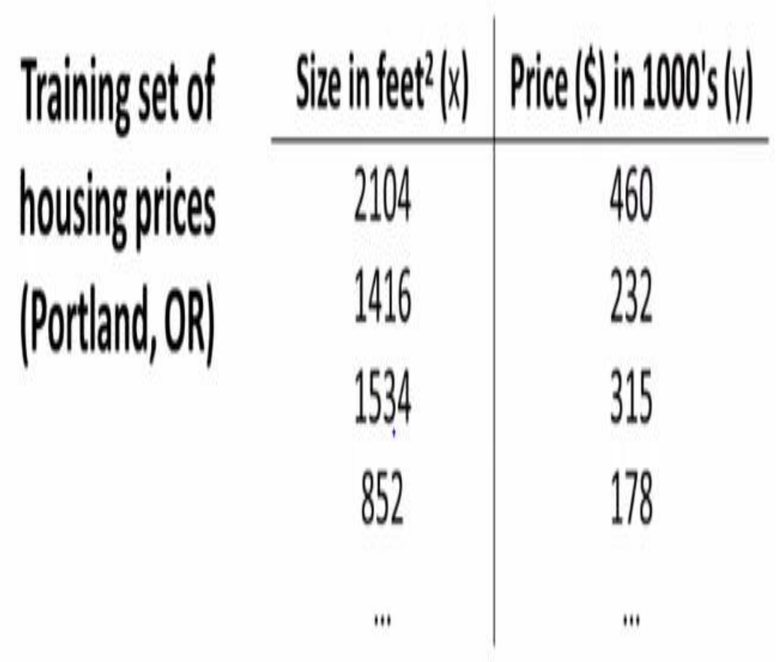
❖So for the housing prices example, we have a data set and this data set is called a ***training set.***

❖Our job is to learn from this data how to predict the prices of the houses.

❖Let's define some notation:

❖**Notation:**

- ❑ **n**=Number of training examples
- ❑ **x's**="input " variable/ features
- ❑ **y's**="output" variable/ "target" variable.



Training set of housing prices (Portland, OR)

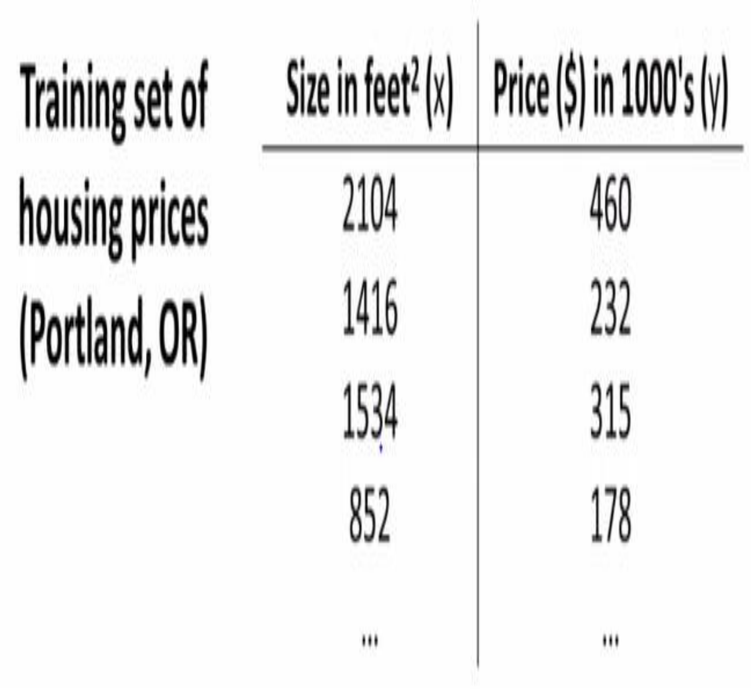
Size in feet^2 (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
...	...

- So in this data set, let's say ***47 rows*** in this table.
- **n** equals **47.**
- **x** it would be the input features.
- **y** to denote output variables or the target variable that is going to predict and so that's the second column.

Linear regression With One variable

Ex: *Predicting housing prices…*

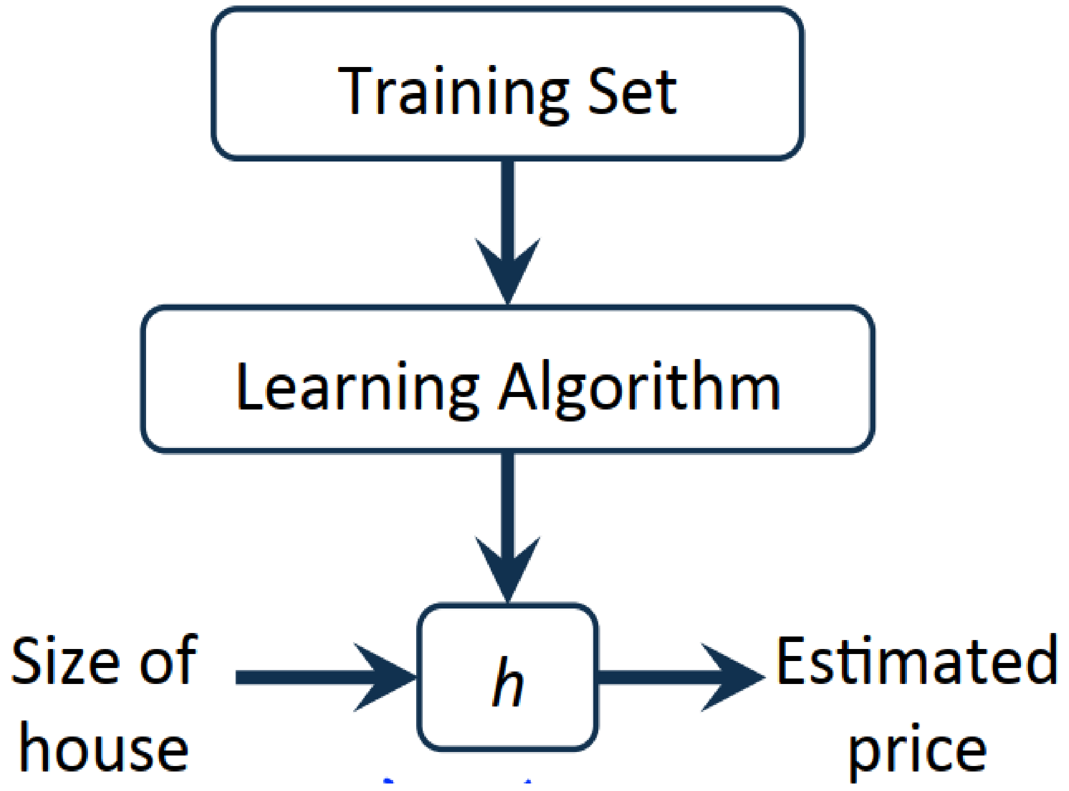
- ❖In the given table **(x, y)** denote a ***single training example.***
- ❖ So, a single row in this table corresponds to a single training example and
- ❖refers to a specific training example or ***one training example.***
- ❖ Another **notation** is **$(x^{(i)}, y^{(i)})$** - this to refer to the ith training example. So this superscript **i** not exponentiation.
- ❖It refers to the ***ith row*** in this table.
- ❖For example, **x(1)** refers to the input value for the first training example so that's ***2104***.
- ❖**x(2)** will be equal to **1416.**
- ❖That's the second **x** and **y(1)** will be equal to **460.**



Training set of housing prices (Portland, OR)

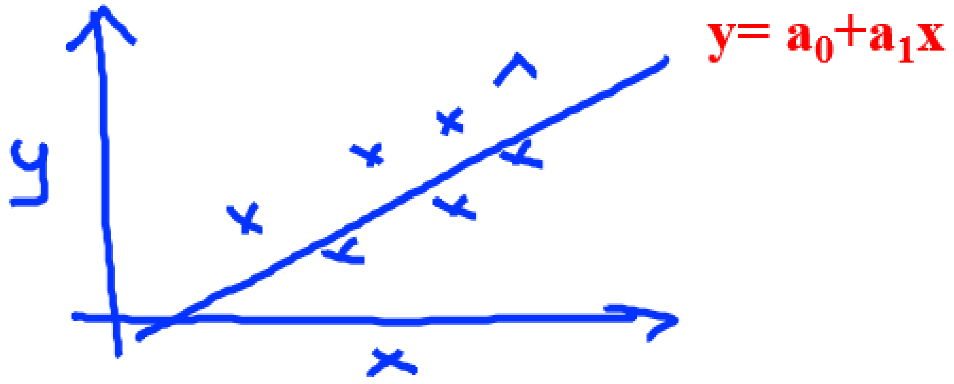
Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
...	...

How this supervised learning algorithm works?



- So, How do we represent *h*?
  - *instead of h, let us use y.*

$$y = a_0 + a_1 x$$



- The job of the hypothesis is a function that takes the size new house x to the estimated price y.
- That means: h maps from x's to y's

❖ **Linear regression with one variable. Or Univariate linear regression.**

Finding the Best fit line

❖**When working with linear regression,**

- ✓ Our main goal is to find the ***best-fit line*** which means the error between ***predicted values*** and ***actual values*** should be minimized.

❖The ***best-fit line*** will have the ***least error.***

- ✓The different values for **weights** or the ***coefficient of lines*** **(a_0, a_1)** gives a different **line of regression,**
- ✓so we need to calculate the best values for **a_0** and **a_1** to find the ***best-fit line***, so to calculate this we use ***cost function*****.**

Cost function

- ❑*Need for Cost function*
- ❑*What is cost Function?*
- ❑*Cost Function for linear Regression*
- ❑*Gradient Descent Algorithm*
- ❑*Gradient Descent for Linear Regression*

Need for Cost function

Cost Function?

❖This section defines the *cost function,* which will let us figure out how to fit the *best possible straight line to our data.* *And Needs of cost function.*

❑A ML model should have a very high level of accuracy in order to perform well with ***real-world applications.***

❑But how to calculate the accuracy of the **model?**

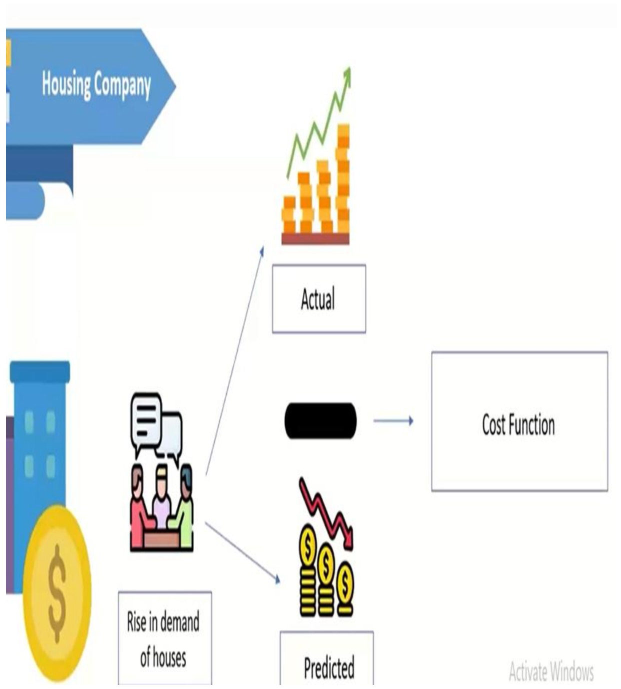
❑ i.e., ***how good or poor*** our model will perform in the real world?

❑ In such a case, the **Cost function** comes into existence.

❑***What is cost Function?***

❑*A cost function is an important parameter that determines how well a ML model performs for a given* ***dataset****. or*

❑**Cost function measures how badly a model is performing.**



❖*Expressed as the difference between actual and predicted values and present that error in the form of a single real number. As shown in the figure above.*

What is cost Function? …

❖Depending on the problem, ***Cost function*** can be formed in many different ways. The purpose of cost function is to be either ***minimized*** or ***maximized***.

- ❑ **Minimized:** The returned value is usually called ***cost, loss or error.***
  - *The goal is to find the values of model parameters for which cost function return as small a number as possible.*
- ❑ **Maximized:** In this case, the value it yields is named a reward.
  - *The goal is to find values of model parameters for which the returned number is as large as possible.*
  - *For algorithms relying on gradient descent to optimize model parameters, every function has to be differentiable.*

What is cost Function? …

Consider the Scenario

- *Consider a robot trained to stack boxes in a factory. The robot might have to consider certain changeable parameters, called **Variables**, which influence how it performs.*
- *Let's say the robot comes across an obstacle, like a **rock.** The robot might bump into the rock and realize that it is not the correct action. It will learn from this, and next time it will learn to avoid rocks. Hence, your machine uses variables to better fit the data. The outcome of all these obstacles will further optimize the robot and help it perform better. It will generalize and learn to avoid obstacles in general, say like a fire that might have broken out. The outcome acts as a **cost function**, which helps you optimize the variable, to get the best variables and fit for the model.*

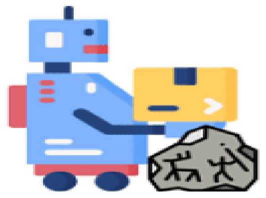
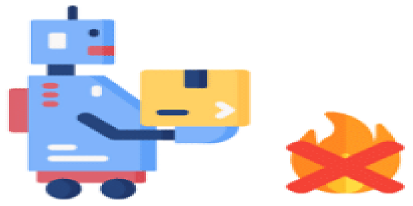



Fig1: Robot learning to avoid obstacles

How to Tailor a Cost Function

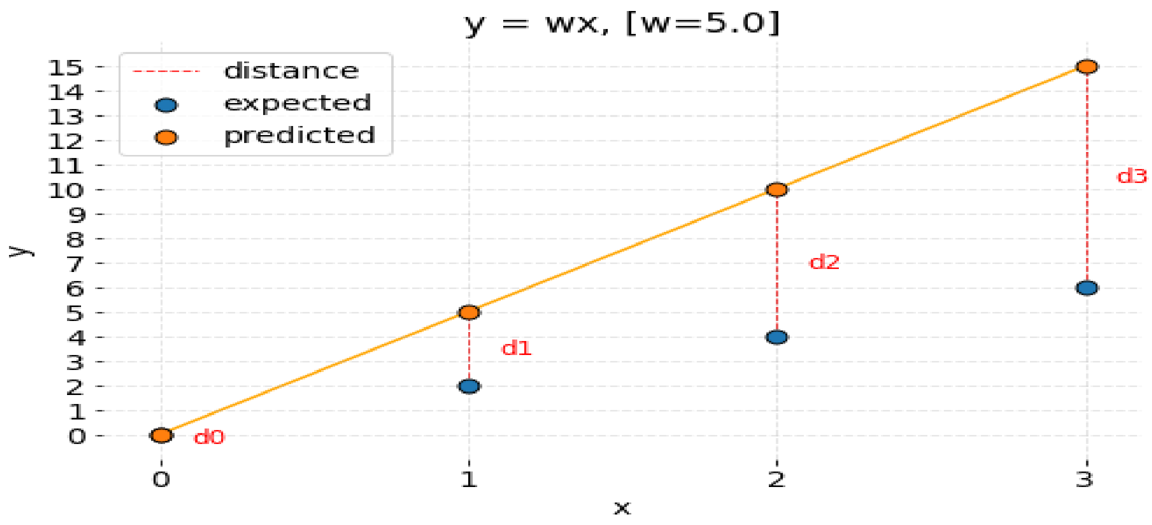
❖Let's start with a model using the following : $\hat{y} = wx$ Formula

- ŷ = predicted value,
- x = vector of data used for prediction or training
- w = weight.

- **Notice** that we've omitted the ***bias on purpose***. $x_0 = [0], x_1 = [1], x_2 = [2], x_3 = [3]$
- Let's try to find the value of **weight parameter**, so for the following data samples:
- The outputs of the model are as close as possible to: $y_0 = 0, y_1 = 2, y_2 = 4, y_3 = 6$
- Now it's time to assign a random value to the ***weight parameter*** and visualize the model's results.

Cont'd

- Let's pick w = 5.0 for now.



- We can observe that the **model predictions** are different than **expected values** but ***how can we express that mathematically?***
- The most ***straightforward idea*** is to subtract both values from each other and see if the result of that operation equals zero. Any other result means that the values differ.
- The size of the received number provides information about how significant the error is.
- From the geometrical perspective, it's possible to state that error is the distance between two points in the coordinate system.

Cont'd

- **Let's define the distance as:** $distance = \hat{y} - y$

$$d_0 = 0 - 0 = 0$$
$$d_1 = 5 - 2 = 3$$
$$d_2 = 10 - 4 = 6$$
$$d_3 = 15 - 6 = 9$$

❖ According to the formula, calculate the errors between the ***predictions*** and ***expected*** values:

❖ As stated before, cost function is a **single number** describing model performance.

❖ Therefore let's sum up the errors.

$$cost(w) = d_0 + d_1 + d_2 + d_3$$

$$cost(5) = 0 + 3 + 6 + 9 = 18$$

❖ However, now imagine there are a million points instead of four.

❖ The accumulated errors will become a bigger number for a model making a prediction on a larger data set than on a smaller data set.

❖ Consequently, we can't compare those models.

❖ That's why we have to scale in some way.

❖ The right idea is to divide the accumulated errors by the number of points.

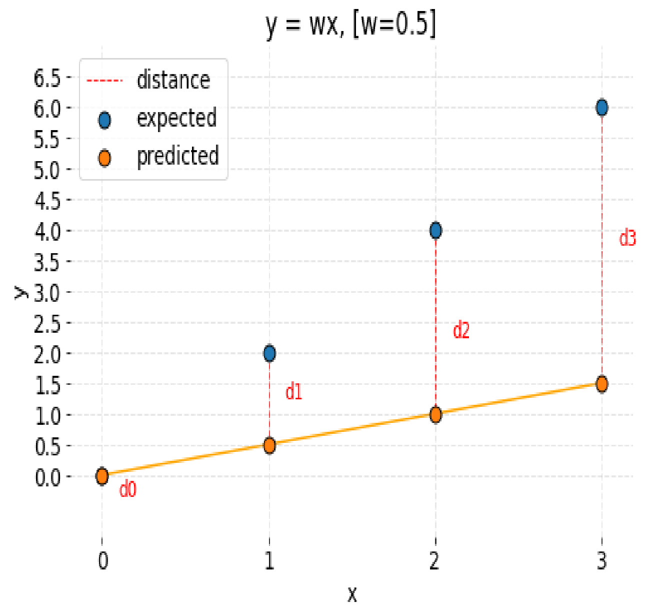
❖ Cost stated like that is mean of errors the model made for the given data set.

$$cost(w) = \frac{1}{4}(d_0 + d_1 + d_2 + d_3)$$

$$cost(5) = \frac{1}{4}(0 + 3 + 6 + 9) = \frac{18}{4} = 4.5$$

Cont'd

- ❖ Unfortunately, the formula isn't complete.
- ❖ We still have to consider all cases so let's try picking
- ❖ Smaller weights and see if the created cost function works.
- ❖ We'll set weight to w = 0.5.



- ❖ The predictions are off again. However, in comparison to the previous case, that predicted points are below expected points.
- ❖ Numerically, predictions are smaller.
- ❖ The cost formula is going to malfunction because calculated distances have negative values.

$$d_0 = 0 - 0 = 0$$

$$d_1 = 0.5 - 2 = -1.5$$

$$d_2 = 1 - 4 = -3$$

$$d_3 = 1.5 - 6 = -4.5$$

Cont’d

❖ The *cost value* is also negative: $cost(0.5) = \frac{1}{4}[0 + (-1.5) + (-3) + (-4.5)] = \frac{-9}{4} = -2.25$

❖ Since distance can’t have a negative value, we can attach a more substantial penalty to the predictions located above or below the expected results (some cost functions do so, e.g. RMSE), but the value shouldn’t be negative because it will cancel out positive errors.

❖ It will then become impossible to properly minimize or maximize the cost function.

❖ So how about fixing the problem by using the absolute value of the distance?

$$distance = |\hat{y} - y|$$

❖ After stating the distance as:

❖ The costs for each value of weights are:

$$cost(5) = \frac{1}{4}(|0| + |3| + |6| + |9|) = \frac{18}{4} = 4.5$$

$$cost(0.5) = \frac{1}{4}(|0| + |-1.5| + |-3| + |-4.5|) = \frac{9}{4} = 2.25$$

❖ Now we’ve correctly calculated the costs for both weights **w = 5.0** and **w = 0.5.**

❖ It is possible to compare the parameters.

❖ The model achieves better results for **w = 0.5** as the cost value is ***smaller***.

❖ The function we created is mean absolute error.

Cost Function for linear Regression

❖We can use the **cost function** to find the accuracy of the **mapping function**, which maps the ***input variable*** to the ***output variable***.

❖This mapping function is also known as ***Hypothesis function***.

❖For **Linear Regression,** we use the **Mean Squared Error (MSE)** cost function, which is the average squared error that occurred between the ***predicted values*** and ***actual values***.

❖ It can be written as:

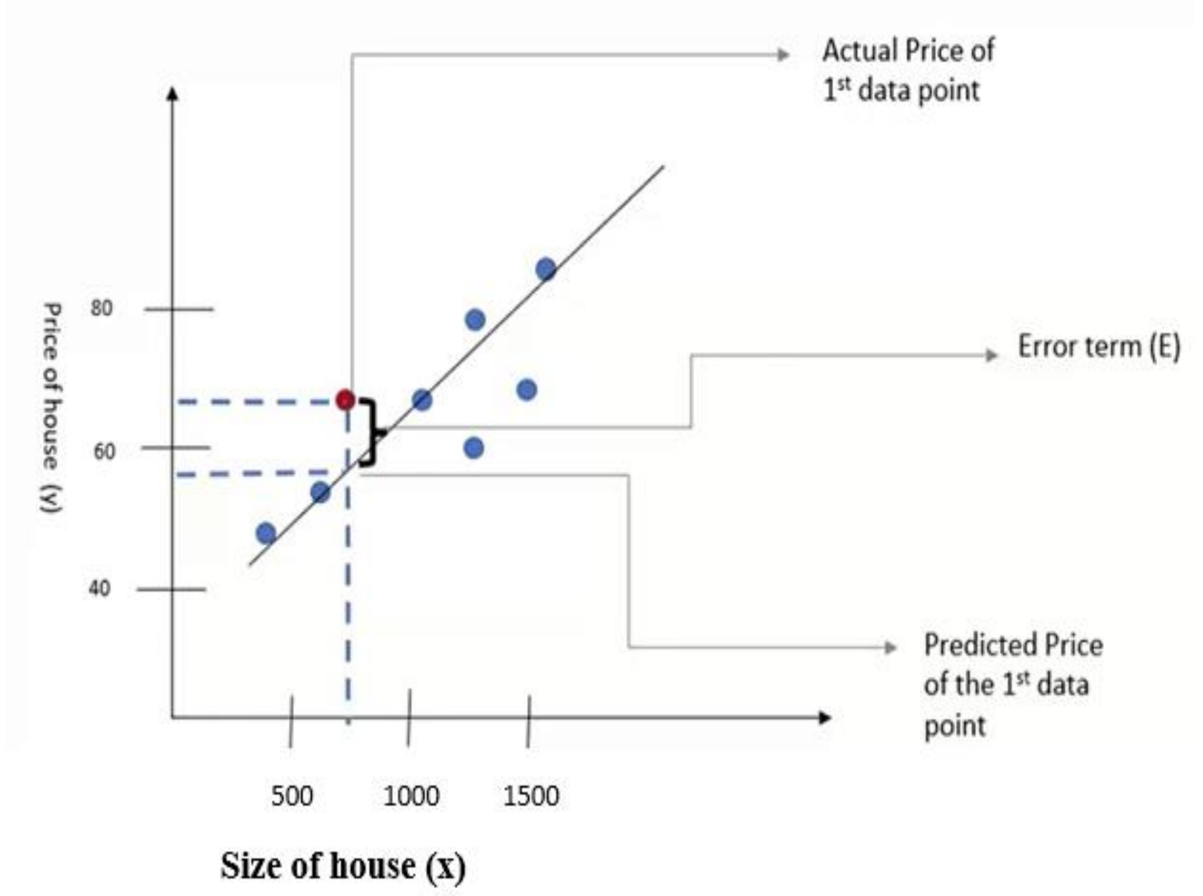
$$\boxed{MSE = 1\frac{1}{N}\sum_{i=1}^{n}(y_i - (a_1 x_i + a_0))^2}$$

❖For the above **linear equation, MSE** can be calculated as:

❖**Note**: *Depending on the problem, cost function can be formed in many different ways.*

Cost Function for Linear Regression…

- How to calculate the error term?



Error Term (E) = ($y_{\text{actual}} - y_{\text{predicted}}$)

$E_1 = (y_{1\ \text{actual}} - y_{1\ \text{predicted}})$

$E_1 = (y_{2\ \text{actual}} - y_{2\ \text{predicted}})$

.

.

.

$E_n = (y_{n\ \text{actual}} - y_{n\ \text{predicted}})$

Square the error terms

$E_n = (y_{n\ \text{actual}} - y_{n\ \text{predicted}})^2$

Cost Function for Linear Regression...

- Now, Let us find the cost function.

Step 1: write the sum of all error terms.

$$\text{Cost function (J)} = \frac{1}{2n}\left(E_1^2 + E_2^2 + E_3^2 + \ldots\ldots\ldots + E_n^2\right)$$

Step 2: Re-write equation #1.

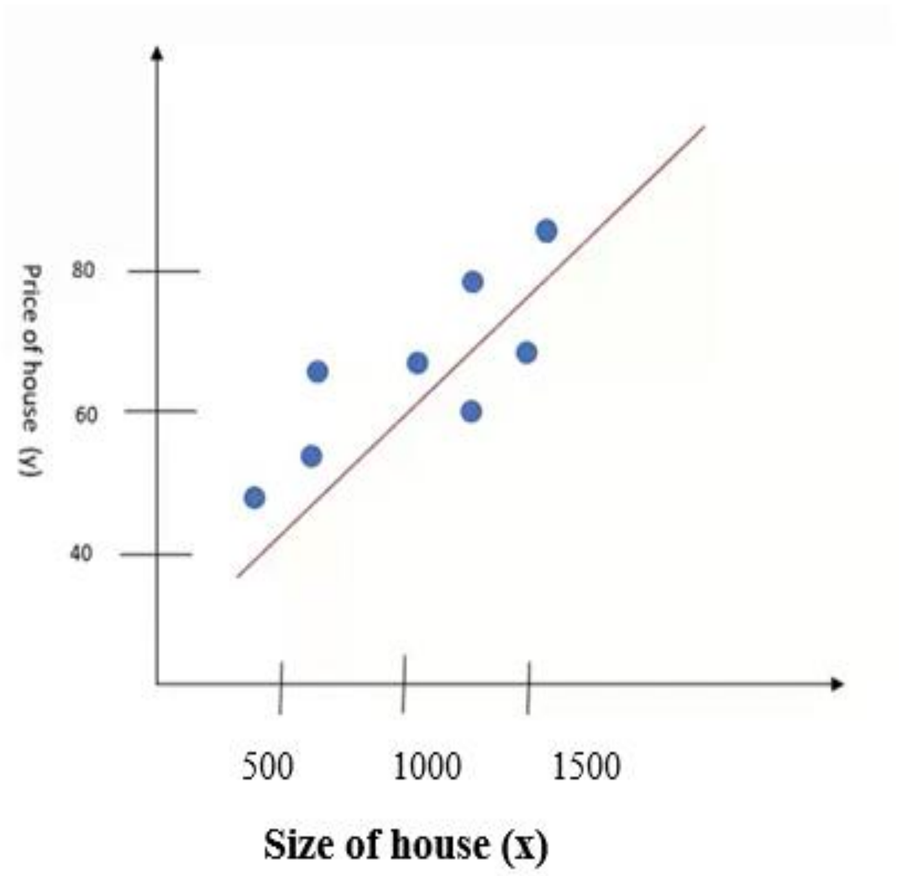
$$J(a) = \frac{1}{2n}\sum_{i=1}^{n}\left(y_{i\,(\text{actual}} - y_{i\,(\text{predicted})}\right)^2$$

Step 3: Re-write equation #2 and expand the predicted term..

$$J(a) = \frac{1}{2n}\sum_{i=1}^{n}\left(y_{i\,(\text{actual})} - \left(a_0 + a_1 x_1^{(i)}\right)\right)^2$$

Cost function(F) is a function of a_0 and a_1

Cost Function for Linear Regression...

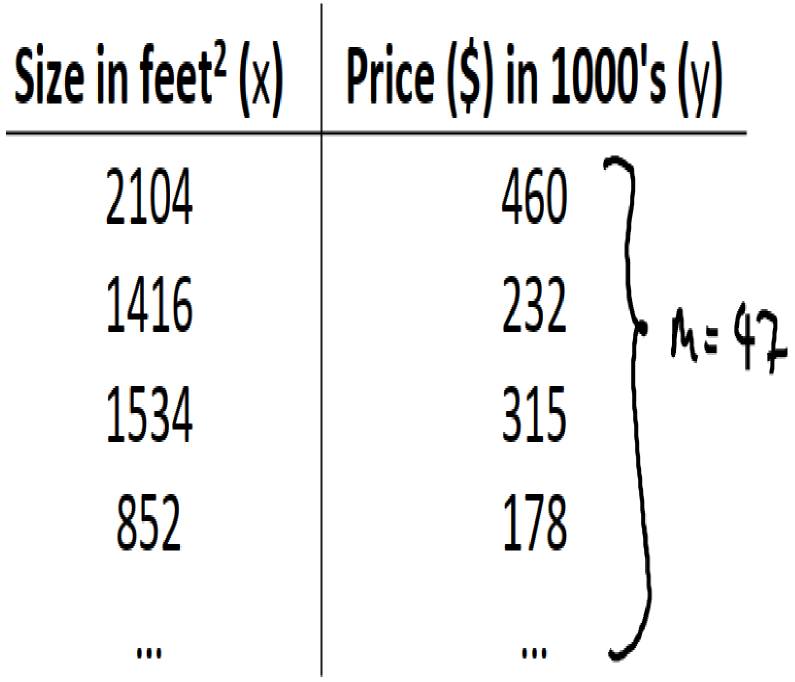


Best Model is the one which reduces the error which is the Cost function.

This Straight line is called line of best fit line for this model.

Cost Function for linear Regression

- In linear regression, we have a training set as shown in the table.
- Notation M is the number of training examples, **n= 47.**
- And the form of our hypothesis, which we use to make predictions is this ***linear function.***
- What we're going to do is how to go about choosing these two **parameter values**, a_0 and $\mathbf{a_1}$.



Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
...	...

$$\left.\begin{matrix} 460 \\ 232 \\ 315 \\ 178 \\ \ldots \end{matrix}\right\} M = 47$$

- With different choices of the parameters $\mathbf{a_0}$ and $\mathbf{a_1}$, we get different hypotheses and different hypothesis functions.

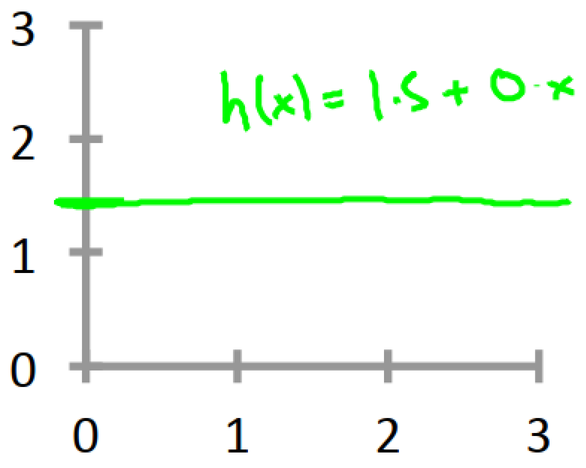
Hypothesis: $\mathbf{y = a_0 + a_1 x}$

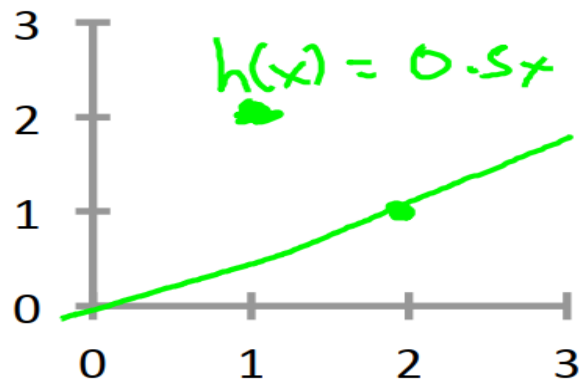
$\mathbf{a_{i\text{'}s}}$ **:** **Parameters**

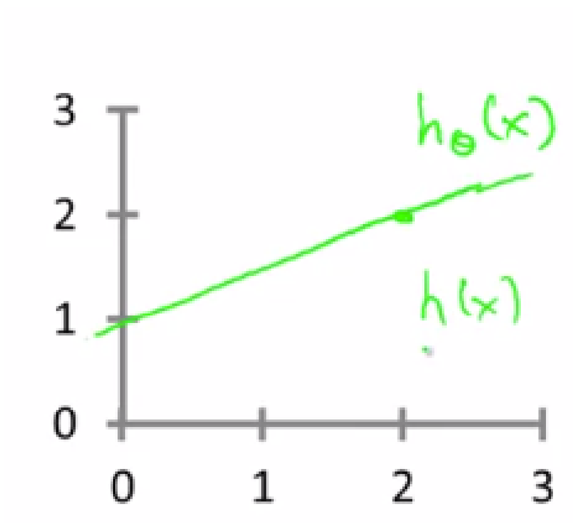
How to choose: $\mathbf{a_{i\text{'}s}}$ **?**

Cost Function for linear Regression

$y = a_0 + a_1 x$







- *If $a_0 = 1.5$ and $a_1 = 0$*

- hypothesis function will be **h(x)= 1.5+0x** which is this constant value function which is phat at **1.5.**

If $a_0 = 0$ and $a_1 = 0.5$

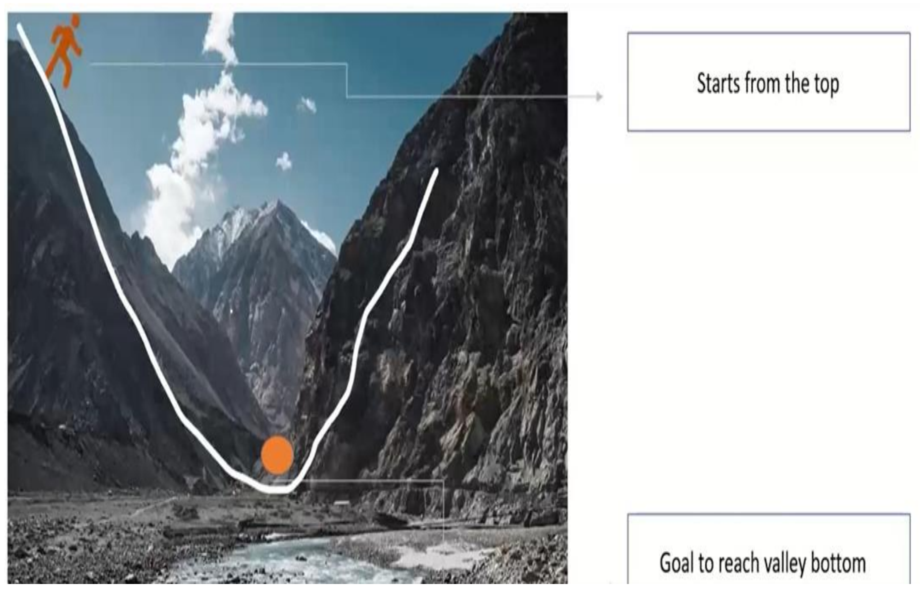
h(x) will be equal to **0.5** times x

- *If $a_0 = 1$ and $a_1 = 0.5$*

- it should pass through the two-two point.
- this is a new vector of x, i.e. h(x)

Gradient Descent Algorithm

- **Gradient Descent** is an algorithm that is used to **optimize** the ***cost function*** or the ***error*** of the model.
- It is used to ***minimize the cost function*** in linear regression.
- **GD-**can be thought of as the direction you have to take to reach the ***least possible error.***
- The error in your model can be *different at different points*, and you have to find the quickest way to **minimize** it, to prevent resource wastage.
- **Gradient Descent** can be visualized as a **ball rolling down a hill.** (***look at the picture***)
- Here, the ball will roll to the lowest point on the hill.
- It can take this point as the point where the error is least as for any model, the error will be minimum at one point and will increase again after that.



Gradient Descent Algorithm

- The goal of the gradient descent algorithm is
    - to ***optimize the cost function*** and ***minimize the error.***
- In ***gradient descent,*** you find the error in your model for different values of input variables.
- This is repeated, and soon you see that the error values keep getting smaller and smaller.
- Soon you’ll arrive at the values for variables when the error is the least, and the ***cost function*** is optimized.

Gradient Descent Algorithm: look at Scenario

- Imagine a pit in the shape of U ***(Fig a).*** You are standing at the topmost point in the pit, and your objective is to ***reach the bottom of the pit***. There is a treasure, and you can only take a discrete number of steps to reach the bottom. If you choose to take longer steps each time, you may get to sooner but, there is a chance that you could overshoot the bottom of the pit and not near the bottom ***(Fig b).*** If you decide to take one footstep at a time, you would eventually get to the bottom of the pit but, this would take a longer time *(fig c).* **So, what will do?**

Starts from the top
Goal to reach valley bottom

Figure-a

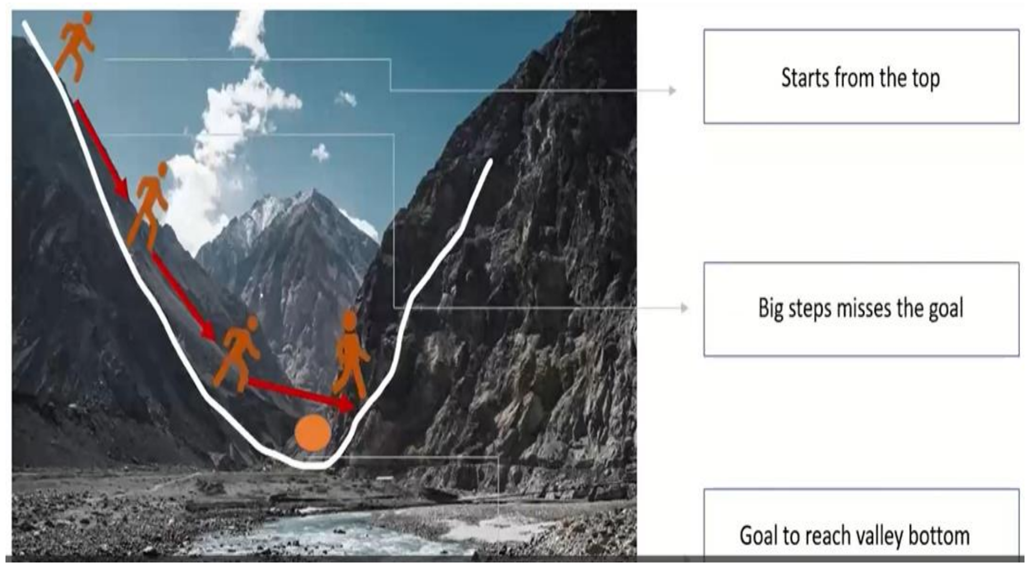


Figure-b

Gradient Descent Algorithm: look at Scenario…

- ❖ **So, what will do:**
  - ❑ *Big step slope is steep.*
  - ❑ *Small steps when near to goal.*
  - ❑ *The slope changes will every step.*

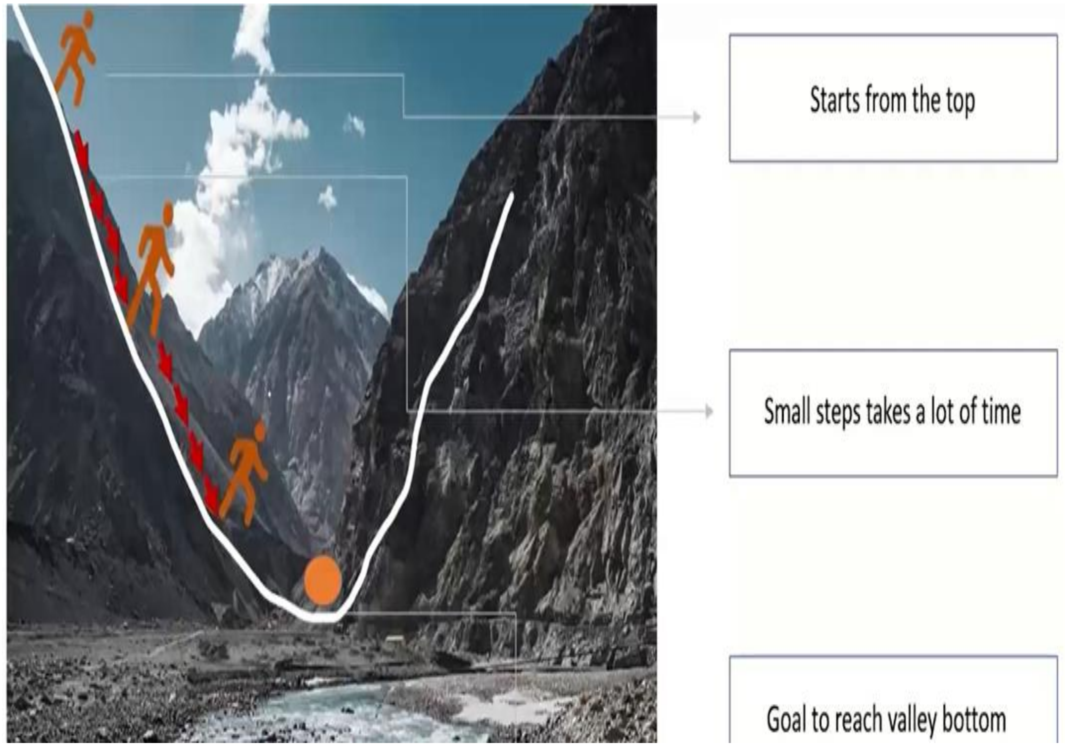


Figure-c

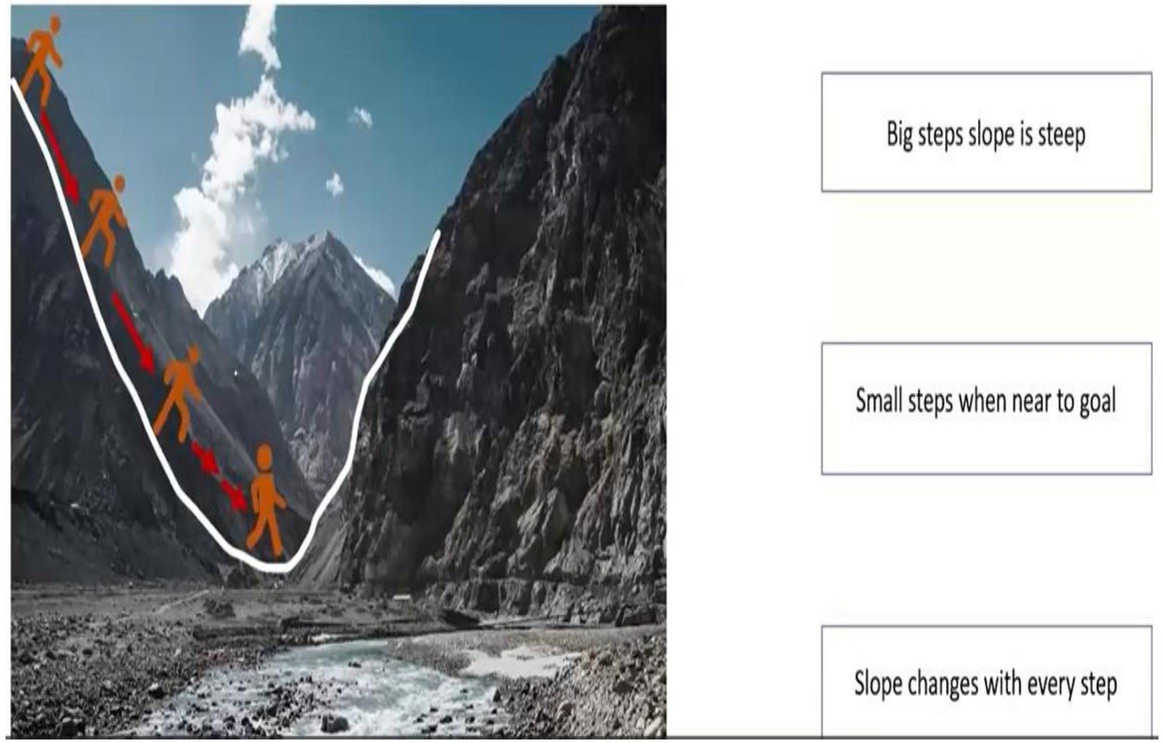


Figure-d

Gradient Descent Algorithm: look at Scenario…

- ❑ In the GDA , the number of steps you take is the determined by ***learning rate*** ***(Fig e)***, and this decides how fast the algorithm converges to the minima.
- ❑ Error will be minimized.

- ❑ Let's plot the graph and understand the slope.
- ❑ How does the boy reach the next step?
- ❑ The slope changes at every position.
- ❑ The arrow indicates the direction. This process continues until the boy reaches the bottom of the pit.
- ❑ Consider the graph

Slope gives direction

Step size determined by learning rate

Errors will be minimized.

Figure-e

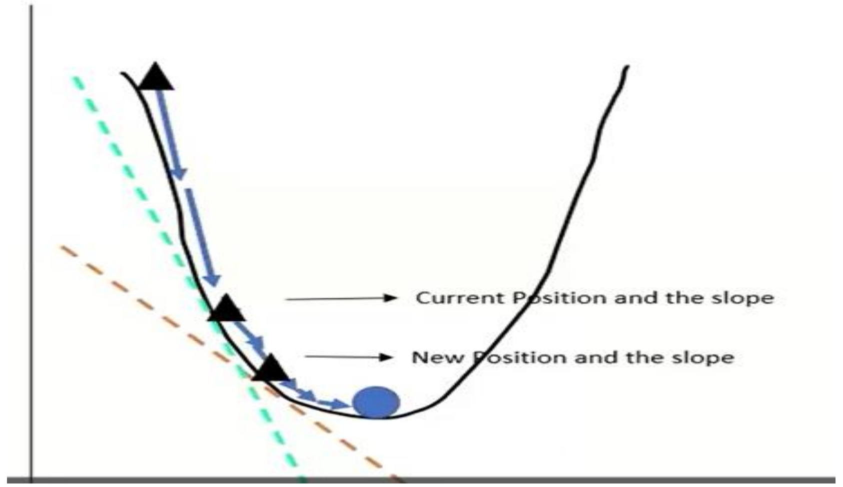


Figure-f

Gradient Descent for Linear Regression

❖Now let's apply all this GDA for linear regression.

$$J(a) = \frac{1}{2n} \sum_{i=1}^{n} \left(y_{i\,(actual)} - \left(a_0 + a_1 x_1^{(i)}\right)\right)^2$$

$$\partial/\partial_{a1} = \frac{2}{n} \sum_{i=1}^{n} -x_i \left(y_i - (a_1 x_i + a_0)\right)$$

$$\partial/\partial_{a0} = \frac{2}{n} \sum_{i=1}^{n} -y_i \left(-(a_1 x_i + a_0)\right)$$

Partial Derivative with respect to slope(a_1) and intercept(a_0)

❑ **Next step. Let's calculate learning rate**

$$a_1 = a_1 - \text{learning rate} * \partial/\partial_{a1}$$

$$a_0 = a_0 - \text{learning rate} * \partial/\partial_{a0}$$

Learning rate

Gradient Descent for Linear Regression…

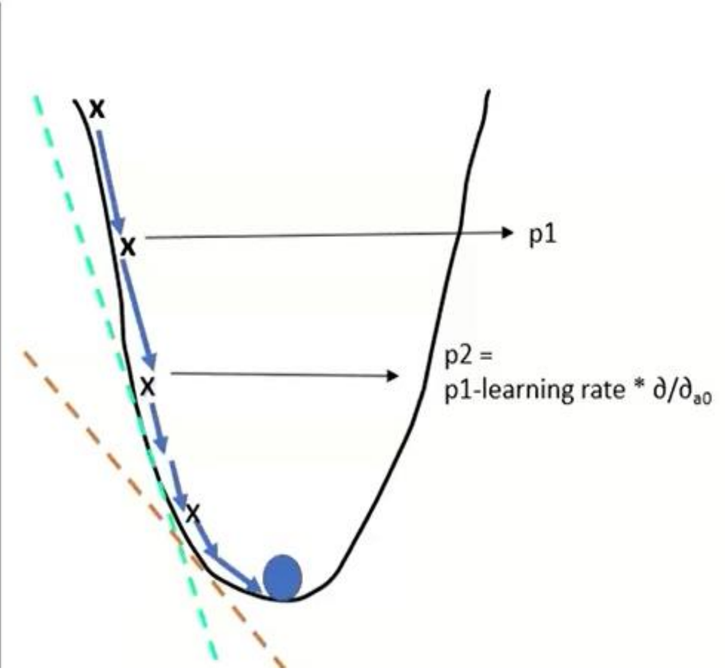
❖**Gradient descent** updating **a0** and **a1** to minimize the cost function **(MSE).**

❖A **regression model** uses gradient descent to update the coefficients of the line (***a0, a1 => xi***) by reducing the cost function by a random selection of coefficient values and then iteratively update the values to reach the minimum cost function.

❑**Consider the graph.**

- ❑The point are learning rate. Initially ***a0*** and ***a1*** are zero.
- ❑P1-represent (point 1)
- ❑P2(point 2) is calculated from p1.

❑This formula applies to understanding the inner algorithm and how to find an ***error minimized*** and the ***cost function is optimized.***



❑ ***Note: Later session we apply this concept and implement it in practice.***

ii. Polynomial Regression

- **Polynomial Regression** is a regression algorithm that models the relationship between a ***dependent(y)*** and ***independent variable(x)*** as nth degree polynomial.
- The Polynomial Regression equation is given below:

$$y= b_0+b_1x_1+ b_2x_1^2+ b_2x_1^3+\ldots\ldots b_nx_1^n$$

- **Polynomial Regression** is add some polynomial terms to the *Multiple Linear regression equation* to convert it into ***Polynomial Regression***.
- It is a linear model with some modifications in order to increase the accuracy.
- The dataset used in Polynomial regression for training is of a ***non-linear nature.***
- It makes use of a ***linear regression model*** to fit the complicated and ***non-linear functions and datasets.***

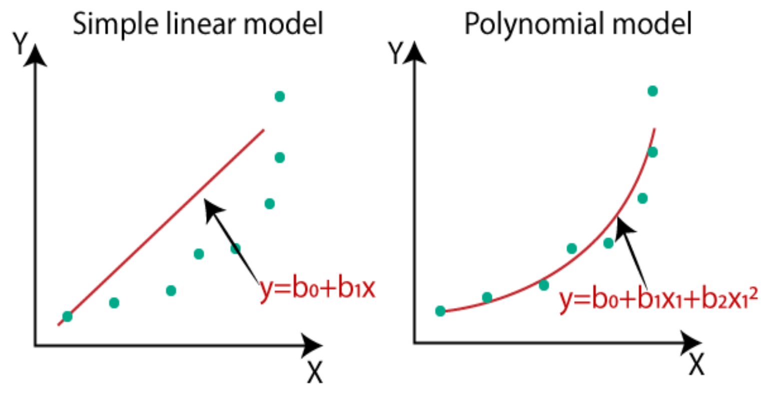
Need for Polynomial Regression:

❖The need of **Polynomial Regression** in ML can be understood in the below points:

- If we apply a linear model on a **linear dataset**, then it provides us a good result as we have seen in ***Simple Linear Regression***, but if we apply the same model without any modification on a ***non-linear dataset***, then it will produce a ***drastic output*.**
- Due to which loss function will increase, the *error rate* will be ***high*,** and accuracy will be decreased.
- So for such cases, **where data points are arranged in a non-linear fashion, we need the Polynomial Regression model**.

Simple Linear Model vs Polynomial Model

- If the relationship between the independent and dependent variable is ***not linear***, then linear regression cannot be used as it will result in large errors.
- As shown in the figure.



- In the above figure, we have taken a ***dataset*** which is arranged ***non-linearly.***
- So if we try to cover it with a ***linear model***, then we can clearly see that it hardly covers any data point.
- On the other hand, a curve is suitable to cover most of the data points, which is of the ***Polynomial model.***
- Hence, *if the datasets are arranged in a non-linear fashion, then we should use the* **Polynomial Regression model** *instead of Simple Linear Regression.*

Equation of the Polynomial Regression Model:

- **Simple Linear Regression equation:** $y = b_0+b_1x$ ....(a)
- **Multiple Linear Regression equation:** $y= b_0+b_1x+ b_2x_2+ b_3x_3+....+ b_nx_n$ ....(b)
- **Polynomial Regression equation:** $y= b_0+b_1x + b_2x^2+ b_3x^3+....+ b_nx^n$ ....(c)

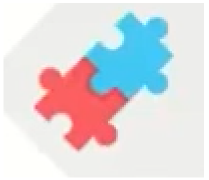
- When we compare the above three equations, we can clearly see that all three equations are Polynomial equations but differ by the degree of variables.
- The Simple and Multiple Linear equations are also Polynomial equations with a single degree, and the Polynomial regression equation is Linear equation with the nth degree.
- So if we add a degree to our linear equations, then it will be converted into Polynomial Linear equations.

Note: A **Polynomial Regression algorithm** is also called Polynomial Linear Regression because it does not depend on the variables, instead, it depends on the coefficients, which are arranged in a linear fashion.

iii. Regularization

- ❑ Sometimes what happens is that our Machine learning model performs well on the ***training data but does not perform well on the unseen* or *test data*.**
- ❑ It means the model is not able to predict the output or target column for the unseen data by introducing noise in the output, and hence the model is called an **overfitted** model.

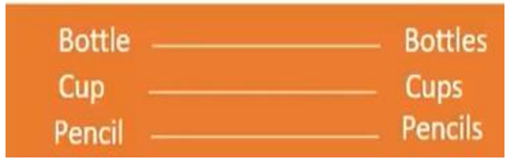
❖ Needs for Regularization



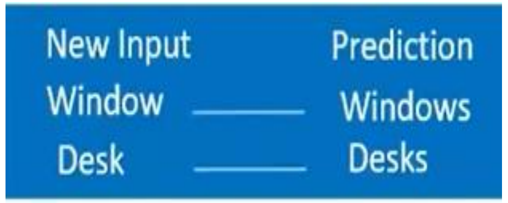
❖Example:

❑Input singular nouns and get the plural nouns for any word.

❑Then, train the model and it learns the pattern that's needs to be added at the end of every word.



Bottle	Bottles
Cup	Cups
Pencil	Pencils



New Input	Prediction
Window	Windows
Desk	Desks

❑ *But the model fails to predict plural nouns for the words like*

	incorrect	correct
box	boxs	boxes
ban	mans	men
beaf	leafs	leaves

Over generalization or overfitting

❑ Question is how do overcome overfitting to make good model? Answer is Regularization

❖What is Regularization?

- ❑ Regularization in machine learning prevents the model from ***overfitting.***
- ❑ It helps the machine to learn more than just ***memorize***.
- ❑ It is a form of regression that ***reduces*** or ***shrinks*** the coefficient features towards zero.
- ❑ In other words, this technique forces us not to learn a more complex or flexible model, to avoid the problem of overfitting.

❖ **Now, let's understand the "How flexibility of a model is represented?"**

- ❑ For regression problems, the increase in flexibility of a model is represented by an increase in its ***coefficients*****,** which are calculated from the ***regression line.***
- ❑ *In simple words, "In the* ***Regularization technique,*** *we reduce the magnitude of the independent variables by keeping the same number of variables".*
- ❑ It maintains accuracy as well as a generalization of the model.

How does Regularization Work?

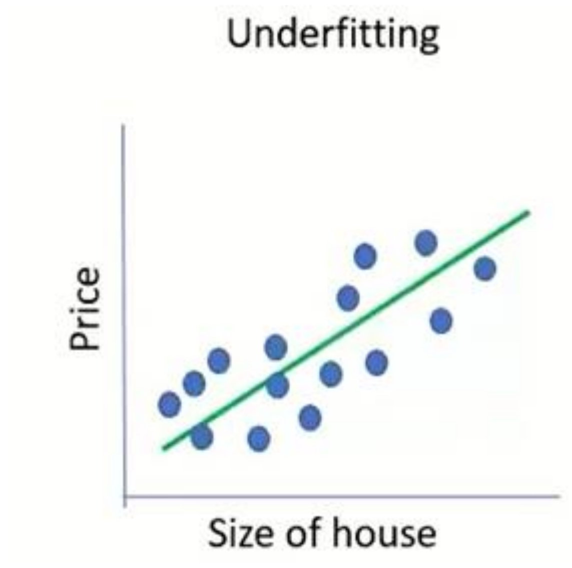
- Regularization works by adding a penalty or complexity term or shrinkage term with ***Residual Sum of Squares (RSS)*** to the complex model.
- Let's consider the **Simple linear regression** equation:
- Here ***Y represents*** the dependent feature. Y is approximated to $\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$
- Here, $X_1, X_2, \ldots X_p$ are the independent features or predictors for Y, and
- $\beta_0, \beta_1, \ldots \beta_n$ represents the coefficient estimates for different variables or ***predictors(X),***
- which describe the weights or magnitude attached to the features, respectively.
- In simple linear regression, our optimization function or loss function is known as the ***residual sum of squares (RSS).***
- ***We choose those sets of coefficients,***

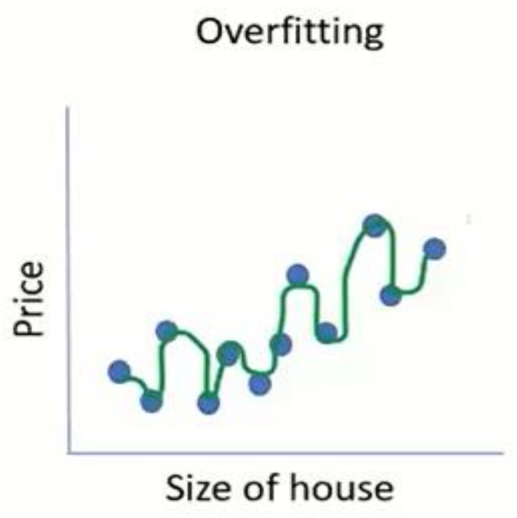
  such that the following loss function is minimized:

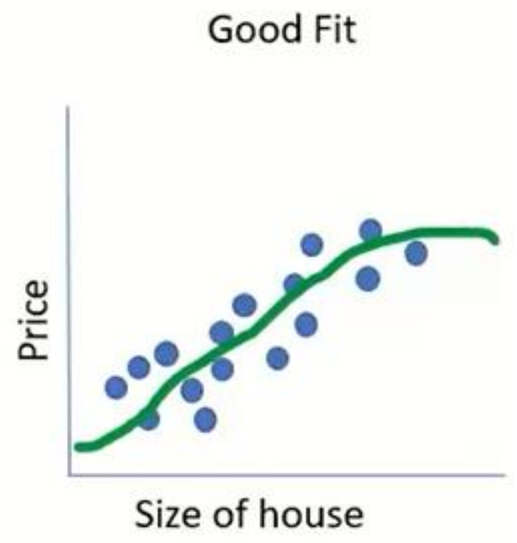
$$\mathrm{RSS} = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2.$$

How does Regularization Work?

❑ **Example:**







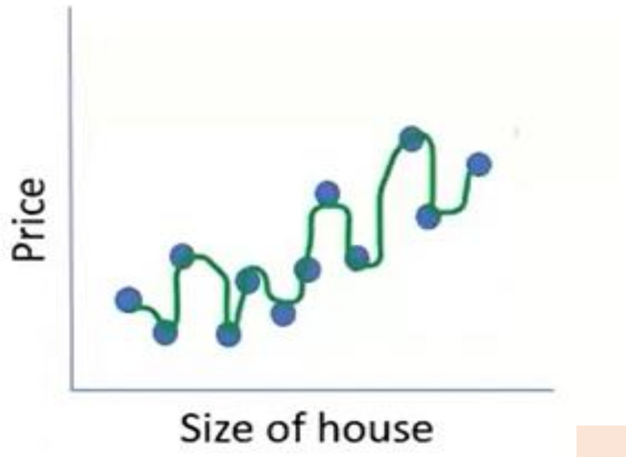
Price=$\beta_0 + \beta_1$*size

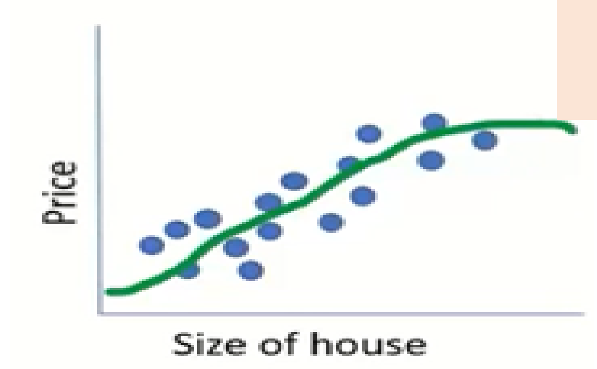
Price=$\beta_0 + \beta_{1*}$size + β_{2*}size + β_{3*}size+β_{4*}size

Price=$\beta_0 + \beta_{1*}$size + β_{2*}size

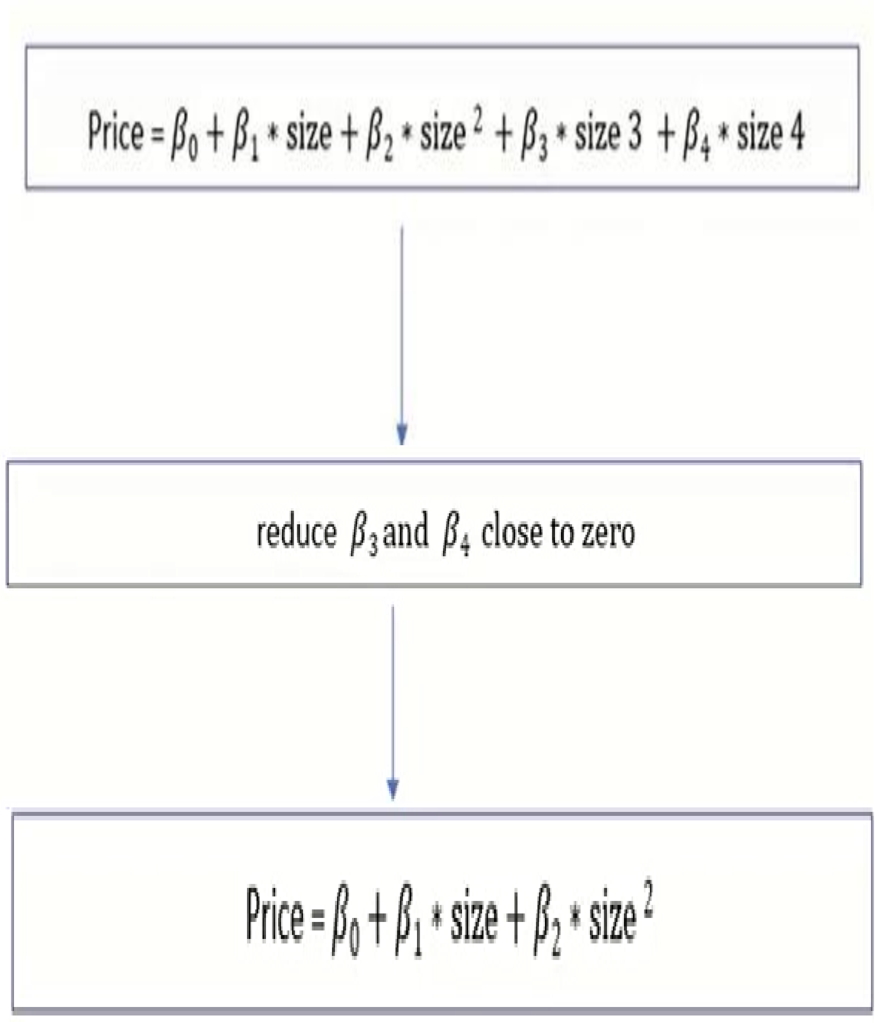
Note: Good fit model balance between underfitting and overfitting

How does Regularization Work?





- Reduces or shrinks the coefficient features towards zero.



$$\begin{tikzcd}
\boxed{\text{Price} = \beta_0 + \beta_1 * \text{size} + \beta_2 * \text{size}^2 + \beta_3 * \text{size } 3 + \beta_4 * \text{size } 4} \arrow[d] \\
\boxed{\text{reduce } \beta_3 \text{ and } \beta_4 \text{ close to zero}} \arrow[d] \\
\boxed{\text{Price} = \beta_0 + \beta_1 * \text{size} + \beta_2 * \text{size}^2}
\end{tikzcd}$$

How does Regularization Work?

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_{i\,(\text{actual}} - y_{i\,(\text{predicted})}\right)^{2}$$

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_{i\,(\text{actual}} - \beta_j(x_i)\right)^{2}$$

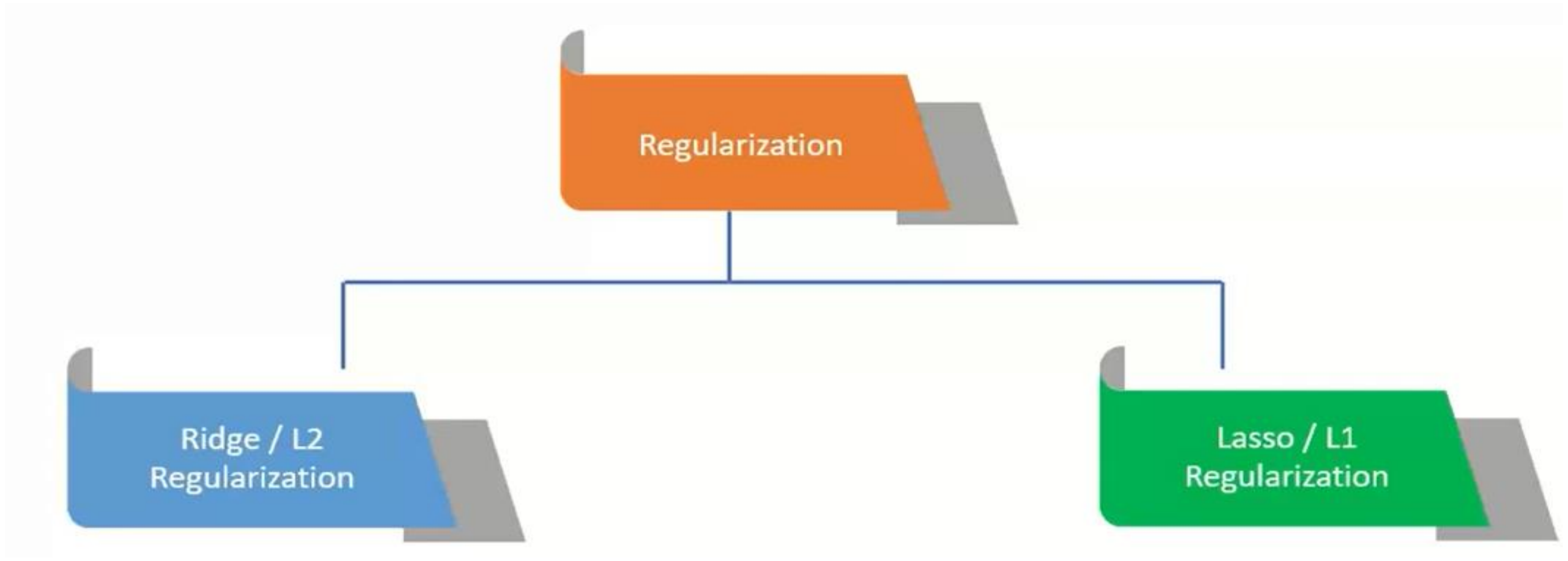
$$\beta_j(x_i) = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2{}^{2} + \beta_3 * x_3{}^{3} + \cdots$$

- **The aim is to reduce the mean squared error value**

- **We use different regularization techniques | L1 and L2**

Regularization techniques

- Mainly, there are two types of regularization techniques, which are given below:



a. Ridge|L2 Regularization

☞ **Ridge Regularization** is one of the types of **linear regression** in which we introduce a small amount of ***bias***, known as ***Ridge regression penalty.***

☞ so that we can get better long-term predictions.

☞ In Statistics, it is known as the ***L-2 norm*****.**

☞ In this technique, the ***cost function*** is altered by adding the ***penalty term*** (shrinkage term), which multiplies the ***lambda*** or tuning parameter with the squared weight of each individual feature.

☞ Therefore, the optimization function(cost function) becomes:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_{i\,(actual} - y_{i\,(predicted)})^2$$

$$Loss = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{P}\beta_j^{\,2}$$

λ = Tuning parameter

b. Lasso|L1 Regularization

☞ **Lasso Regularization** is used to reduce the complexity of the model.

☞ It is similar to the Ridge **Regularization** except that the penalty term includes the **absolute weights** instead of a square of weights.

☞ In statistics, it is known as the ***L-1 norm***.

❑ **Therefore, the optimization function becomes:**

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_{i\,(actual} - y_{i\,(predicted)})^{2}$$

$$Loss = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \longrightarrow$$

Penalty term regularizes the coefficients

λ = Tuning parameter

Key Differences between Ridge and Lasso Regression

☞***Ridge Regularization*** helps us to reduce only the ***overfitting*** in the model while keeping all the features present in the model.

☞ It reduces the complexity of the model by shrinking the coefficients whereas

☞ ***Lasso Regularization*** helps in reducing the problem of overfitting in the model as well as automatic feature selection.

☞ ***Lasso Regularization*** tends to make coefficients to absolute zero whereas ***Ridge regression*** never sets the value of the coefficient to absolute zero.

Understand the metrics used to Evaluate Regression

- ❑Some of the most popular regression evaluation metrics can help to assess the effectiveness of the model.

i. Mean squared error (MSE)

- ❑ MSE is one of the most popular evaluation metrics.
- ❑As shown in the following formula
- ❑ MSE is closely related to the ***residual sum*** of squares.
- ❑ The difference is that you are now interested in the *average error* instead of the total error.

$$MSE = \frac{1}{N}\Sigma_{i=1}^{N}(y_i - \hat{y}_i)^2$$

- ❑ MSE uses the mean (instead of the sum) to keep the metric independent of the dataset size.

Understand the metrics used to Evaluate Regression…

ii. Root mean squared Error (RMSE)

- ❑ **RMSE** is closely related to **MSE,** as it is simply the square root of the latter.
- ❑Take the square to bring the metric back to the scale of the target variable, so it is easier to interpret and understand.

$$RMSE = \sqrt{\frac{1}{N}\Sigma_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

iii. Mean absolute error (MAE)

- ❑ MAE is similar to the MSE formula.
- ❑Replace the square with the absolute value.

$$MAE = \frac{1}{N}\Sigma_{i=1}^{N}|y_i - \hat{y}_i|$$

- **A Case Study in regression**

Classification

❖What is Classification in ML?

- ❑**Classification** is a Supervised Learning technique that is used to identify *a given dataset*
- ❑Unlike ***regression,*** the output variable of Classification is a ***category***
- ❑In Classification, a program learns from the given ***dataset*** or ***observations*** and then classifies new observations into a number of **classes** or ***groups***.
  - ❖*Such as, **Yes or No**, **0 or 1**, **Spam or Not Spam**, **cat or dog**, etc.*
- ❑In a classification problem, the result is predicted in a ***discrete output***.
- ❑Let **X, Y** (Pre-classified training examples)
  - ❑**Classification** when Y is ***discrete.***

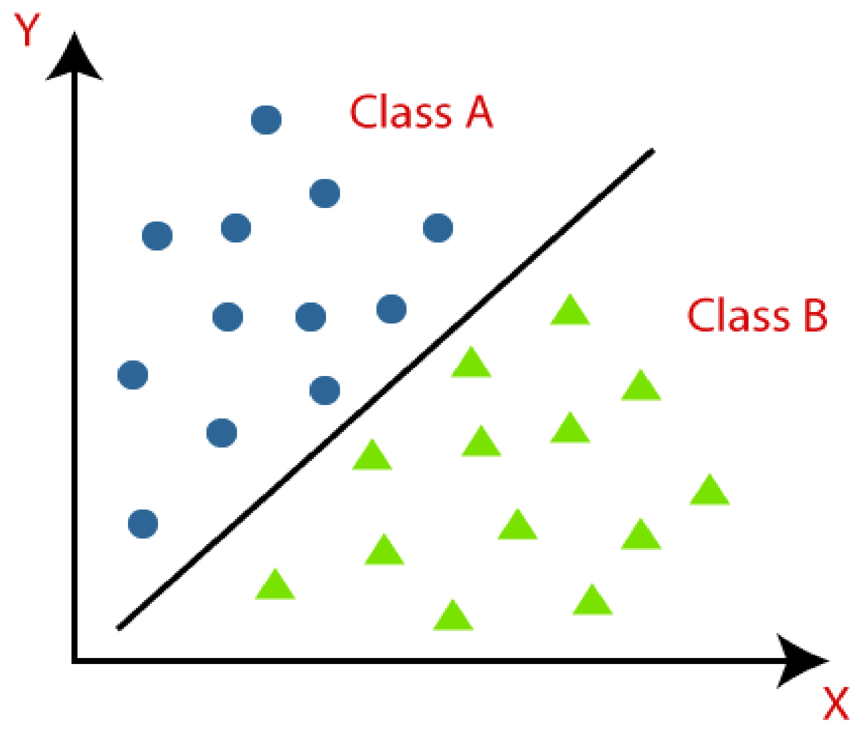
Classification…

- ***Classification algorithms*** can be better understood using the given diagram.

- In the diagram, there are two classes, ***class A*** and ***Class B.***
- *These classes have features that are similar to each other and dissimilar to other classes.*

Best Example of ML classification algorithm

- Used to predict whether
  - **Credit Scoring:** Differentiating between **low-risk** and **high-risk** customers from their income and savings.
  - Breast cancer (malignant or Benign)
  - a **new email** is *spam* or *not spam*,



Classification...

❖The algorithm which implements the classification on a dataset is known as a ***classifier.***

❖ **There are two types of Classifications:**

❑***Binary Classifier:*** If the classification problem has only two possible outcomes, then it is called a ***Binary Classifier***.

❑**Examples:** YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.

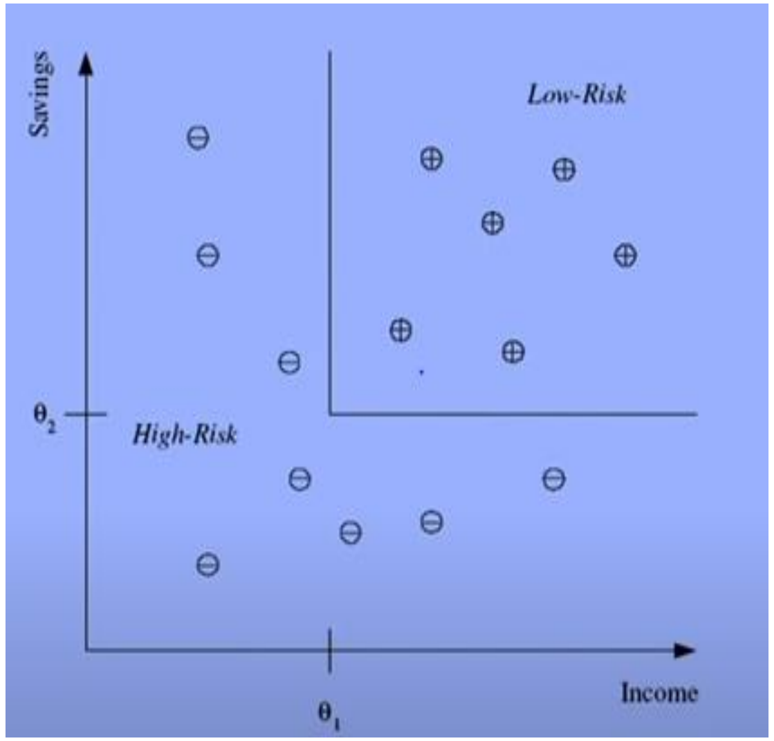
❑***Multi-class Classifier:*** If a classification problem has more than two outcomes, then it is called a ***Multi-class Classifier***.

❑**Example:** Classifications of types of *crops*, Classification of types of music.

Classification: Ex #1- Credit Scoring

- ❖Let us look at some examples of classification problems.
- ❖**Example: Credit Scoring:**
- ❖Differentiating between **low_risk** and **high_risk** customers from their income and savings.

- ❖**So, How to solve this problem?**
  - ❑Suppose in this problem we are describing each input in terms of two features, ***income*** and ***Savings***.
  - ❑We have a person, we look at the ***income*** and ***savings*** of a person and we want to predict whether this is a ***high-risk person*** or a ***low-risk person***.



- ❖ As shown in the figure the high-risk persons are labeled with ***minus*** and the ***low-risk*** persons are labeled with ***plus***.

Classification: Ex #1- Credit Scoring

❖Now given this data you want to come up with a classifier that given the attributes of a new person will predict whether that person is ***high-risk*** or ***low-risk***.

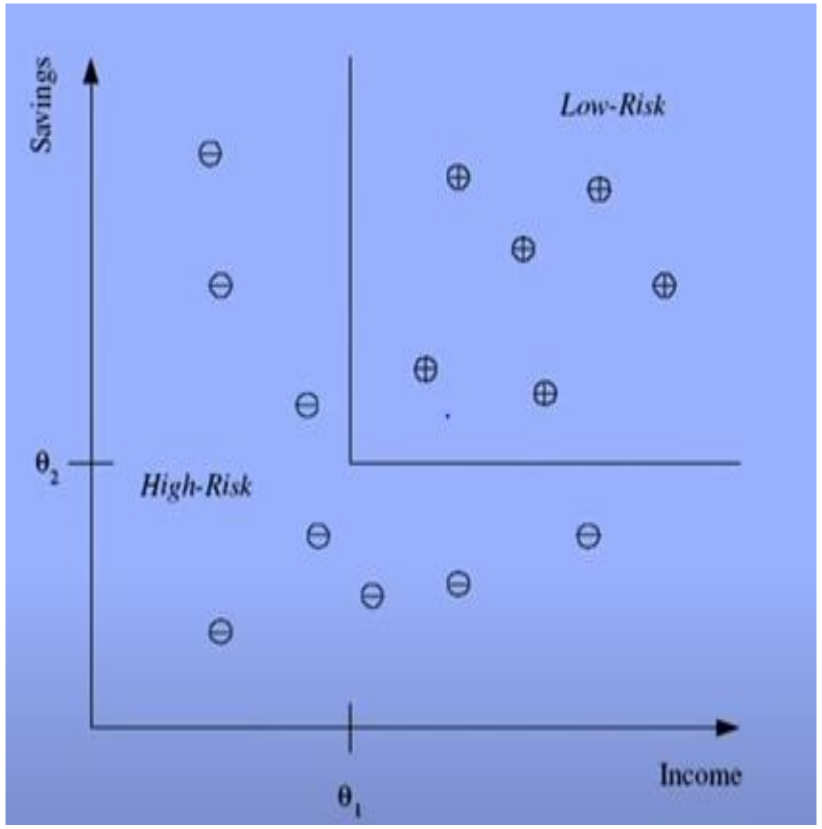
❖**Discriminant rule:**

If Income >θ1 AND Savings > θ2

Then low-risk Else high-risk

❖This rule is come up with the rule based on the data.

❖In this case, you can visualize the data and income with the rule.



Learners in Classification Problems:

❖In the classification problems, there are two types of learners:

❖***Lazy Learners:***

- ❑ Lazy Learner first stores the training dataset and waits until it receives the test dataset.
- ❑ In the Lazy learner case, classification is done on the basis of the most related data stored in the training dataset.
- ❑ It takes less time in training but more time for predictions.

  Example: *K-NN algorithm, Case-based reasoning*

❖***Eager Learners:***

- ❑Eager Learners develop a classification model based on a training dataset before receiving a test dataset.
- ❑*Opposite to Lazy learners, Eager Learner takes more time in learning and less time in prediction.*
- ❑***Example:*** *Decision Trees, Naïve Bayes, ANN.*

Understand the operation of classifiers

❖Classification Algorithms can be further divided into the Mainly two main categories:

❖***Linear Models***

❑*Logistic Regression*

❑*Support Vector Machines*

❖***Non-linear Models***

❑*K-Nearest Neighbors (KNN)*

❑*Naïve Bayes*

❑*Decision Tree Classification*

❑*Random Forest Classification*

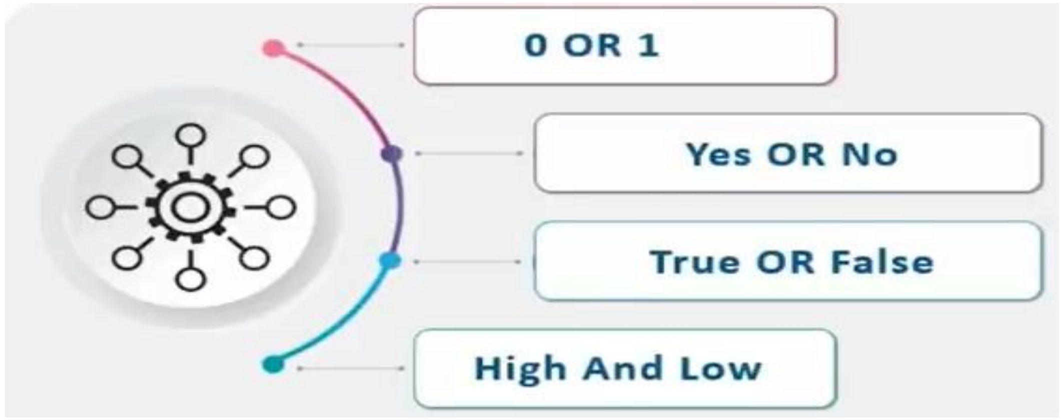
❑ Logistic Regression

What and Why?

❖**Logistic Regression** is used for predicting the *categorical dependent variable* using a given set of ***independent variables.***

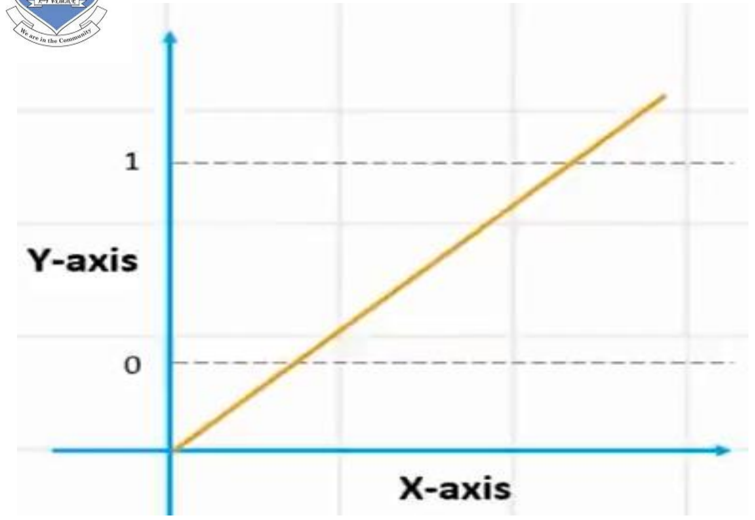
❖***The results in a binary format which is used to predict the out come of categorical dependent variable.***

❖So, the outcome should be a ***categorical*** or ***discrete value such as:***



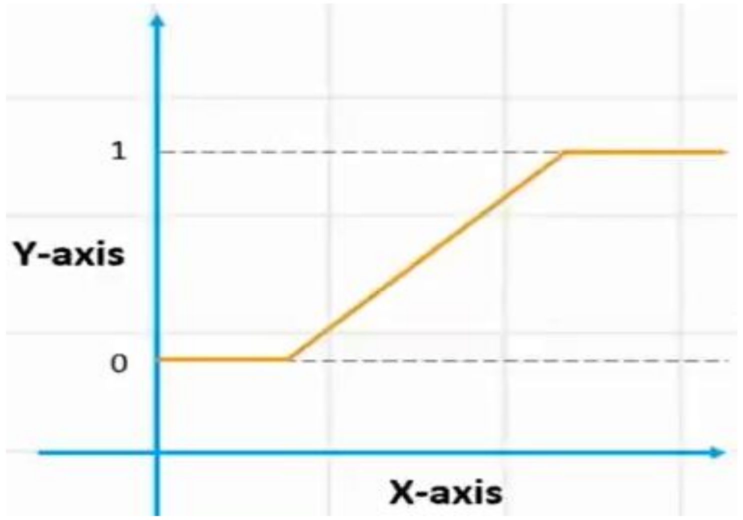
Note: *Logistic regression uses the concept of* ***predictive modeling*** *as regression; therefore, it is called logistic regression, but is used to classify samples;*
Therefore, it falls under the classification algorithm.

Why not Linear Regression



- Fig 1

- Because *We don't need the value below zero and above 1*
- In logistic regression, the value of Y will be between 0 and 1,
- The linear line has to be clipped at 0 and 1.



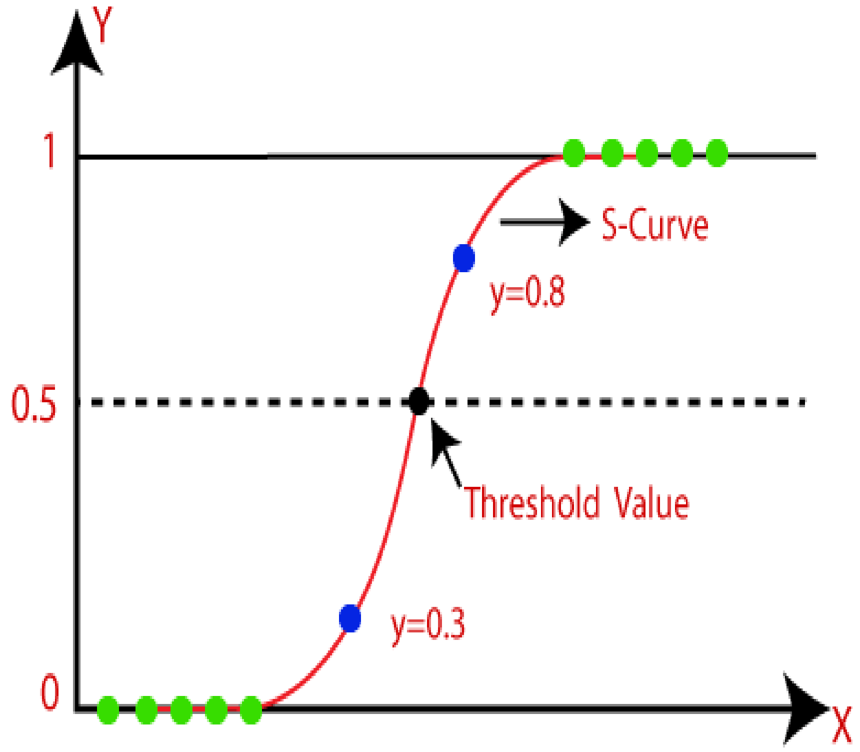
- Fig 2

- Here in the figure two the outcome is either 0 or 1.
- With this, our results cannot be formulated into a single formula.
- Hence we came up with logistic

Logistic Regression Curve

- ❑In Logistic regression, instead of fitting a ***regression line,***
- ❑we fit an ***"S"*** shaped logistic function, which predicts two maximum values (***0 or 1***).
- ❑***We call the line Sigmoid S-Curve***

- ❑ The curve indicates the ***likelihood of something.***
- ❑ ***The threshold value*** indicates the probability of either ***0 or 1.***



- ❑ The figure shows the logistic Regression curve:

- ❑ ***Logistic Regression*** is a significant ML algorithm because it has the ability to provide probabilities and classify new data using ***continuous*** and ***discrete datasets***.

Logistic Function (Sigmoid Function):

- The ***sigmoid function*** is a mathematical function used to map the ***predicted values* to probabilities.**
- It maps any real value into another value within a range of ***0*** and ***1***.
- The value of the ***logistic regression*** must be between **0** and 1, which cannot go beyond this limit, so it forms a curve like the "S" form.
- The **S-form curve** is called the ***Sigmoid function*** or the ***logistic function.***
- In **logistic regression,** The values **above the threshold value** tend to be **1**, and a value ***below the threshold*** values tends to **0.**

Logistic Regression Equation

- The Logistic regression equation can be obtained from the Linear Regression equation.
- Equation of the straight line can be written as:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$

- In **Logistic Regression y** can be between 0 and 1 only,
- so for this let's divide the above equation by **(1-y):**

$$\frac{y}{1-y}\text{; 0 for y= 0, and infinity for y=1}$$

- But we need a range between **[0] to +[infinity],** then take the ***logarithm*** of the equation it will become:

- $$log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n$$ **final equation for Logistic Regression.**

Type of Logistic Regression:

❑On the basis of the categories, Logistic Regression can be classified into three types:

i. Binomial: in this type, there can be only two possible types of dependent variables, such as ***0 or 1, Pass or Fail,*** etc.

ii. Multinomial: in this type, there can be 3 or more possible unordered types of the dependent variable, such as ***"cat", "dogs", or "sheep"***

iii. Ordinal: in this type, there can be 3 or more possible ordered types of dependent variables, such as ***"low", "Medium", or "High".***

Note: lab section: Implementation of Logistic Regression

Linear Regression vs Logistic Regression

Linear Regression

- Continuous variable
- Solve regression problem
- Straight line
- Example: weather prediction
- ***In linear regression predicting what will be the weather tomorrow.***

Logistic Regression

- Categorical variables
- Solve classification problem problem
- S-Curve or
- *Sigmoid function*
- ***Example:*** weather prediction
- **But in logistic regression only tell if it rains or not. Cloudy or not etc.**

❖ K-Nearest Neighbor(KNN)

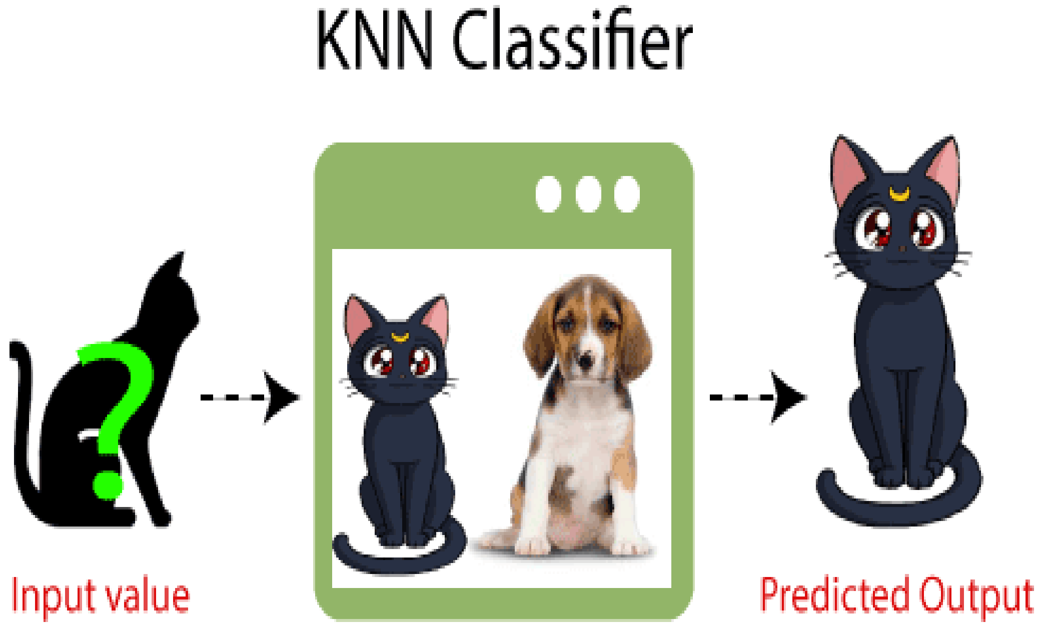
- ❑ ***K-NN algorithm*** assumes the similarity between the ***new case/data*** and ***available cases*** and puts the new case into the category that is most similar to the available categories.
- ❑ The ***k-NN algorithm*** stores all the available data and classifies a new data point based on the similarity measure.
  - ✓ *This means when new data appears then it can be easily classified into a well-suited category by using the K- NN algorithm.*
- ❑ **K-NN algorithm** can be used for Regression as well as for Classification but mostly it is used for Classification problems.

K-Nearest Neighbor(KNN) ...

- **K-NN** is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
- It is also called a **lazy learner algorithm** because
  - *it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.*
- **KNN algorithm** at the ***training phase just stores the dataset*** and when it gets new data, it classifies that data into a category that is ***similar to the new data.***

KNN- Example

- ❑Suppose, we have an image of a creature that looks similar to ***cat*** and ***dog***, but we want to know either it is a ***cat or dog***.

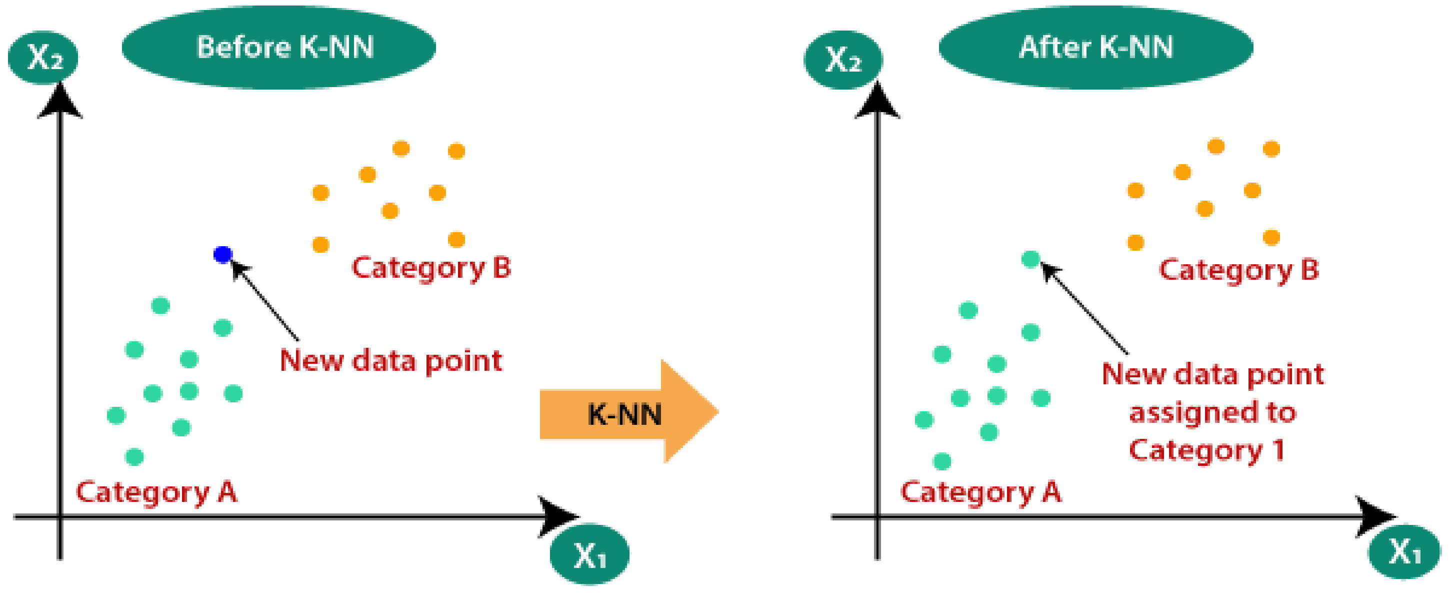


- ❑*So for this identification, we can use the KNN algorithm, as it works on a similarity measure.*
- ❑*Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either the cat or dog category.*

Why do we need a K-NN Algorithm?

- ❑Suppose there are two categories, i.e., ***Category A*** and ***Category B***, and we have a ***new data point x1***, so this data point will lie in which of these categories?
- ❑To solve this type of problem, we need a ***K-NN algorithm.***
- ❑With the help of **K-NN**, we can easily identify the category or class of a particular dataset.

❑Consider the diagram:



How does K-NN work?

❑The K-NN working can be explained on the basis of the below algorithm:

Step-1: Select the number ***K*** of the neighbors

Step-2: Calculate the ***Euclidean distance*** of **K number of neighbors**

Step-3: Take the *K nearest neighbors* as per the calculated ***Euclidean distance.***

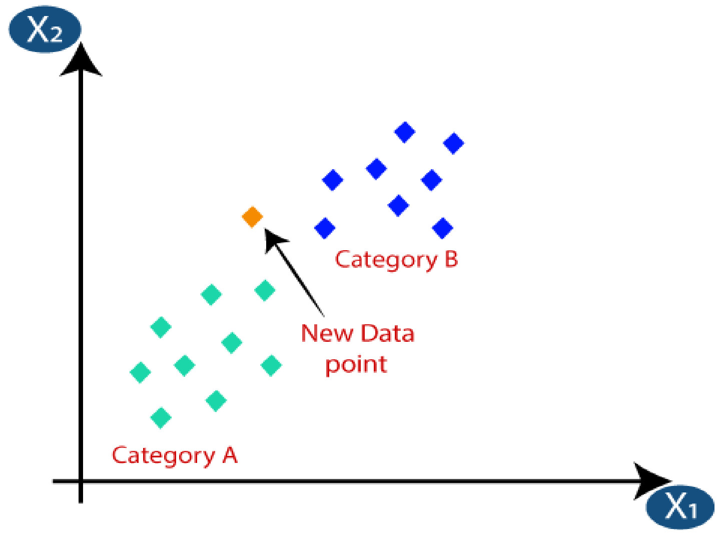
Step-4: Among these ***k neighbors,*** count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.

Example: How does K-NN work?

- Suppose we have a new data point and we need to put it in the required category.
- Consider the below image:

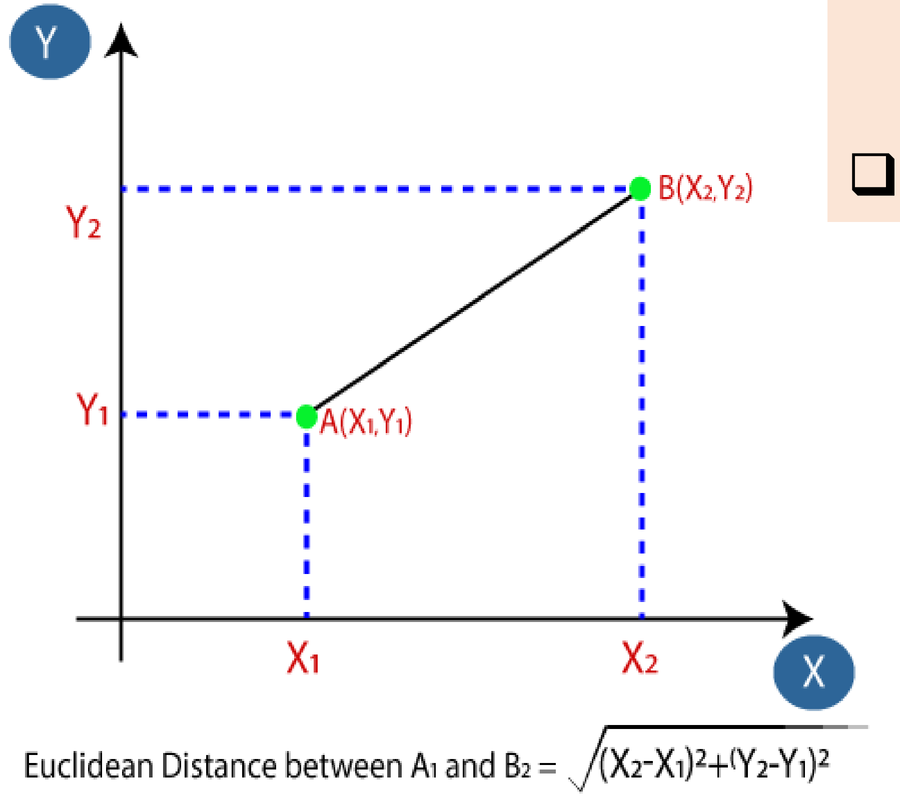


- ***Firstly,*** choose the number of neighbors, let's choose ***k=5.***
- Next, calculate the **Euclidean distance** between the data points.

- The ***Euclidean distance*** is the distance between two points, which we have already studied in geometry.

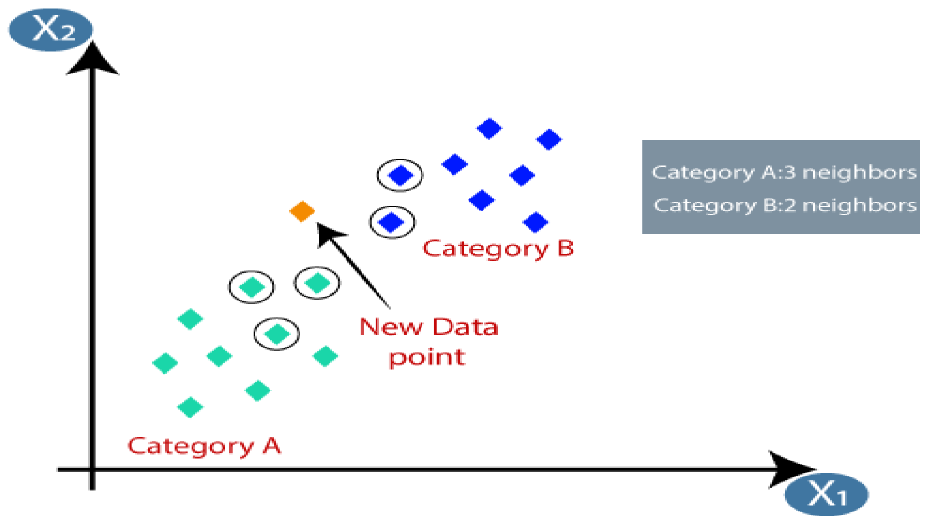
Example: How does K-NN work?...

- It can be calculated as:



Euclidean Distance between A_1 and $B_2 = \sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

- *By calculating the* ***Euclidean distance*** *we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B.*
- *Consider the below figure:*



- *As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.*

How to select the value of K?

❑**To select the value of K in the K-NN algorithm:**

- ❑There is no particular way to determine the best value for ***"K"***, so we need to try some values to find the *best out of them*.
- ❑The most preferred value for ***K is 5***.
- ❑A very low value for K such as ***K=1 or K=2***, can be noisy and lead to the effects of outliers in the model.
- ❑Large values for K are good, but it may find some difficulties.

Advantages of KNN Algorithm:

- ❑ *It is simple to implement.*
- ❑ *It is robust to noisy training data*
- ❑ *It can be more effective if the training data is large.*

Disadvantages of KNN Algorithm:

- ❑ *Always needs to determine the value of K which may be complex sometimes.*
- ❑ *The computation cost is high because of calculating the distance between the data points for all the training samples.*

Note: Lab session- implementation of the KNN algorithm

❑ Naïve Bayes

- ❑**Naïve Bayes algorithm** is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems.
- ❑It is mainly used in ***text classification*** that includes a *high-dimensional training dataset.*
- ❑***Naïve Bayes Classifier*** is one of the simplest and most effective Classification algorithms that help in building fast ML models that can make quick predictions.
- ❑It is a ***probabilistic classifier,*** which means it predicts on the basis of the probability of an object.
- ❑Some popular examples of Naïve Bayes Algorithm are ***spam filtration, Sentimental analysis, and classifying articles***.

Why is it called Naïve Bayes?

- The Naïve Bayes algorithm is comprised of two words ***Naïve*** and **Bayes**, Which can be described as:
- ***Naïve**: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features.*
- *Such as if the fruit is identified on the basis of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple.*
- Hence each feature individually contributes to identifying that it is an apple without depending on each other.
- **Bayes**: It is called Bayes because it depends on the principle of Bayes' Theorem.

Bayes' Theorem:

- Bayes' theorem is also known as **Bayes' Rule** or **Bayes' law**, which is used to determine the probability of a hypothesis with prior knowledge.
- It depends on the conditional probability.
- The formula for Bayes' theorem is given as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- **Where,**
  - ***P(A|B) is Posterior probability****:* Probability of hypothesis A on the observed event B.
  - ***P(B|A) is Likelihood probability****:* The probability of the evidence given that the probability of a hypothesis is true.
  - ***P(A) is Prior Probability:*** The probability of the hypothesis before observing the evidence.
  - **P(B) is Marginal Probability**: Probability of Evidence.

Working of Naïve Bayes' Classifier: Example

- Suppose we have a dataset of ***weather conditions*** and the corresponding target variable **"Play"**.
- So using this dataset we need to decide whether we should play or not on a particular day according to the weather conditions.
- ***So to solve this problem, we need to follow the following steps:***

  Step 1: *Convert the given dataset into frequency tables.*

  Step 2: *Generate a Likelihood table by finding the probabilities of given features.*

  Step 3: *Now, use Bayes theorem to calculate the posterior probability.*

❑***Problem:***

- *If the weather is sunny, then the Player should play or not?*

Solution: To solve this,

1st. consider the below dataset:

	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

2nd. frequency table for the Weather Conditions:

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	5

3rd. Likelihood table weather condition:

Weather	No	Yes	
Overcast	0	5	5/14= 0.35
Rainy	2	2	4/14=0.29
Sunny	2	3	5/14=0.35
All	4/14=0.29	10/14=0.71	

4th. Applying Bayes'theorem:

P(Yes|Sunny)= P(Sunny|Yes)*P(Yes)/P(Sunny)
P(Sunny|Yes)= 3/10= 0.3
P(Sunny)= 0.35
P(Yes)=0.71
So P(Yes|Sunny) = 0.3*0.71/0.35= 0.60

P(No|Sunny)=P(Sunny|No)*P(No)/P(Sunny)
P(Sunny|NO)= 2/4=0.5
P(No)= 0.29
P(Sunny)= 0.35
*So P(No|Sunny)=0.5*0.29/0.35 =* ***0.41***

- So as we can see from the above calculation that ***P(Yes|Sunny)>P(No|Sunny)***
- **Hence on a Sunny day, Player can play the game.**

Naïve Bayes Classifier:

Advantages

- ❑ It can be used for Binary as well as Multi-class Classifications.
- ❑ It performs well in Multi-class predictions as compared to the other Algorithms.
- ❑ It is the most popular choice for **text classification problems**.

Disadvantages

- ❑ Naive Bayes assumes that all features are independent or unrelated,
- ❑ so it cannot learn the relationship between features.

❖ Applications of Naïve Bayes Classifier:

- ❑ It is used for ***Credit Scoring***.
- ❑ It is used in ***medical data classification***.
- ❑ It can be used in ***real-time predictions*** because Naïve Bayes Classifier is an eager learner.
- ❑ It is used in Text classification such as *Spam filtering* and ***Sentiment analysis***.

Types of Naïve Bayes Model:

- ❖There are three types of Naive Bayes Model, which are given below:
- ❖***Gaussian:***
  - ❑In this model features follow a ***normal distribution.***
  - ❑This means if predictors take continuous values instead of discrete, then the model assumes that these values are *sampled from the Gaussian distribution*.
- ❖***Multinomial:***
  - ❑it is used when the data is multinomial distributed.
  - ❑It is primarily used for document classification problems, it means a particular document belongs to which category such as ***Sports, Politics, education***, etc.
  - ❑The classifier uses the frequency of words for the predictors.
- ❖***Bernoulli:***
  - ❑This model works similarly to the Multinomial classifier, but the predictor variables are the *independent Booleans variables.*
  - ❑Such as if a particular word is present or not in a document.
  - ❑This model is also famous for document classification tasks.

Note: lab session -Implementation of the Naïve Bayes algorithm

Decision Tree Classification

- ***Decision Tree*** *is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.*
- It is a tree-structured classifier, where ***internal nodes represent the features of a dataset, branches represent the decision rules*** *and* ***each leaf node represents the outcome.***
- In a ***Decision tree,*** there are two nodes, which are the ***Decision Node*** and ***Leaf Node.***
  - ✓ ***Decision nodes*** are used to make any decision and have multiple branches,
  - ✓ ***Leaf nodes*** are the output of those decisions and do not contain any further branches.
- *It* is used for both *classification* and *Regression problems*, but mostly it is preferred for ***solving Classification problems.***
- The *decisions* or the *tests* are performed on the basis of features of the given dataset.

Decision Tree Terminology

❑***Branch/Sub Tree:*** A tree formed by splitting the tree.

❑***Pruning:*** is the process of removing the unwanted branches from the tree.

❑***Splitting:***

❑Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.

❑***Parent/Child node:*** The root node of the tree is called the parent node, and other nodes are called the child nodes.

❑***Root Node:* is** from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.

❑***Leaf Node:*** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.

Why use Decision Trees?

- ❑There are various algorithms in ML, so choosing the *best algorithm* for the given *dataset* and
- ❑problem is the *main point of creating a machine-learning model.*
- ❑The two reasons for using the *Decision tree:*
  - ✓*Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.*
  - ✓*The logic behind the decision tree can be easily understood because it shows a tree-like structure.*

How does the Decision Tree Algorithm Work?

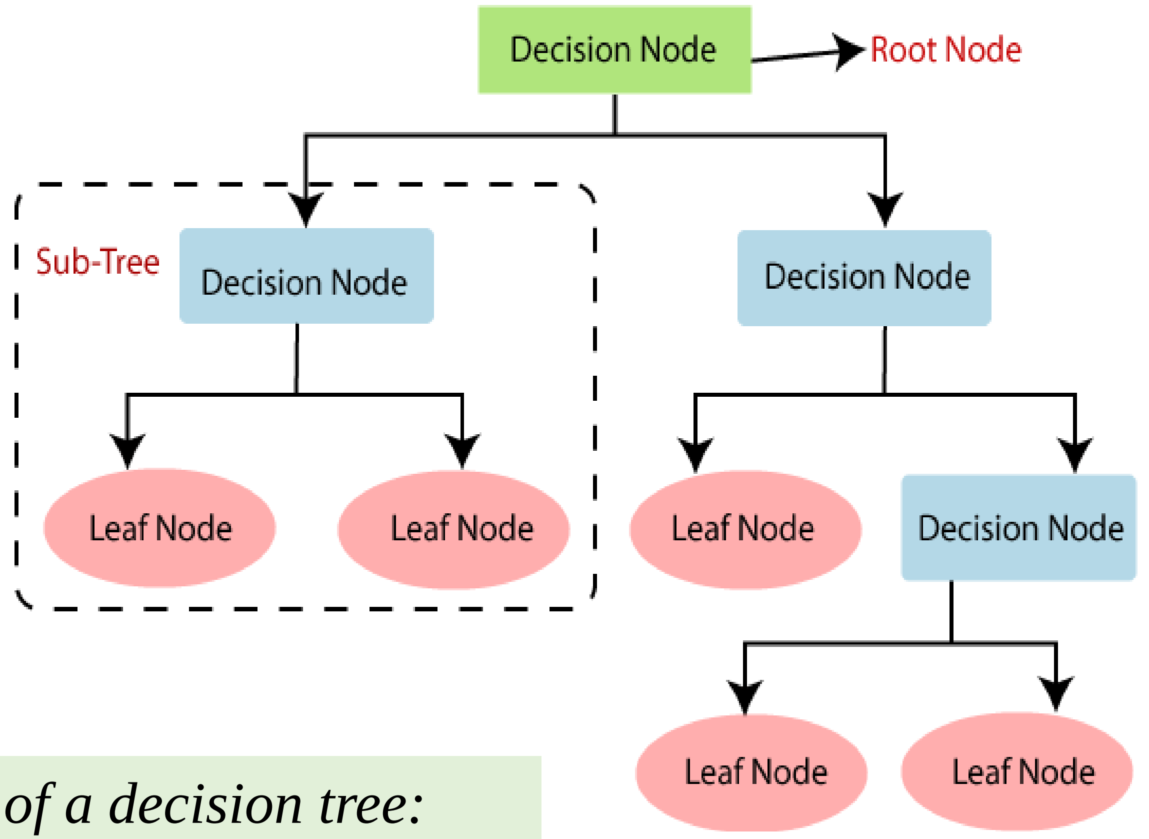
- In a decision tree, the algorithm starts from the root node of the tree.
- This algorithm compares the values of the ***root attribute*** with the record (real dataset) attribute and, based on the comparison, follows the *branch* and jumps to the ***next node.***
- For the next node, the algorithm again compares the attribute value with the other sub-nodes and moves further.
- ***The complete process can be better understood using the below algorithm:***

Step-1: *Begin the tree with the root node, says S, which contains the complete dataset.*
Step-2: *Find the best attribute in the dataset using* ***Attribute Selection Measure (ASM).***
Step-3: *Divide the S into subsets that contains possible values for the best attributes.*
Step-4: *Generate the decision tree node, which contains the best attribute.*
Step-5: *Recursively make new decision trees using the subsets of the dataset created in* ***step -3.*** *Continue this process until a stage is reached where you cannot further classify the nodes and call the final node as a leaf node.*

CART Algorithm

- In order to build a tree, we use the **CART algorithm**, which stands for ***Classification and Regression Tree algorithm.***

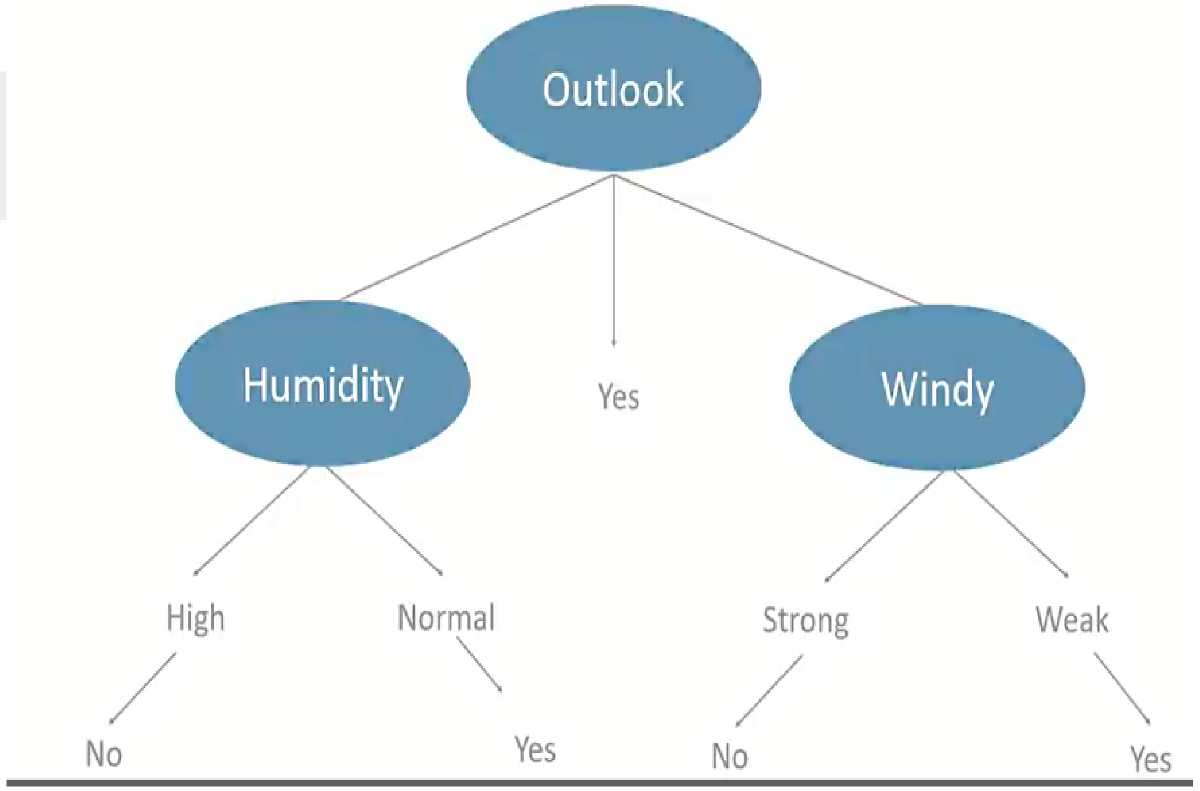
- A decision tree simply asks a question, and based on the answer *(Yes/No)*,
- it further splits the *tree* into *subtrees*.



- *The diagram explains the general structure of a decision tree:*

Let's first Visualize The Decision Tree

- ❑Which question to ask and when?



- ❑This decision tree create manual

Example: Learn About the Decision Trees

❑Data set

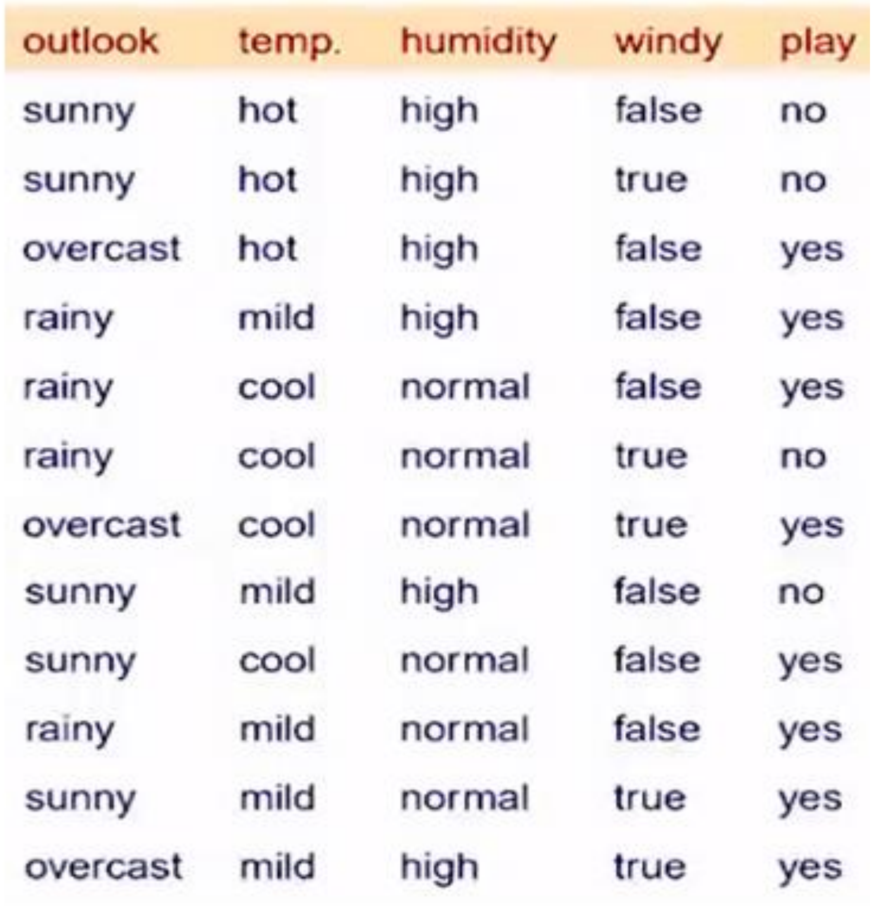
❑Which one among them should you pick first?

❑Answer: Determine the attribute that best classifies the training data.

❑But how do we choose the best attribute?

or

❑How does a tree decide where to split?



outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
Over cast	hot	normal	false	yes
Rainy	mild	high	true	no

Attribute Selection Measures

❑While implementing a ***Decision tree,*** the main issue arises as to how to select the best attribute for the ***root node*** and ***for sub-nodes.***

❑So, to solve such problems there is a technique which is called an ***Attribute selection measure or ASM.***

❑By this measurement, we can easily select the best attribute for the nodes of the tree.

❑There are some techniques/terminology we should know.

- ***Information Gain***
- ***Gini Index***
- ***Reduction in variance***
- ***Chi-Square***

How Does A Tree Decide Where To Split?

❑Before move to split. Let us know some terminologies

1st. Gini Index

- ❑ *The measure of impurity (or purity) used in building Decision tree in **CART***

.

4th. Chi-Square

- ❑ *It is an algorithm to find out the statistical significance between the differences between sub-nodes and parent node.*

2nd. Information Gain

- ❑ *The information gain is the decrease in **entropy** after a dataset is split on the basis of an attribute.*
- ❑ *Constructing a decision tree is all about finding an attribute that returns the highest information gain.*

3rd. Reduction in variance

- ❑ *Is an algorithm used for continuous target variables (regression problems).*
- ❑ *The split with lower variance is selected as the criteria to split the population.*

❖ ***Note: How to decide the best attribute?***
Ans: Calculate the information gain

But, Let's first look at What is Entropy?

What is Entropy?

- ***Entropy*** is just a metric which measures the ***impurity*** in a given attribute or
- *The first step to solving the problem of a decision tree*
- *Defines randomness in the data*
- ***Note:*** The **impurity** is a measure of the homogeneity of the labels at the node.
- **Entropy** can be calculated as:

$$Entropy(s)= -P(yes)\log_2 P(yes)- P(no)\log_2 P(no)$$

- **Where,**
  - ***S= Total number of samples***
  - ***P(yes)= probability of yes***
  - ***P(no)= probability of no***
- ***If number of yes=number of no i.e. p(s)=0.5*** ➔ ***Entropy(s)=1***
- ***If it contains all yes or all no. i.e p(s)= 1 or 0*** ➔ ***Entropy(s)=0***
- ***The value of the Entropy maximum is 0.5.***

Calculating Entropy

$E(S) = -P(Yes) \log_2 P(Yes)$

When $P(Yes) = P(No) = 0.5$ ie $YES + NO = Total\ Sample(S)$

$E(S) = 0.5 \ \log_2 0.5 - 0.5 \ \log_2 0.5$

$E(S) = 0.5(\log_2 0.5 - \log_2 0.5)$

$E(S) = 1$

$E(S) = -P(Yes) \log_2 P(Yes)$

When $P(Yes) = 1$ ie $YES = Total\ Sample(S)$

$E(S) = 1 \ \log_2 1$

$E(S) = 0$

$E(S) = -P(No) \log_2 P(No)$

When $P(No) = 1$ ie $No = Total\ Sample(S)$

$E(S) = 1 \ \log_2 1$

$E(S) = 0$

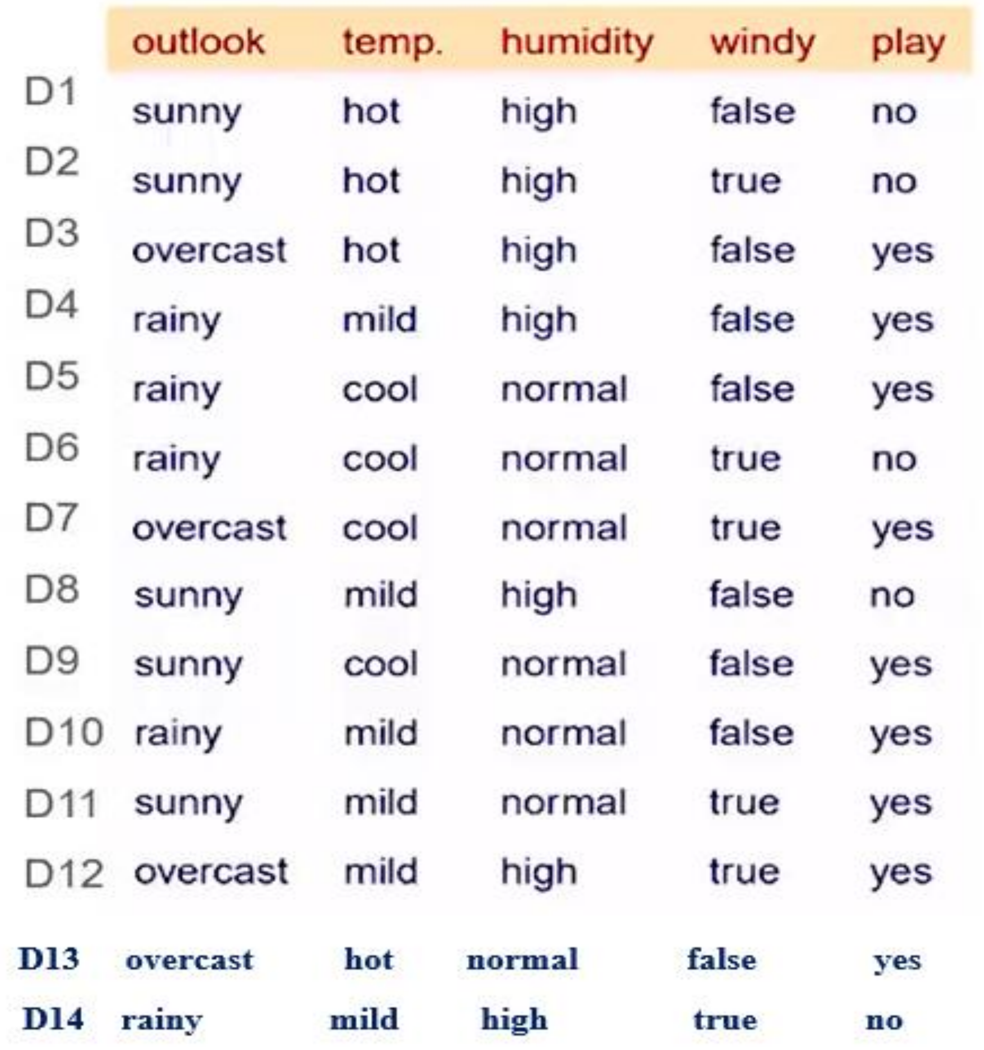
What is Information Gain

- ***Information gain*** Measures the *reduction in* ***entropy***
- Decide which attribute should be selected as the decision node
- It calculates how much information a feature provides us about a class.
- According to the value of *information gain*, we split the node and built the decision tree.
- A ***decision tree algorithm*** always tries to maximize the value of information gain, and a node/attribute having the *highest information gain* is split first.
- It can be calculated using the below formula:

Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)

- If S is our total collection,

Example: Let’s build our sample Decision tree dataset



	outlook	temp.	humidity	windy	play
D1	sunny	hot	high	false	no
D2	sunny	hot	high	true	no
D3	overcast	hot	high	false	yes
D4	rainy	mild	high	false	yes
D5	rainy	cool	normal	false	yes
D6	rainy	cool	normal	true	no
D7	overcast	cool	normal	true	yes
D8	sunny	mild	high	false	no
D9	sunny	cool	normal	false	yes
D10	rainy	mild	normal	false	yes
D11	sunny	mild	normal	true	yes
D12	overcast	mild	high	true	yes
D13	overcast	hot	normal	false	yes
D14	rainy	mild	high	true	no

Step1: Compute the Entropy for the Data Set

Out of 14 instances we have 9 YES and 5 NO

So we have the formula

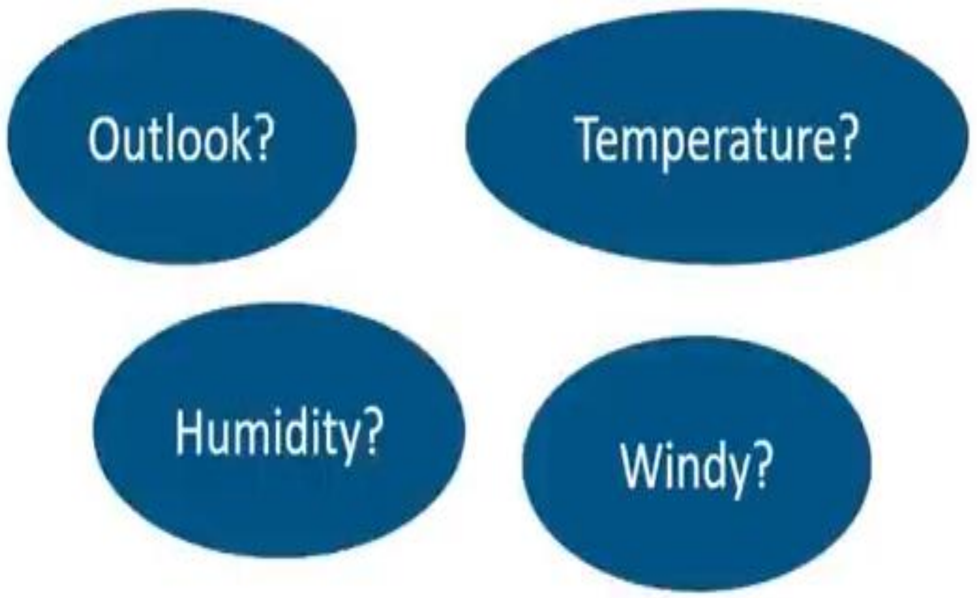
$E(S)=-P(yes)\log_2 P(yes)-P(no)\log_2 P(no)$

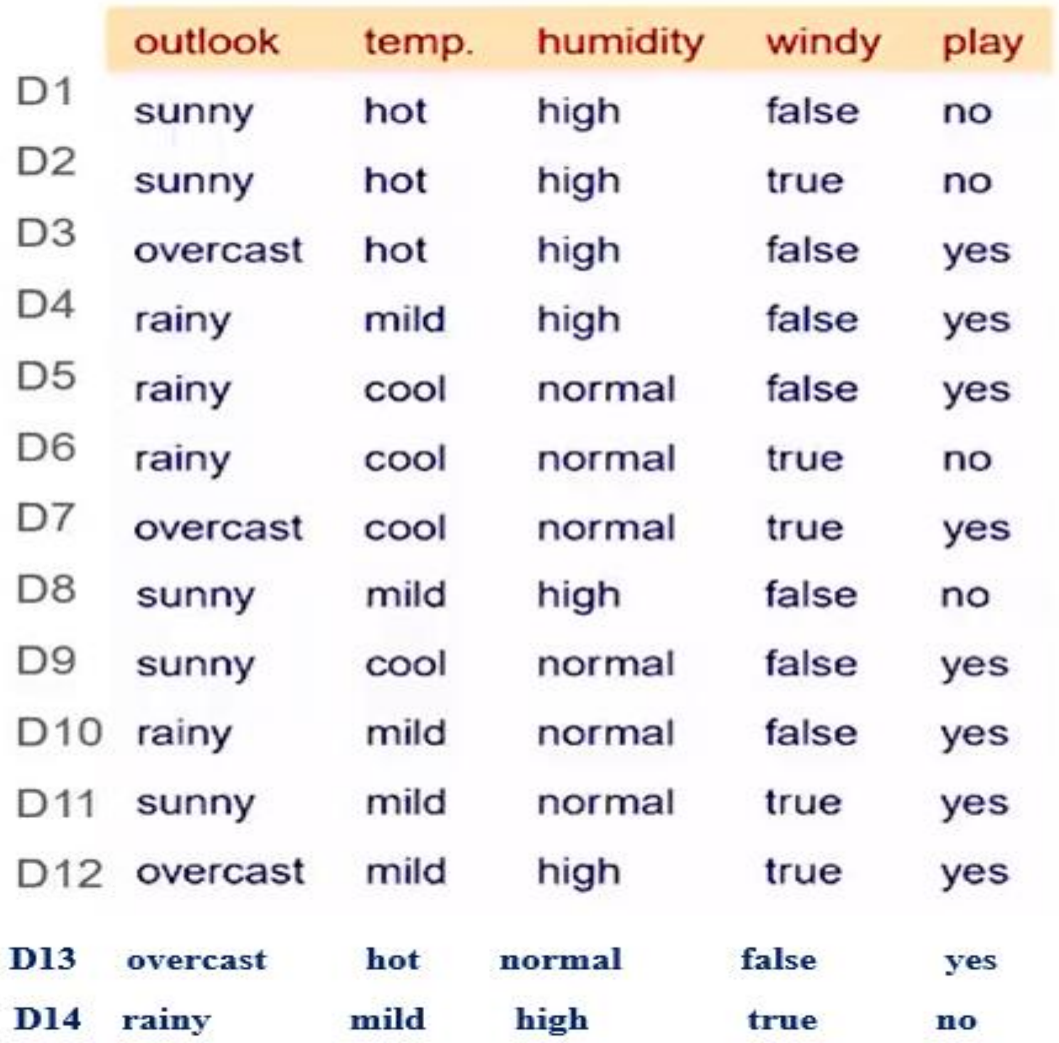
$E(S)=-(9/14)*\log_2 9/14-(5/14)\log_2 5/14$

$E(S)=0.41+0.53=0.94$

❖ ***Note:*** *This is the first step to compute the entropy for the entire data set.*

Question: Which Node to Select As Root Node?

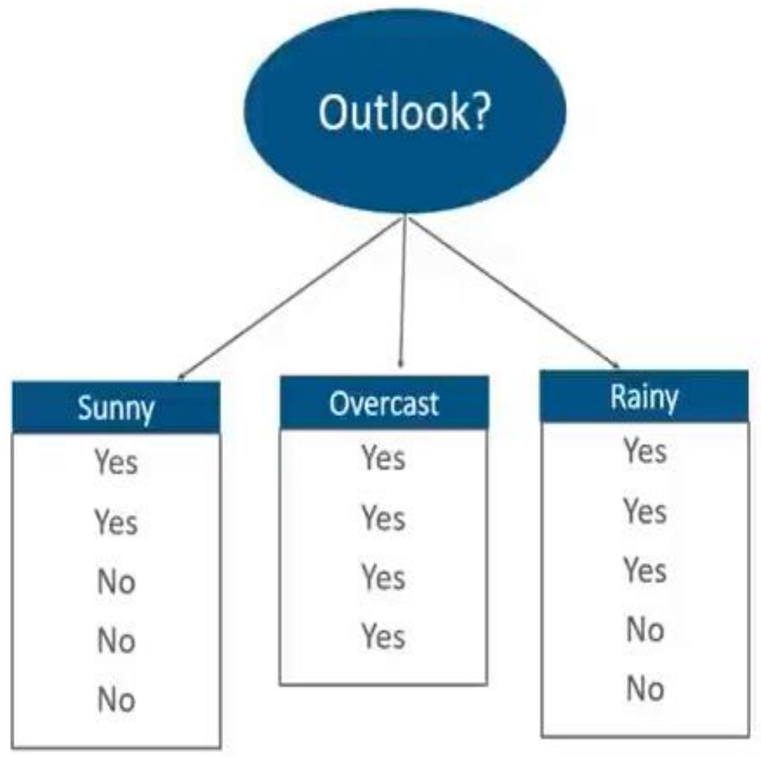




	outlook	temp.	humidity	windy	play
D1	sunny	hot	high	false	no
D2	sunny	hot	high	true	no
D3	overcast	hot	high	false	yes
D4	rainy	mild	high	false	yes
D5	rainy	cool	normal	false	yes
D6	rainy	cool	normal	true	no
D7	overcast	cool	normal	true	yes
D8	sunny	mild	high	false	no
D9	sunny	cool	normal	false	yes
D10	rainy	mild	normal	false	yes
D11	sunny	mild	normal	true	yes
D12	overcast	mild	high	true	yes
D13	overcast	hot	normal	false	yes
D14	rainy	mild	high	true	no

- ❖ **So, we have to calculate Entropy and Information gain for each node**

1st. Select As Root Node= Outlook



- ❖***Outlook*** has three different parameters (*sunny, overcast, and rainy*)
- ❖Calculate the **entropy** for each feature.

- ❑ ***E(Outlook=Sunny)=*** $-(2/5)*\log_2 2/5-3/5 \log_2 3/5=0.971$
- ❑ ***E(Outlook=Overcast)=*** $-1 \log_2 1- 0 \log_2 0=0$
- ❑ ***E(Outlook=Rainy)=*** $-3/5*\log_2 3/5-2/5 \log_2 2/5= 0.97$

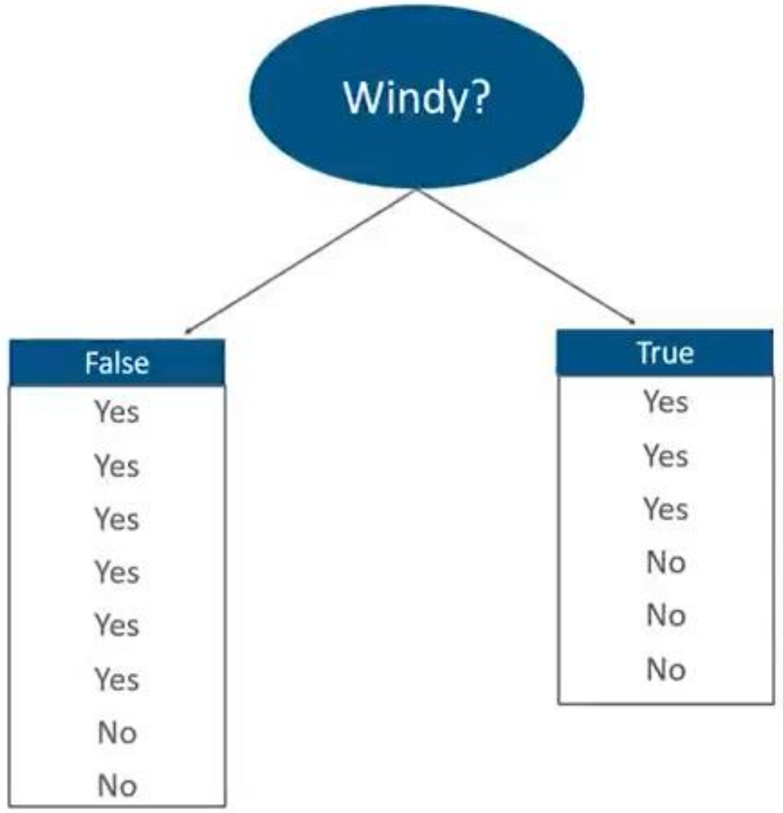
- ❑ **Calculate *Information* from Outlook. i.e. Weight average**

I(Outlook)= $5/14*0.971+4/14*0+5/14*0.971=0.693$

- ❑ **Now Calculate *Information Gained* from Outlook**
- ❑ **Gain(Outlook)=E(S)-I(Outlook)**

0.94-0.693=0.247

2nd. Select As Root Node= Windy



❖ ***Windy*** has two different parameters (*True and False*

❖Calculate the **entropy** for each feature.

❖In the case of true, we have equal no of true and false, so

- **E(Windy=True)=1**
- **E(Windy=False)=0.811**

❖**Information from windy,**

- **I(Windy)=8/14 * 0.811+6/14 *1= 0.892**

❑ **Now Calculate *Information Gained* from Outlook**

Gain(Windy)=E(S)-I(Windy)

0.94-0.892=0.048

❖ ***Note: Similarly we calculate for humidity and temperature***

Which Node to Select As Root Node?

Outlook:
Info 0.693
Gain: 0.940-0.693= 0.247

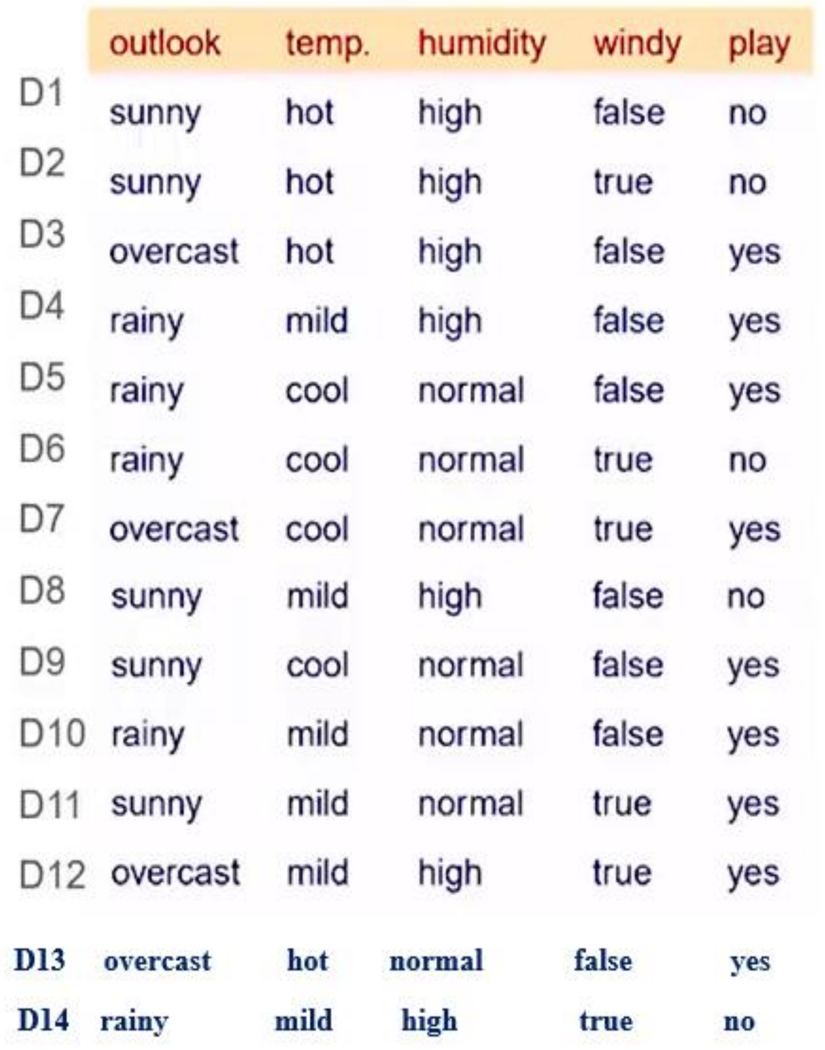
Temperature:
Info 0.911
Gain: 0.940-0.693=0.029

Humidity:
Info 0.788
Gain: 0.940-0.788= 0.512

Windy:
Info 0.892
Gain: 0.940-0.982= 0.048

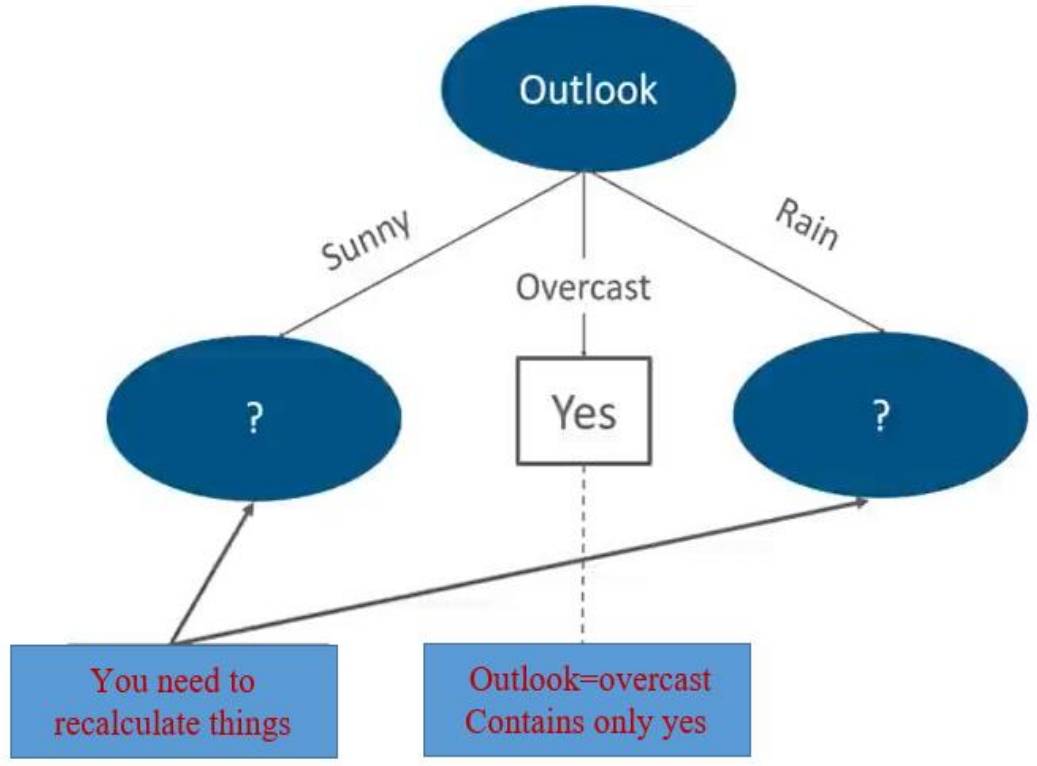
- **Now, select the attribute with the max gain**

- **Since Max gain = 0.247,**
- Outlook is our Root Node

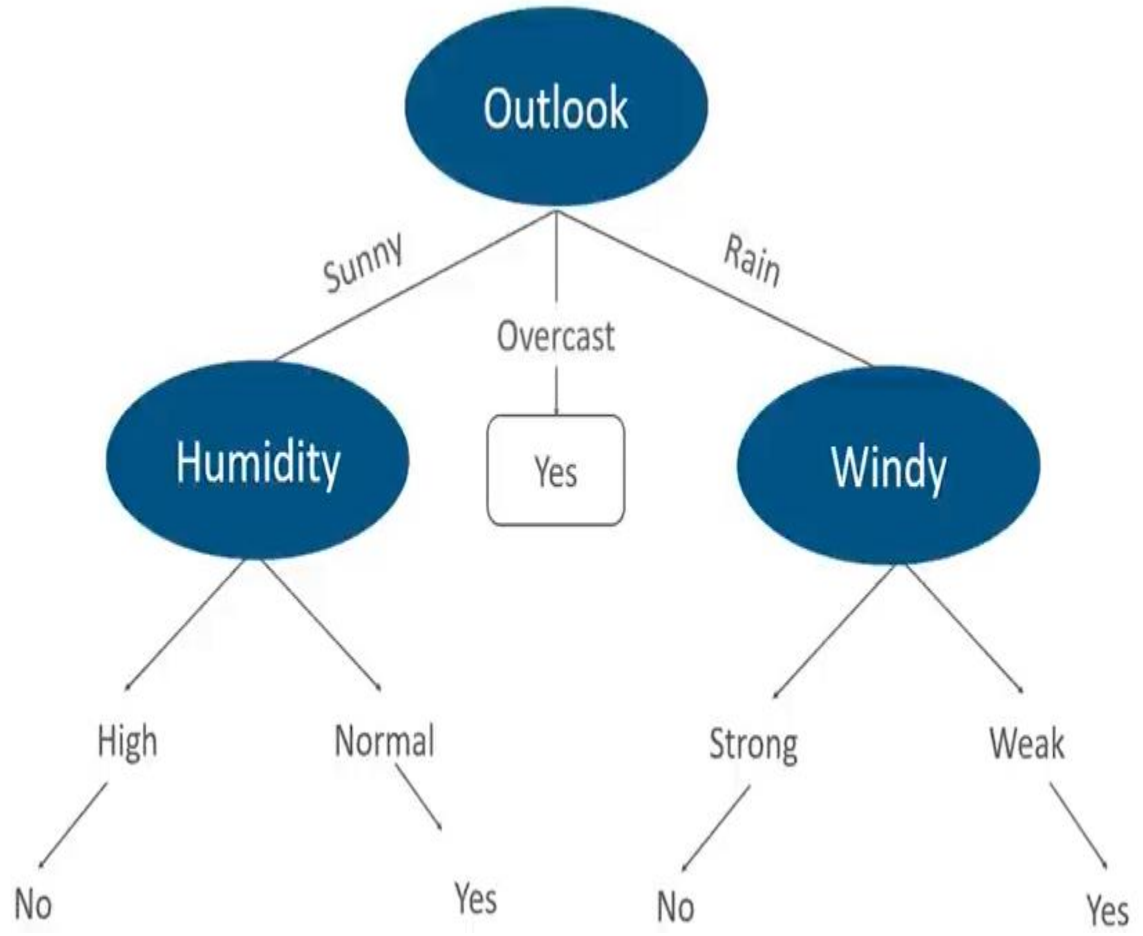


	outlook	temp.	humidity	windy	play
D1	sunny	hot	high	false	no
D2	sunny	hot	high	true	no
D3	overcast	hot	high	false	yes
D4	rainy	mild	high	false	yes
D5	rainy	cool	normal	false	yes
D6	rainy	cool	normal	true	no
D7	overcast	cool	normal	true	yes
D8	sunny	mild	high	false	no
D9	sunny	cool	normal	false	yes
D10	rainy	mild	normal	false	yes
D11	sunny	mild	normal	true	yes
D12	overcast	mild	high	true	yes
D13	overcast	hot	normal	false	yes
D14	rainy	mild	high	true	no

Which Node to Select Further?


Outlook
Sunny
Overcast
Rain
?
Yes
?
You need to recalculate things
Outlook=overcast Contains only yes

This is What your complete tree will look like.



Pruning: Getting an Optimal Decision tree

❑What should I Do to play-Pruning?

❑**What is Pruning?**

- *“A **decision tree** is a graphical representation of all the possible solutions to a decision based on certain conditions”*

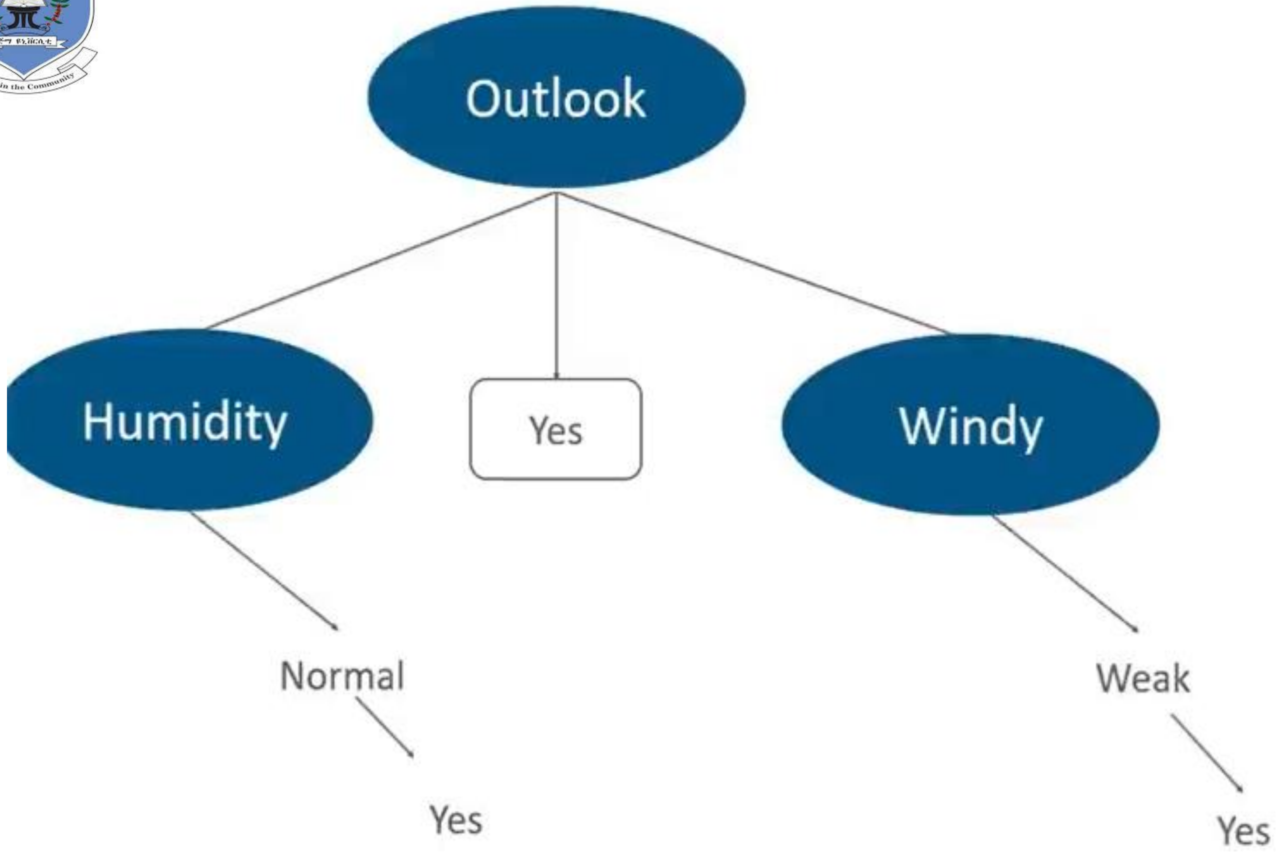
❑***Pruning*** *is a process of deleting unnecessary nodes from a tree in order to get the optimal decision tree.*

❑*A **too-large tree** increases the risk of overfitting, and a small tree may not capture all the important features of the dataset.*

❑Therefore, a technique that decreases the size of the learning tree without reducing accuracy is known as ***Pruning.***

❑***Pruning:*** *Reducing the complexity*

Pruning: Getting an Optimal Decision Tree



❑The graph only shows the result for the **yes.**

❖ **Common Questions:** *Are tree-based models better than linear models?*
❖ *Ans: It depends type of algorithm we solve it.*

Advantages and Disadvantages of the Decision Tree

❖**Advantages of the Decision Tree**

- ❑*It is simple to understand as it follows the same process that a human follows while making any decision in real life.*
- ❑*It can be very useful for solving decision-related problems.*
- ❑*It helps to think about all the possible outcomes for a problem.*

❖**Disadvantages of the Decision Tree**

- ❑*The decision tree contains lots of layers, which makes it complex.*
- ❑*It may have an overfitting issue, which can be resolved using the* ***Random Forest algorithm.***
- ❑*For more class labels, the computational complexity of the decision tree may increase.*

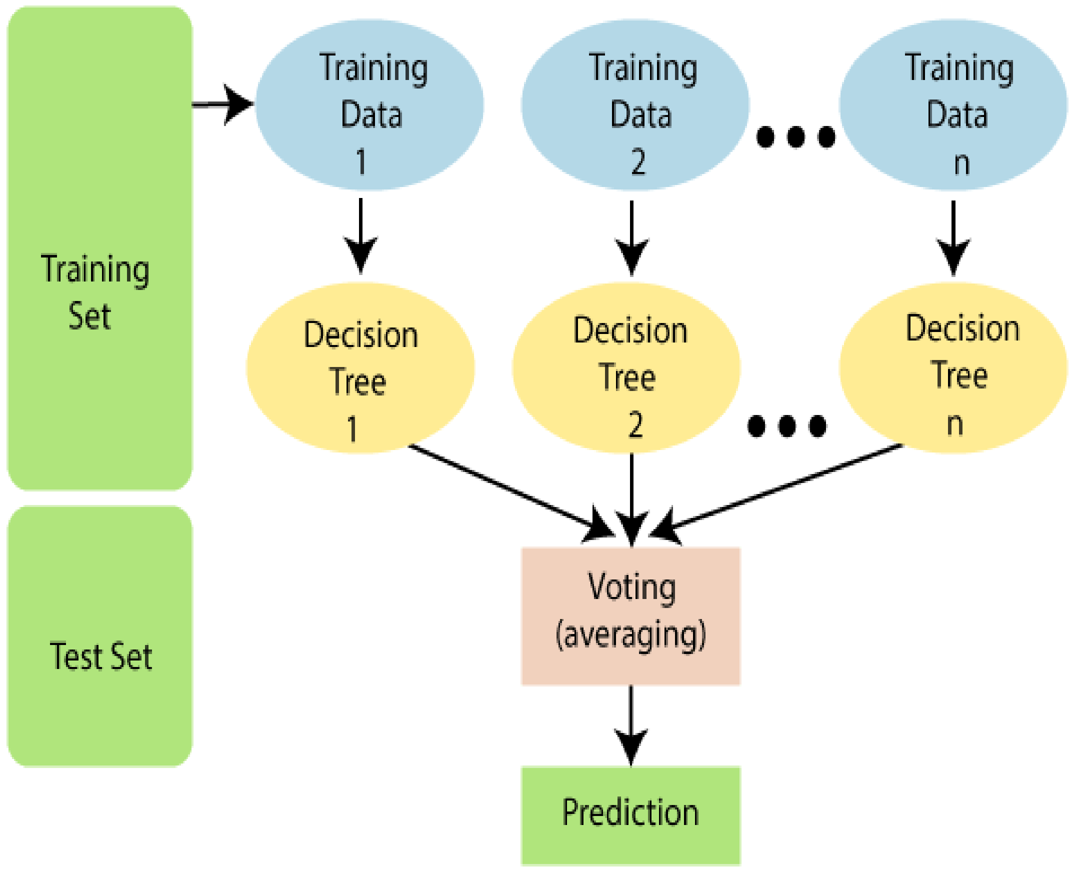
Random Forest Classification

❖ What is Random Forest?

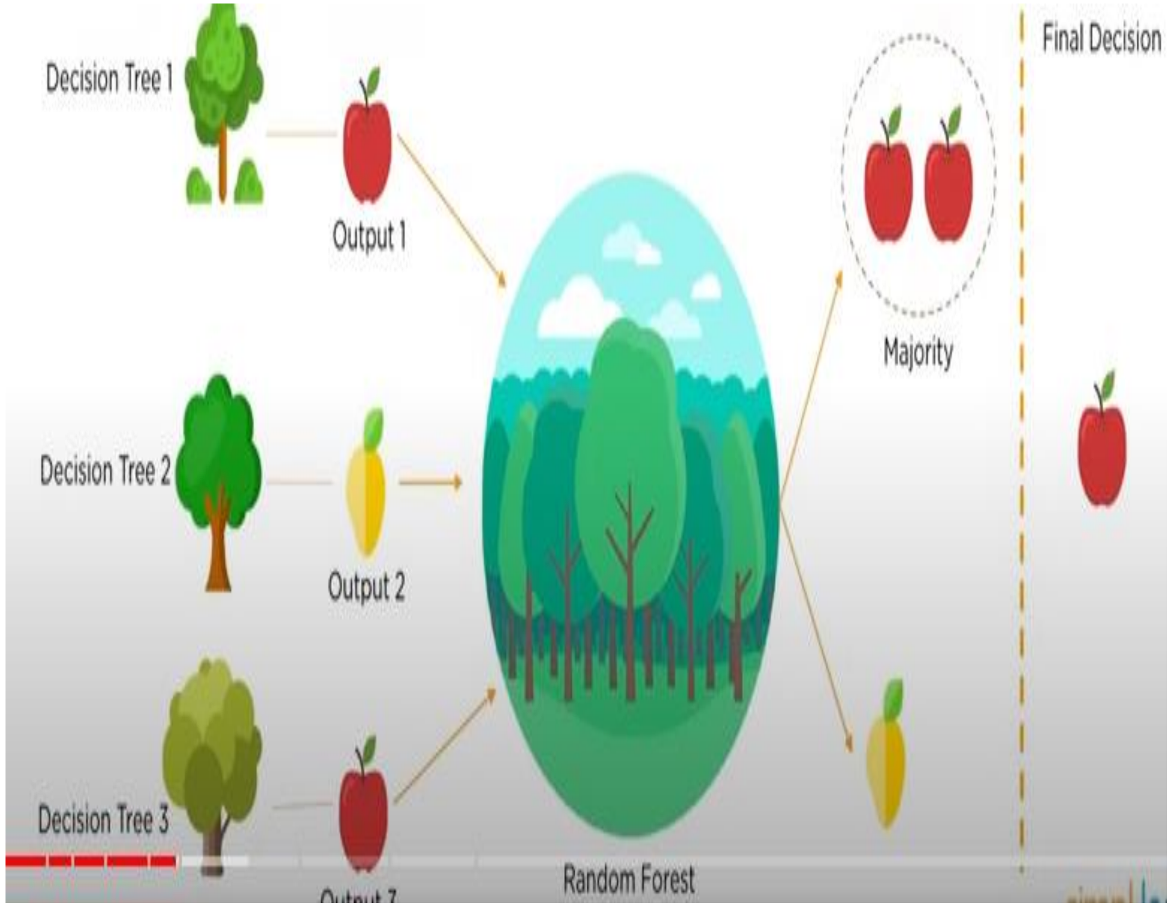
- ❑ ***Random Forest*** is a popular ML algorithm that belongs to the supervised learning technique.
- ❑*Random forest* or *Random Decision Forest* is a method that operates by constructing multiple Decision trees.
- ❑The decision of the majority of the trees are chosen by the random forest as the ***final decision.***
- ❑It is based on the concept of ***ensemble learning,*** which is a process of *combining multiple classifiers to solve a complex problem and improve the performance of the model*.
- ❑ Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, predicts the final output.
- ❑The greater number of trees in the forest leads to higher accuracy and prevents the problem of ***overfitting.***

Example: Random Forest

❑ The below diagram explains the working of the Random Forest algorithm:



Example #1



Example #2

Why use Random Forest?

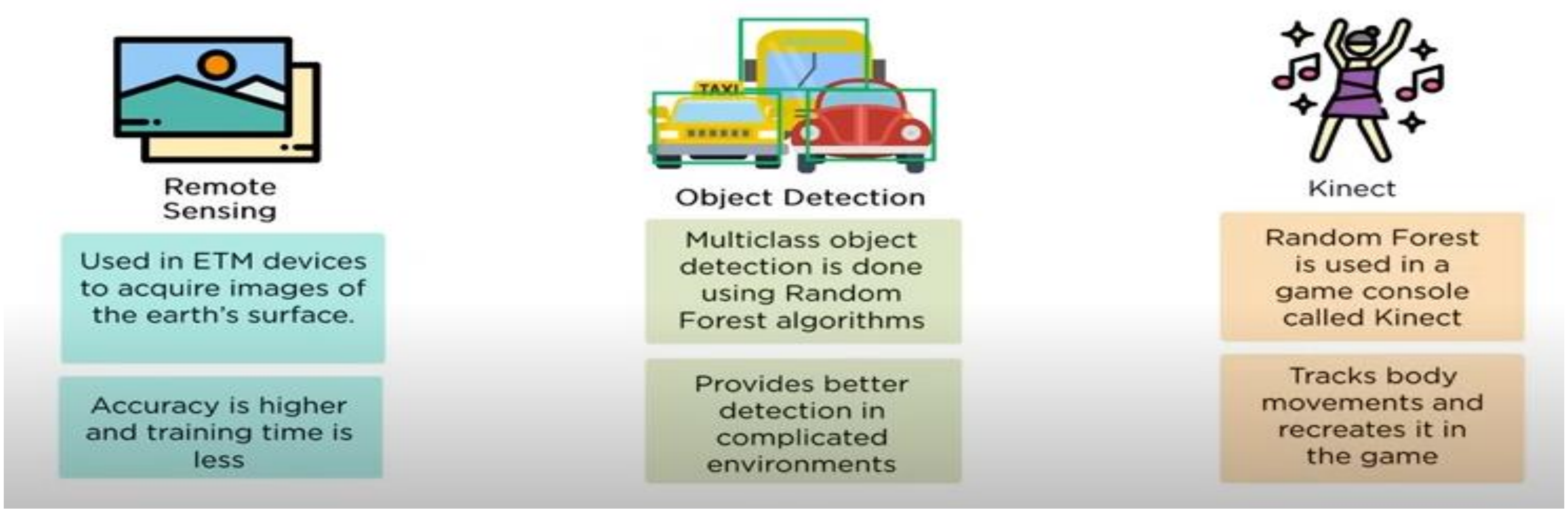
❖Below are some points that explain why we should use the *Random Forest algorithm:*

- ❑ *The use of multiple trees reduces the risk of overfitting*
- ❑ *It takes less training time as compared to other algorithms.*

- ❑ *Runs efficiently on a large database*
- ❑ *For large data, it produces highly accurate predictions.*

- ❑ Random Forest can maintain accuracy when a large proportion of data is missing
- ❑ For large data, it produces highly accurate predictions.

Application of Random Forest



Remote Sensing

Used in ETM devices to acquire images of the earth's surface.

Accuracy is higher and training time is less

Object Detection

Multiclass object detection is done using Random Forest algorithms

Provides better detection in complicated environments

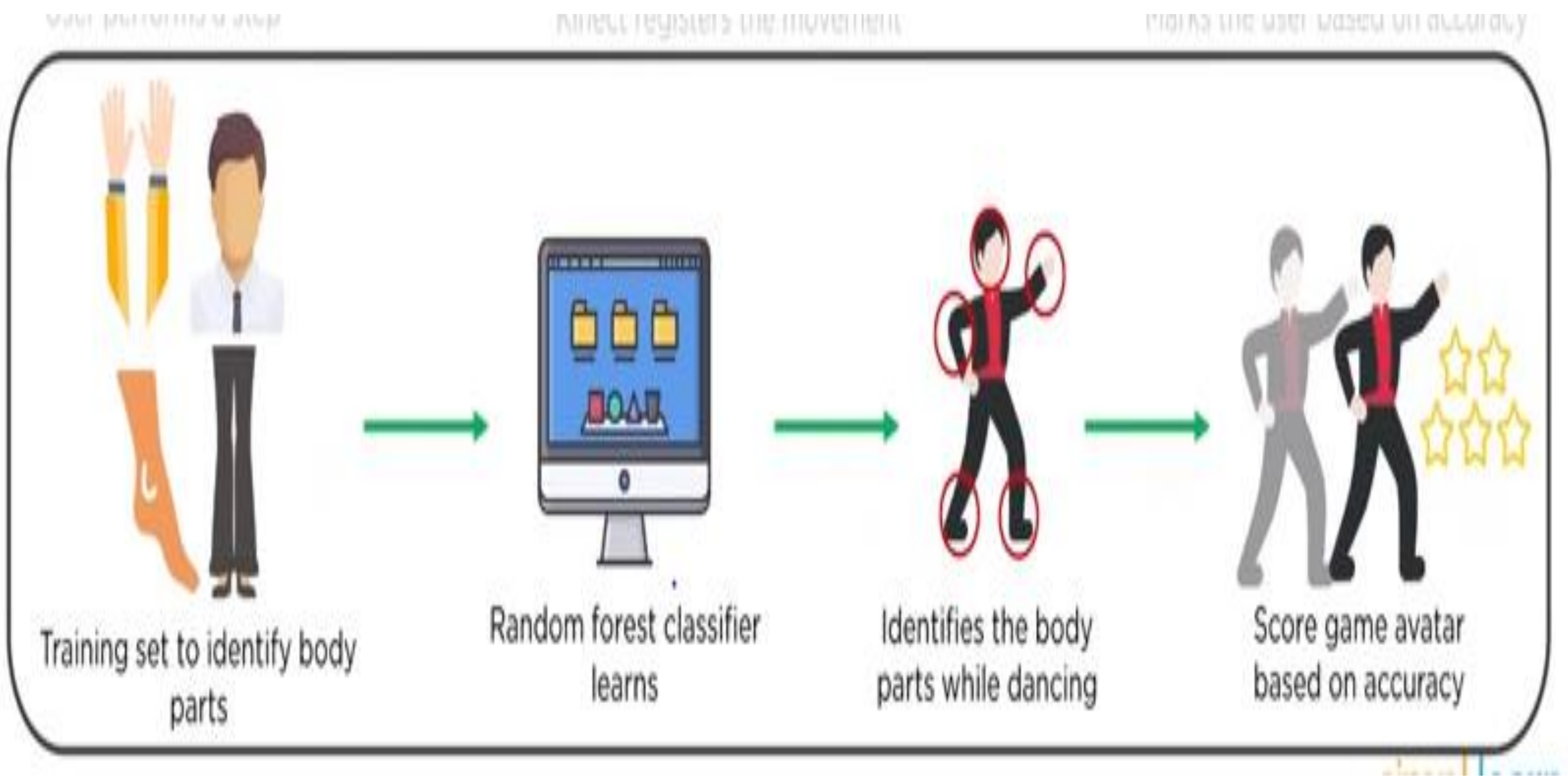
Kinect

Random Forest is used in a game console called Kinect

Tracks body movements and recreates it in the game

- **Banking:** *The banking sector mostly uses this algorithm for the identification of loan risk.*
- **Medicine:** *With the help of this algorithm, disease trends and risks of the disease can be identified.*
- **Land Use**: *We can identify the areas of similar land use by this algorithm.*
- **Marketing:** *Marketing trends can be identified using this algorithm.*

Application of Random Forest: Example


User performs a step
Kinect registers the movement
Marks the user based on accuracy
Training set to identify body parts
Random forest classifier learns
Identifies the body parts while dancing
Score game avatar based on accuracy

How Does a Random Forest Algorithm Work?

- ❖ ***Random Forest*** works in two phases.
  - ❖ *1st is to create the random forest by combining the N decision tree*
  - ❖ *2nd is to make predictions for each tree created in the first phase.*

❖ **The Working process can be explained in the below steps and diagram:**

Step-1: *Select random **K data points** from the **training set.***

Step 2: *Build the decision trees associated with the selected data points (Subsets).*

Step 3: *Choose the number N for the decision trees that you want to build.*

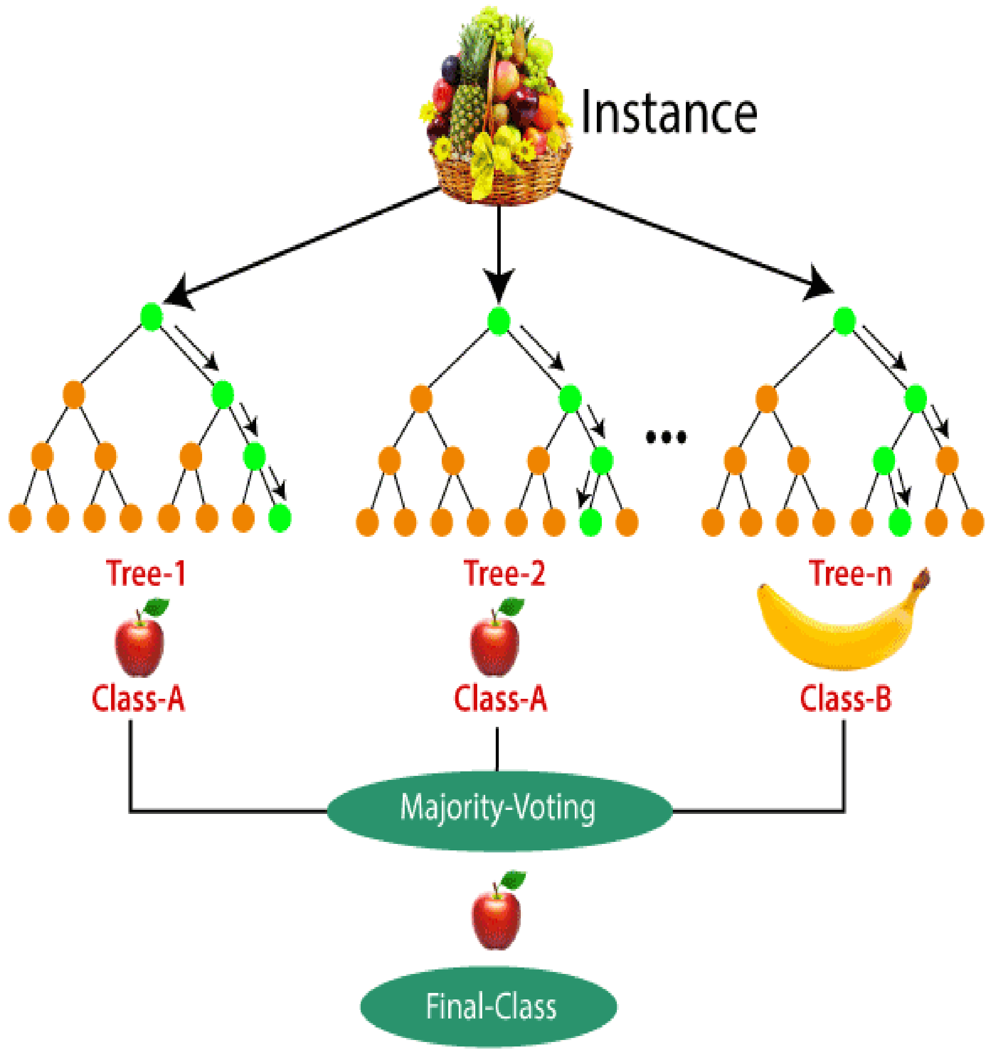
Step 4: *Repeat Steps 1 & 2.*

Step-5: *For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.*

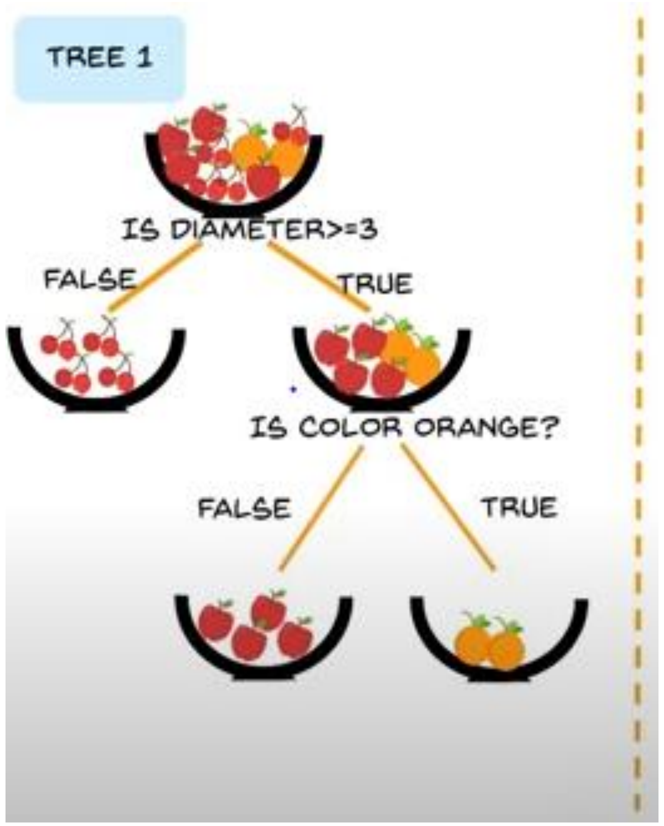
Example: How Does a Random Forest Algorithm Work?

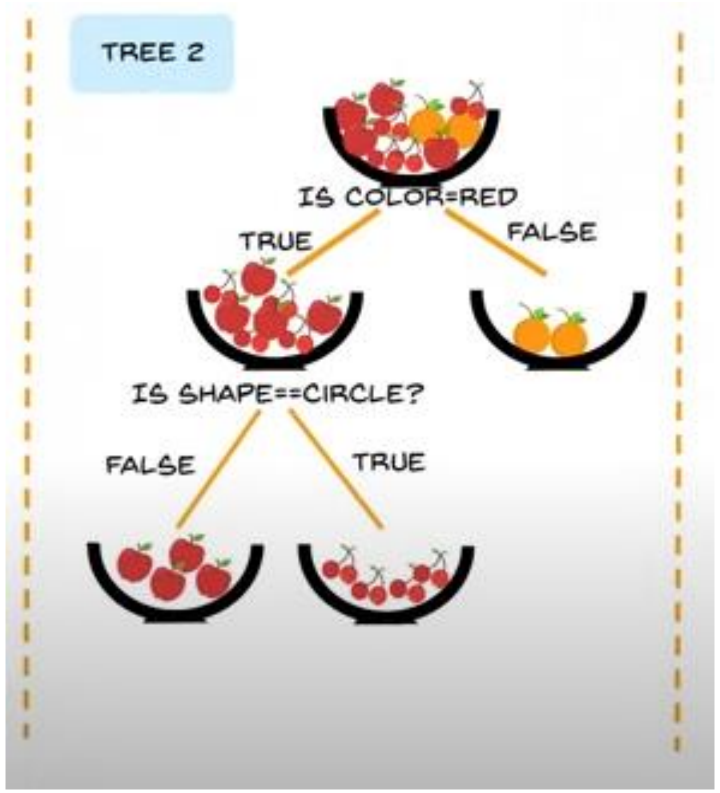
- ❑*Suppose there is a dataset that contains multiple fruit images. So, this dataset is given to the Random forest classifier.*
- ❑*The dataset is divided into subsets and given to each decision tree.*
- ❑*During the training phase, each decision tree produces a prediction result, and when a new data point occurs, then based on the majority of results, the Random Forest classifier predicts the final decision.*

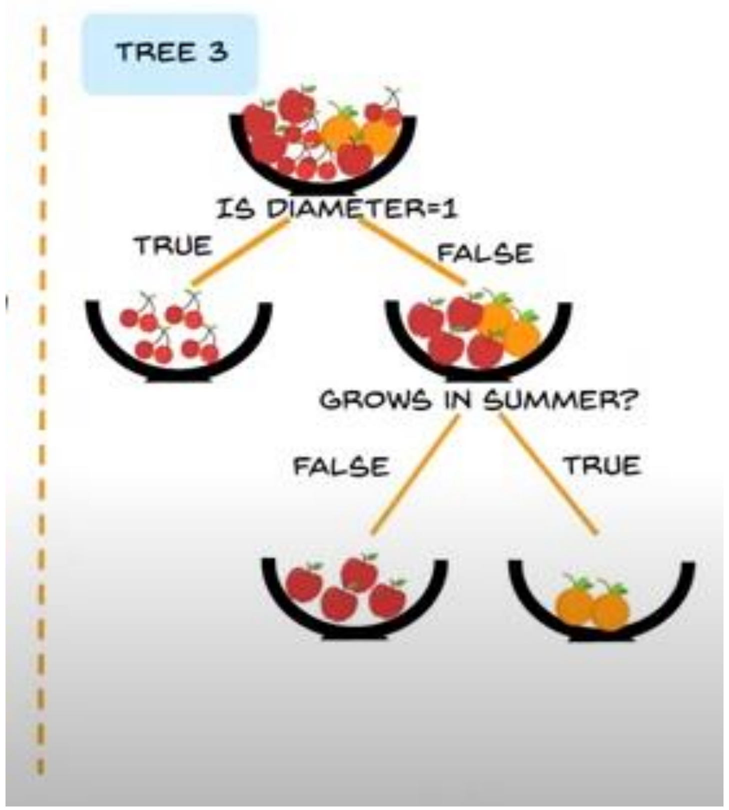
❑*Consider the image:*



Example: How Does a Random Forest Algorithm Work? …







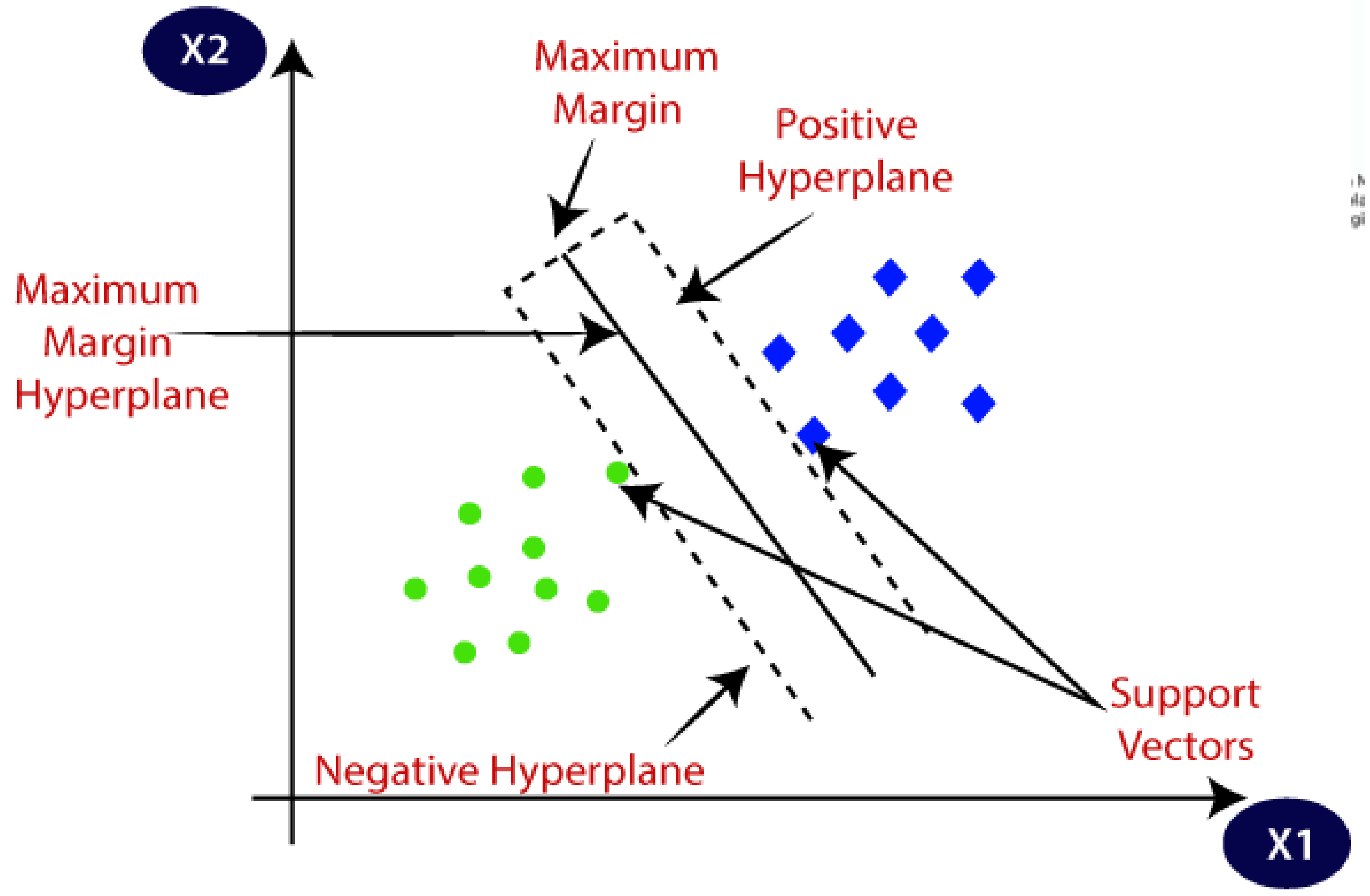
- ***This is three different tree categories the fruit***

Support Vector Machines (SVM)

- ❑ **SVM** is a supervised classification method that separates data using ***hyperplanes***
- ❑**SVM** can be used for both *classification* and *regression* problems. *However, it is mostly used in classification problems, such as* ***text classification****.*
- ❑**The goal of the *SVM algorithm* is:**
  - ✓ to create the ***best line*** or ***decision boundary*** *that can* ***segregate n-dimensional*** *space into classes so that we can easily put the new data point in the correct category in the future.*
- ❑This best decision boundary is called a ***hyperplane.***
- ❑***SVM*** chooses the extreme ***points/vectors*** that help in creating the hyperplane.
- ❑ These extreme cases are called ***support vectors***, and hence algorithm is termed an ***SVM.***

Support Vector Machines (SVM)

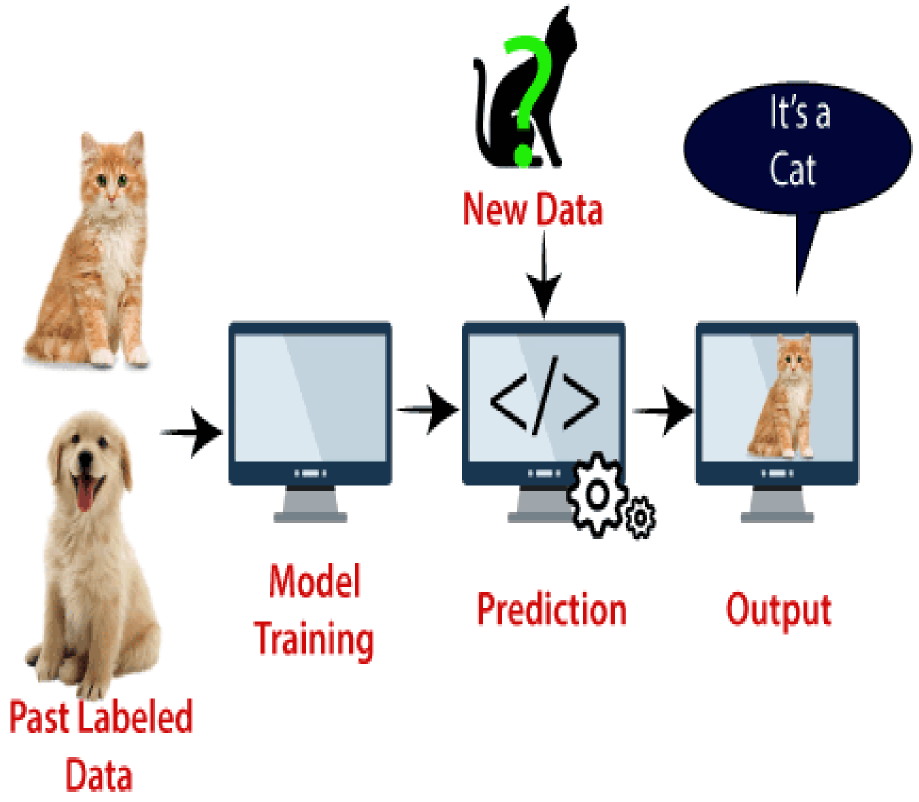
❑*Consider the diagram in which there are two different categories that are classified using a decision boundary or hyperplane:*



Why SVM?

❑ **Why not build a model which can predict unknown data??.** ***Consider the Example***

- ❑*Suppose we see a strange cat that also has some features of dogs, if we want a model that can accurately identify whether it is a cat or dog, such a model can be created by using the SVM algorithm.*
- ❑ *We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature.*
- ❑*So as the support vector creates a decision boundary between these two data (cat and dog) and chooses extreme cases (support vectors), it will see the extreme case of* ***cat and dog.***
- ❑*On the basis of the support vectors, it will classify it as a cat.*



❑*Consider the above diagram:*

Types of SVM

❖ **SVM can be of two types:**

❑ **Simple or Linear SVM:**

✓ *is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed linearly separable data, and a classifier is called a* ***simple or Linear SVM classifier*****.**

❑ **Kernel or Non-linear SVM:**

✓ *is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data, and the classifier used is called a* ***Kernel or Non-linear SVM classifier*****.**

Hyperplane and Support Vectors in the SVM algorithm:

- **Hyperplane:** There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the *best decision boundary* that helps to classify the data points.
- This best boundary is known as the ***hyperplane of SVM.***
- The dimensions of the hyperplane depend on the features present in the dataset, which means if there are ***2 features*** (as shown in Figure a), then the hyperplane will be a straight line.
- And if there are 3 features, then the hyperplane will be a ***2-dimension plane.***
- *We always create a **hyperplane** that has a **maximum margin**, which means the maximum distance between the data points.*

❖ **Support Vectors:**

- The ***data points or vectors*** that are the closest to the hyperplane and which affect the position of the hyperplane are termed ***Support Vector.***
- Since these vectors support the hyperplane, hence called a ***Support vector***.

How does SVM work?

❖**Linear SVM: Example.**

- ❑*Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2.*
- ❑*We want a classifier that can classify the pair(**x1, x2**) of coordinates in either green or blue.*

❑Consider the below image:

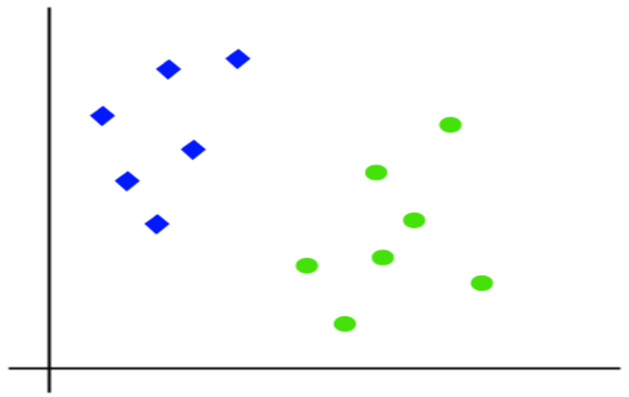


Figure a

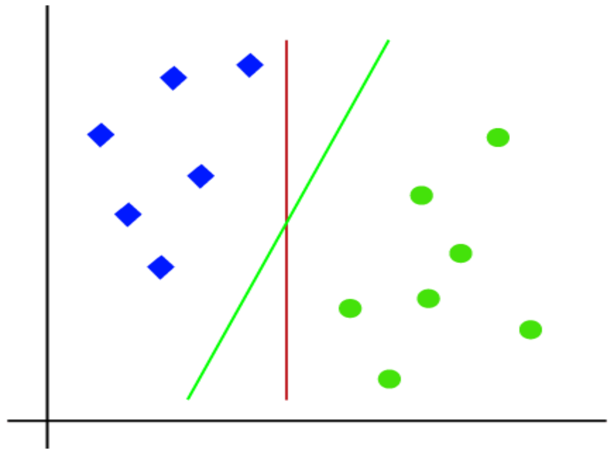
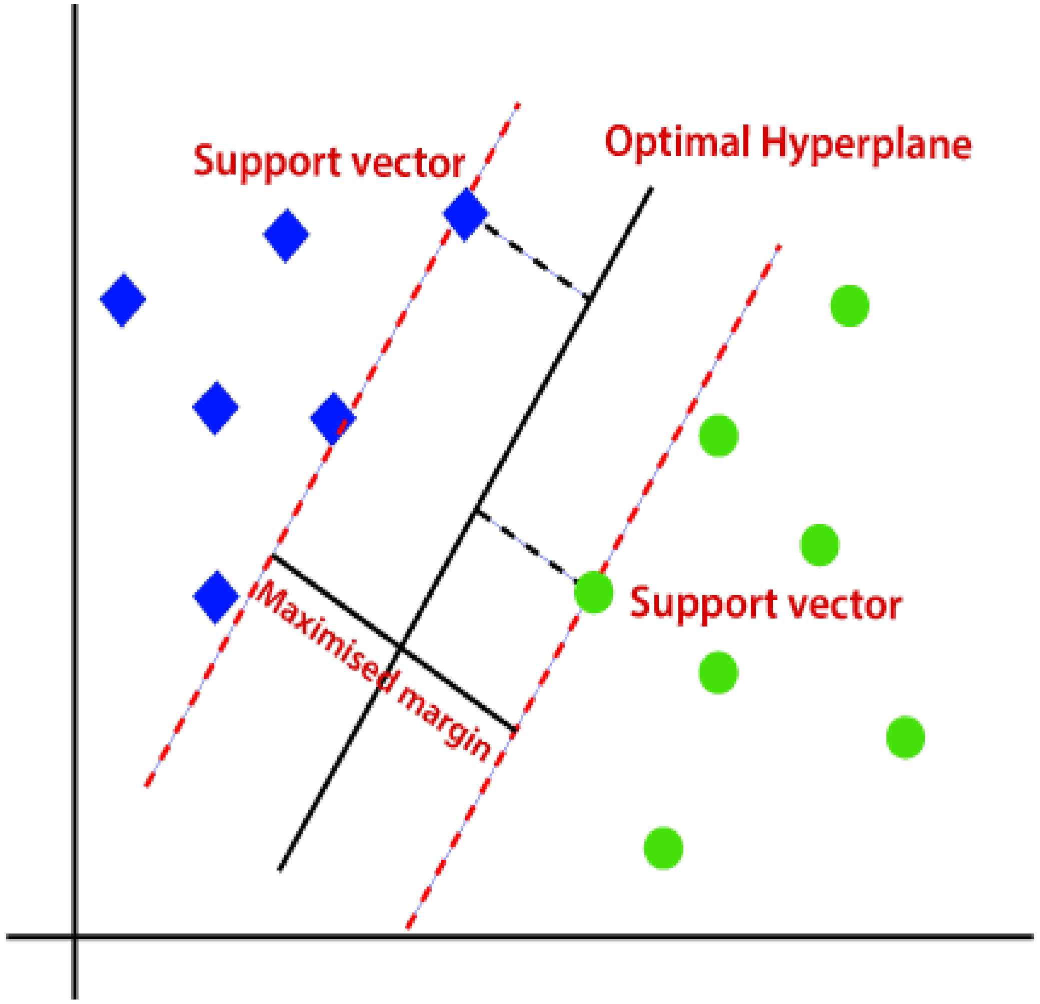


Figure b

- ❑ *So as it is 2-d space by just using a straight line, we can easily separate these two classes.*
- ❑ *But there can be multiple lines that can separate these classes. Consider the image:*

How does SVM work?....

- ❑Hence, the **SVM algorithm** helps to find the *best line or decision boundary*; this best boundary or region is called a **hyperplane**.
- ❑SVM algorithm finds the closest point of the lines from both classes. These points are called ***support vectors.***
- ❑The distance between the vectors and the hyperplane is called as ***margin***.
- ❑The goal of ***SVM*** is to maximize this margin.
- ❑The ***hyperplane*** with the maximum margin is called the **optimal hyperplane**.

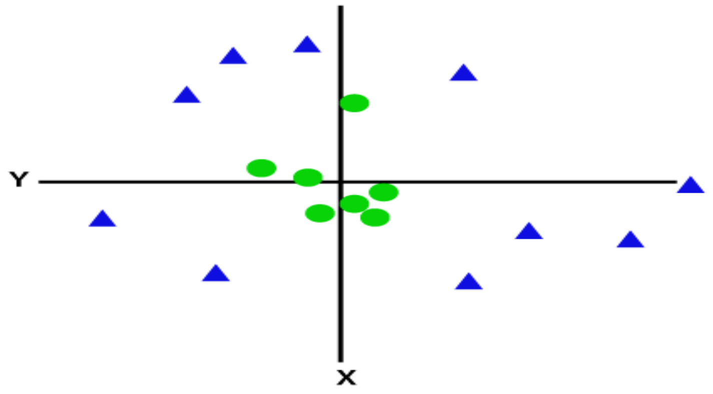


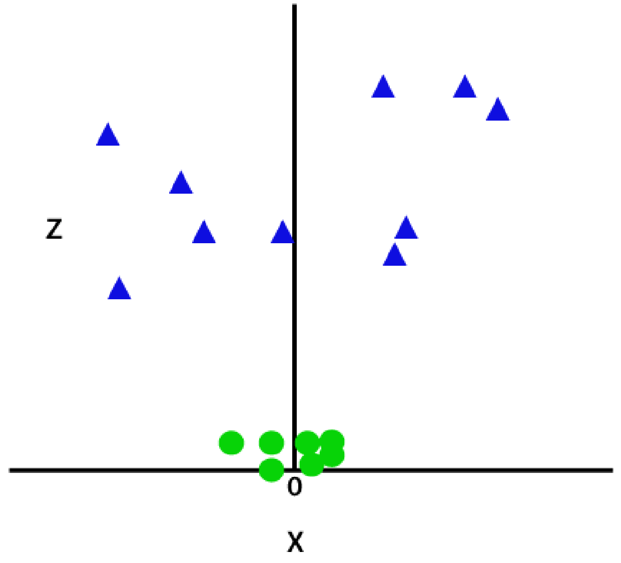
How does SVM work?

❑ **Non-Linear SVM:**

✓If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line.

❑Consider the figure:





❑*So to separate these data points, we need to add one more dimension.*

❑*for non-linear data, we will add a third dimension z.*

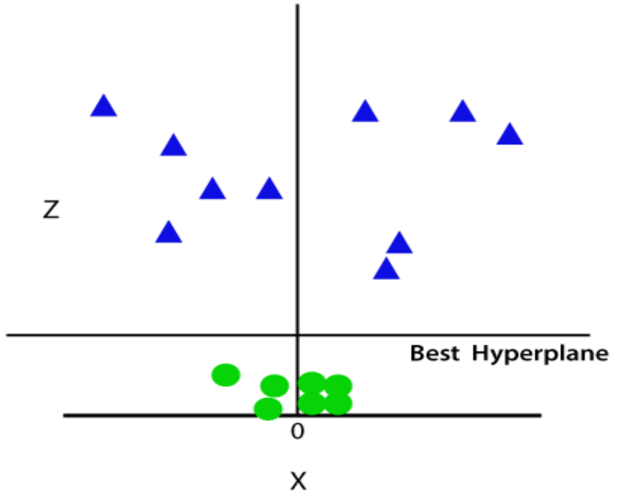
❑ *It can be calculated as:* $z=x^2+y^2$

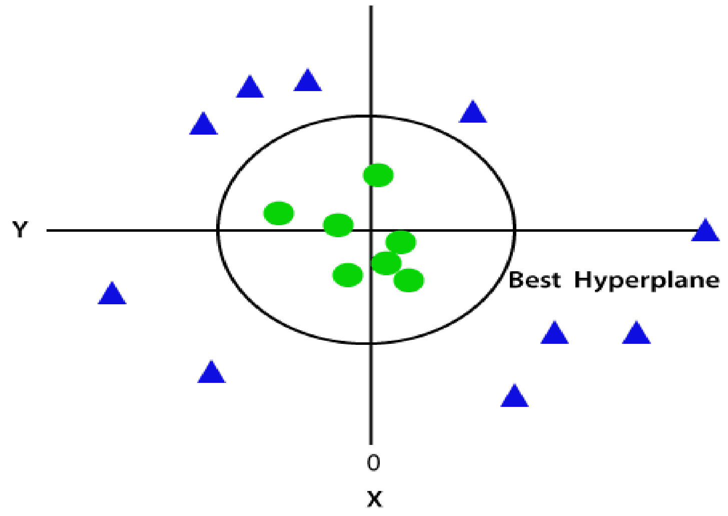
❑ *By adding the third dimension, the sample space will become as shown in the figure above*

How does SVM work? ...

- So now, SVM will divide the datasets into classes in the following way.

- Consider the below figure:

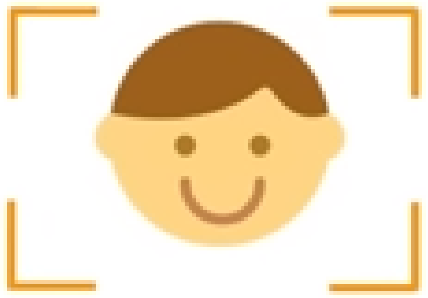




- Since we are in 3-d Space, it is looking like a plane parallel to the x-axis.

- If we convert it in 2d space with z=1, then it will become as figure above:
- *Hence we get a circumference of radius 1 in the case of non-linear data.*

Applications of SVM



- ✓ Face Detection
- ✓ Text and hypertext categorization
- ✓ Image classification
- ✓ Bioinformatics

A case study in classification.

Understand the metrics used to evaluate classifiers.

- ❖Evaluation metrics are tied to machine learning tasks.
- ❖Classification metrics are evaluation measures used to assess the performance of a classification model.

❑The commonly used metrics for evaluating classifier performance are:

❑***Accuracy:*** Accuracy simply measures how often the classifier correctly predicts.

❑***Precision:*** It explains how many of the correctly predicted cases actually turned out to be positive.

❑Recall: It explains how many of the actual positive cases we were able to predict correctly with our model.

❑***F1 Score:*** It gives a combined idea about Precision and Recall metrics.

Note: *this part the details will be discussing at chapter 5*