

- **UNIT # 2**

Unit	Topic	Proposed Lecture
I	INTRODUCTION – Learning, Types of Learning, Well defined learning problems, Designing a Learning System, History of ML, Introduction of Machine Learning Approaches – (Artificial Neural Network, Clustering, Reinforcement Learning, Decision Tree Learning, Bayesian networks, Support Vector Machine, Genetic Algorithm), Issues in Machine Learning and Data Science Vs Machine Learning;	08
II	REGRESSION: Linear Regression and Logistic Regression BAYESIAN LEARNING - Bayes theorem, Concept learning, Bayes Optimal Classifier, Naïve Bayes classifier, Bayesian belief networks, EM algorithm. SUPPORT VECTOR MACHINE: Introduction, Types of support vector kernel – (Linear kernel, polynomial kernel, and Gaussian kernel), Hyperplane – (Decision surface), Properties of SVM, and Issues in SVM.	08
III	DECISION TREE LEARNING - Decision tree learning algorithm, Inductive bias, Inductive inference with decision trees, Entropy and information theory, Information gain, ID-3 Algorithm, Issues in Decision tree learning. INSTANCE-BASED LEARNING – k-Nearest Neighbour Learning, Locally Weighted Regression, Radial basis function networks, Case-based learning.	08
IV	ARTIFICIAL NEURAL NETWORKS – Perceptron's, Multilayer perceptron, Gradient descent and the Delta rule, Multilayer networks, Derivation of Backpropagation Algorithm, Generalization, Unsupervised Learning – SOM Algorithm and its variant; DEEP LEARNING - Introduction, concept of convolutional neural network, Types of layers – (Convolutional Layers, Activation function, pooling, fully connected), Concept of Convolution (1D and 2D) layers, Training of network, Case study of CNN for eg on Diabetic Retinopathy, Building a smart speaker, Self-driving car etc.	08
V	REINFORCEMENT LEARNING –Introduction to Reinforcement Learning, Learning Task, Example of Reinforcement Learning in Practice, Learning Models for Reinforcement – (Markov Decision process, Q Learning - Q Learning function, Q Learning Algorithm), Application of Reinforcement Learning, Introduction to Deep Q Learning. GENETIC ALGORITHMS: Introduction, Components, GA cycle of reproduction, Crossover, Mutation, Genetic Programming, Models of Evolution and Learning, Applications.	08

Text books:

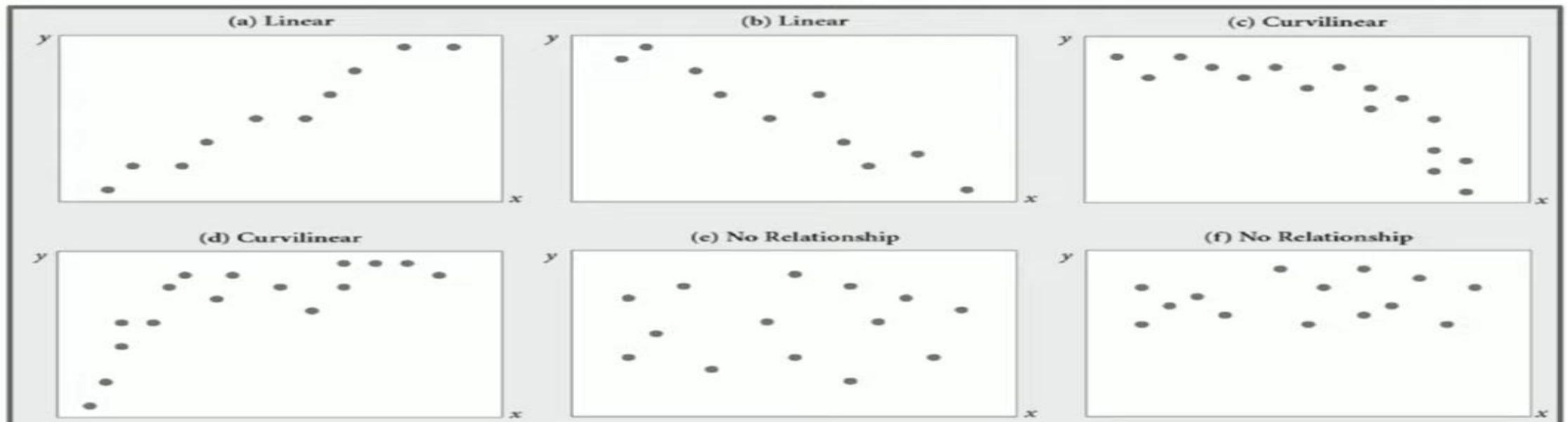
1. Tom M. Mitchell, —Machine Learning, McGraw-Hill Education (India) Private Limited, 2013.
2. Ethem Alpaydin, —Introduction to Machine Learning (Adaptive Computation and Machine Learning), The MIT Press 2004.
3. Stephen Marsland, —Machine Learning: An Algorithmic Perspective, CRC Press, 2009.
4. Bishop, C., Pattern Recognition and Machine Learning. Berlin: Springer-Verlag.

Linear Regression

- It's a supervised learning technique.
- Regression is a technique to model the predictive relationship between one or more independent variables and one dependent variable.
- The objective of regression is to find the best fitting curve for dependent variable which is the function of one or more independent variables. If independent variable is one then its called 1 dimensional regression analysis and if independent variables are more than one then its called Multidimensional regression analysis.
- The curve could be a straight line or Non linear curve.
- The quality of fitting of the curve is measured by calculating coefficient of correlation(r). Depending on the value of r the quality of regression is declared.
- Coefficient of correlation is the square root of the amount of variance given by the curve.

Possible Cases

- For the given cases two variables x & y are shown. In the first case as x increases y also increases and a straight line can be drawn which fits the curve.
- In the second case, as x increases y decreases. Here also a straight line can be drawn which will perfectly fit with the data set.
- In third case a straight line cannot be drawn but a curve is drawn to fit the given data set. Its called curvilinear regression analysis. Fourth case is also an example of curvilinear regression analysis.
- In the last two cases neither a straight line nor a curve can be drawn. So no relationship can be established.



Correlation

- For two numeric variables x and y , correlation can be calculated as

$$r = \frac{[(x - \bar{x})][y - \bar{y}]}{\sqrt{[(x - \bar{x})^2][(y - \bar{y})^2]}}$$

- \bar{x} is the mean of x and \bar{y} is the mean of y .
- Correlation defines the kind and strength of relationship developed between two variables.
- It's a quantitative measure which that is measured in the range from 0 to 1.
- A correlation of 1 indicates perfect relationship and correlation of 0 indicates no relationship.

Coefficient of correlation

- The relationship can be positive or it can be an inverse relationship i.e. variables can move in the same direction or in the opposite direction.
- So better measure is the correlation coefficient instead of correlation. Correlation coefficient is the square root of the correlation. It varies in the range of -1 to +1.
- 1 shows perfect relationship in the same direction and -1 shows the perfect relationship in the opposite direction. 0 shows no relationship.

Steps of Regression Analysis

- List all the variables available for making the model.
- Establish a dependent variable of interest.
- Examine the relationship between variables of interest.
- Find a way to predict dependent variable using other variables.

Linear Regression Model

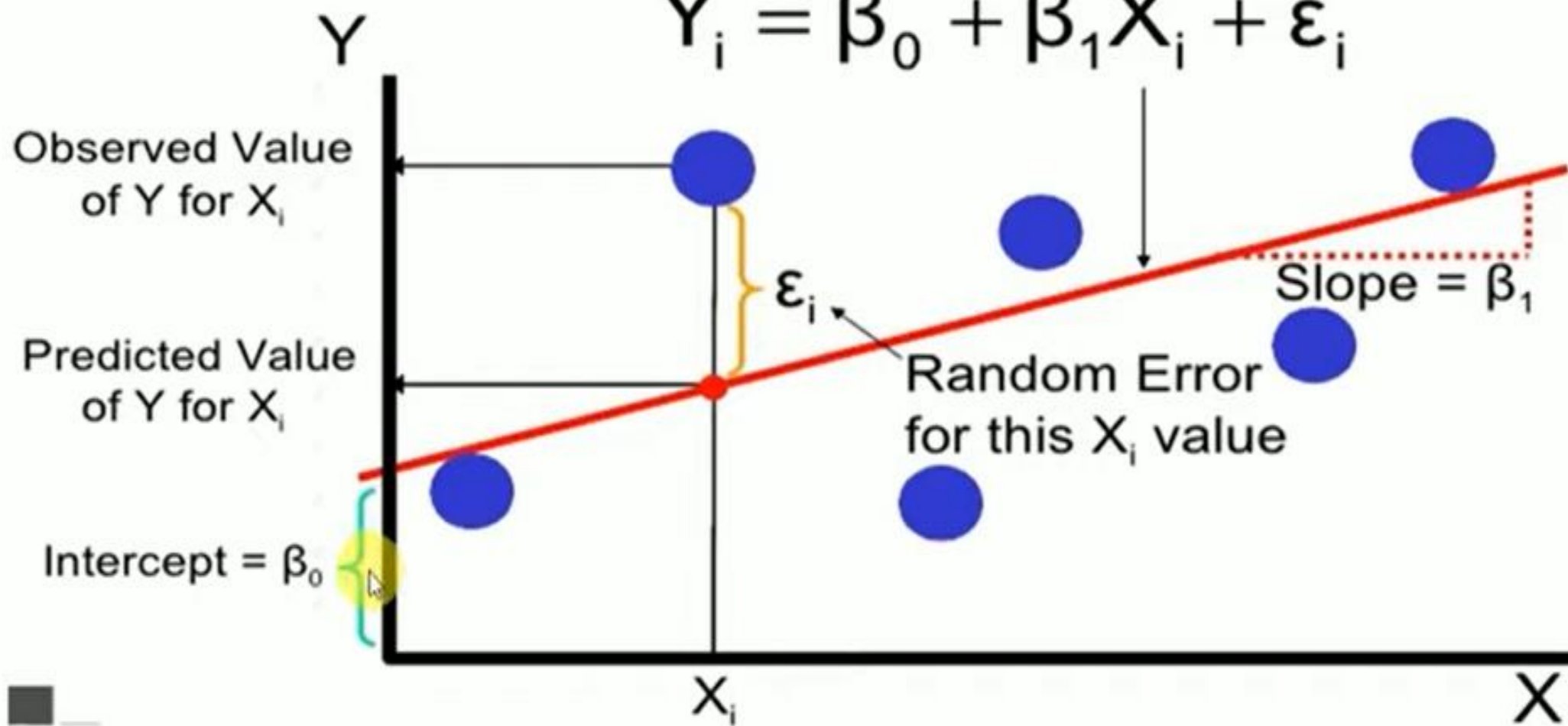
The diagram illustrates the Linear Regression Model equation: $Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$. Each term is labeled with an arrow pointing to it: Y_i is the Dependent Variable, β_0 is the Population Y intercept, β_1 is the Population Slope Coefficient, X_i is the Independent Variable, and ϵ_i is the Random Error term. Below the equation, two blue curly braces group the terms: the first brace under $\beta_0 + \beta_1 X_i$ is labeled 'Linear component', and the second brace under ϵ_i is labeled 'Random Error component'.

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

Labels and components:

- Dependent Variable: Y_i
- Population Y intercept: β_0
- Population Slope Coefficient: β_1
- Independent Variable: X_i
- Random Error term: ϵ_i
- Linear component: $\beta_0 + \beta_1 X_i$
- Random Error component: ϵ_i

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$



Multiple Regression Model

Multiple Regression Model with k Independent Variables:

The diagram shows the equation $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k + \varepsilon$. Above the equation, three labels in pink boxes are connected to specific terms by arrows: 'Y-intercept' points to β_0 , 'Population slopes' points to β_1 , β_2 , and β_k , and 'Random Error' points to ε .

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k + \varepsilon$$

Labels and arrows:

- Y-intercept points to β_0
- Population slopes points to β_1 , β_2 , and β_k
- Random Error points to ε

Example 1 of Regression equation to predict Glucose level

S.No.	Age (x)	Glucose Level (y)
1	43	99
2	21	65
3	25	79
4	42	75
5	57	87
6	59	81
7	55	?

Linear Regression Equation, $Y_i = b_0 + b_1 X_i$

where b_0 and b_1 can be computed as follows:

$$b_0 = \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2}$$

$$b_1 = \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2}$$

Step 1: Calculate the values of XY X^2

S.No.	Age (X)	Glucose Level (Y)	XY	X^2
1	43	99	4257	1849
2	21	65	1365	441
3	25	79	1975	625
4	42	75	3150	1764
5	57	87	4959	3249
6	59	81	4779	3481
Sum	247	486	20485	11409

Step 2: Find the value of b_0 and b_1

$$\begin{aligned} b_0 &= \frac{(\sum y)(\sum x^2) - (\sum x)(\sum xy)}{n(\sum x^2) - (\sum x)^2} \\ &= \frac{(486)(11409) - (247)(20485)}{6(11409) - (247)^2} \\ b_0 &= \frac{4848979}{7445} = 65.14 \end{aligned}$$

$$\begin{aligned} b_1 &= \frac{n(\sum xy) - (\sum x)(\sum y)}{n(\sum x^2) - (\sum x)^2} \\ b_1 &= \frac{6(20485) - (247)(486)}{6(11409) - (247)^2} \\ b_1 &= \frac{2868}{7445} = 0.385335 \end{aligned}$$

Step 3: Insert the values in the equation

$$Y_i = b_0 + b_1 X_i$$

$$Y = 65.14 + 0.385225X$$

Step 4: Prediction of the y for the given value of $x = 55$

$$\begin{aligned} Y &= 65.14 + 0.385225 * 55 \\ &= 86.327 \end{aligned}$$

So the glucose level predicted for the age of 55 is 86.327

2. Find the linear regression equation for the following data set

S.No.	x	y
1	2	3
2	4	7
3	6	5
4	8	10
Sum	20	25

S.No.	x	y	X ²	xy
1	2	3	4	6
2	4	7	16	28
3	6	5	36	30
4	8	10	64	80
Sum	20	25	120	144

$$\bullet b_1(\text{slope}) = \frac{4*144 - 20*25}{4*120 - 400} = 0.95$$

$$\bullet b_0 = \frac{25*120 - 20*144}{4*120 - 20*20} = 1.5$$

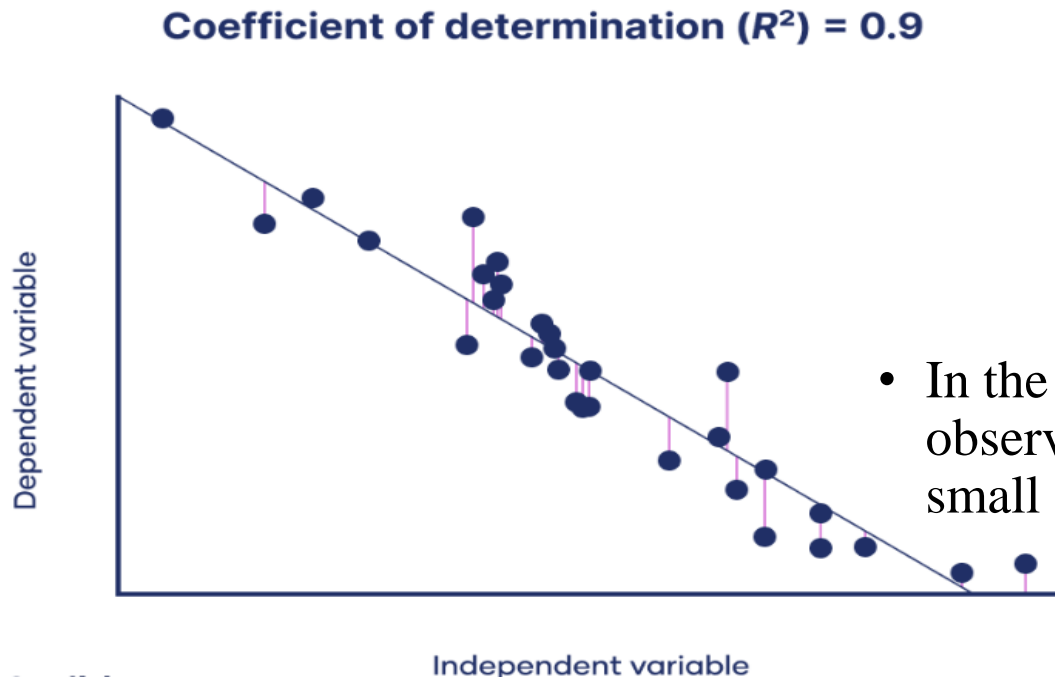
$$\bullet y = 0.95x + 1.5$$

Coefficient of determination (R^2)

- Determination coefficient tells the goodness of the fit of the model. The value of R^2 lies in range of 0-1.

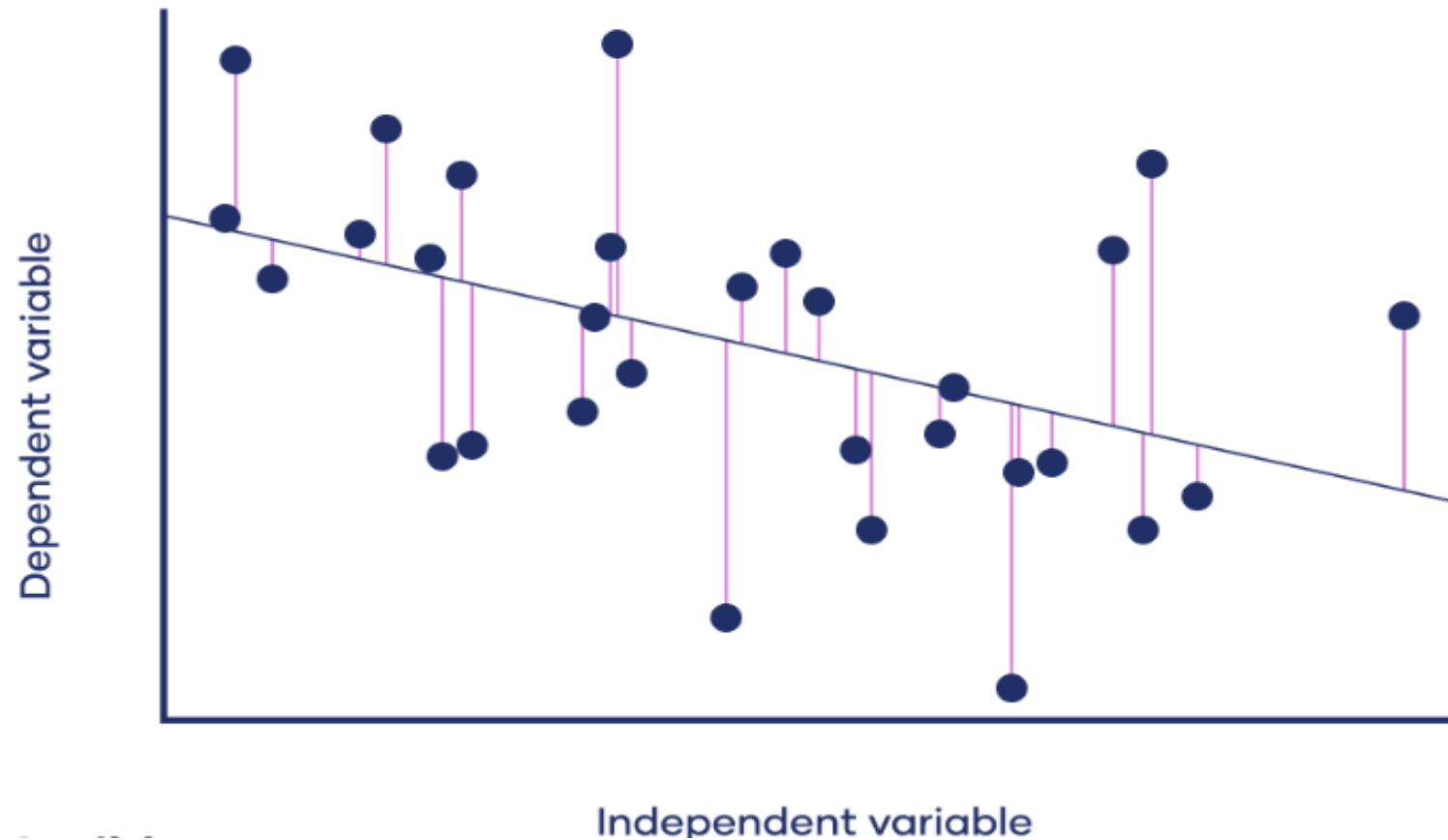
$$R^2 = 1 - \frac{\Sigma(y - y_p)^2}{\Sigma(y - \bar{y})^2}$$

- Where y is the observed output , y_p is the predicted output and \bar{y} is the mean value of observed output.



- In the given figure $R^2 = 0.9$ tells the model is a good fit model as the observed values and the predicted values are almost equal or have small error

Coefficient of determination (R^2) = 0.2



Model with $R^2 = 0.2$ tells the model is not a good fit model as the observed values and the predicted values are far away from each other or have large error. Model is not able to fit the data points

For the dataset given in example 2 determine the value of R^2

- $y=0.95x+1.5$ (regression model)
- $\bar{y}=6.25$ (mean of y)

S.No.	x	y	y_p	$y-y_p$	$(y-y_p)^2$	$(y-\bar{y})$	$(y-\bar{y})^2$
1	2	3	3.4	-0.4	0.16	-3.25	10.562
2	4	7	5.3	1.7	2.89	0.75	0.562
3	6	5	7.2	-2.2	4.84	-1.25	1.562
4	8	10	9.1	0.9	0.81	3.75	14.062
Sum	20	25	25		8.7		26.748

$$R^2 = 1 - \frac{\Sigma(y-y_p)^2}{\Sigma(y-\bar{y})^2}$$

$$R^2 = 1 - \frac{8.7}{26.748} = 0.67$$

As determination coefficient is close to 1, so predicted model is good.

3. The values of x and their corresponding y is given in the table

- (i) Find the regression line for the given data points
- (ii) Check whether it is a best fit line or not

S.No.	x	y
1	1	3
2	2	4
3	3	2
4	4	4
5	5	5

Finding the best fit line:

- When working with linear regression, our main goal is to find the best fit line that 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 (b_0 , b_1) gives a different line of regression, so we need to calculate the best values for b_0 and b_1 to find the best fit line, so to calculate this we use cost function.

Cost function

- The different values for weights or coefficient of lines (b_0, b_1) gives the different line of regression, and the cost function is used to estimate the values of the coefficient for the best fit line.
- Cost function optimizes the regression coefficients or weights. It measures how a linear regression model is performing.
- 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 of squared error occurred between the predicted values and actual values. It can be written as:
- For the above linear equation, MSE can be calculated as:

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

Gradient Descent

- Gradient descent is an optimization algorithm used to find the values of parameters of a function that minimizes a cost function. It is an iterative algorithm. We use gradient descent to update the parameters of the model. Parameters refer to coefficients in Linear Regression and weights in neural networks.
- Gradient descent is used to minimize the MSE by calculating the gradient of the cost function.
- A regression model uses gradient descent to update the coefficients of the line by reducing the cost function.
- It is done by a random selection of values of coefficient and then iteratively update the values to reach the minimum cost function.

Multiple Linear Regression

- In the previous topic, we have learned about Simple Linear Regression, where a single Independent/Predictor(X) variable is used to model the response variable (Y). But there may be various cases in which the response variable is affected by more than one predictor variable; for such cases, the Multiple Linear Regression algorithm is used.
- Moreover, Multiple Linear Regression is an extension of Simple Linear regression as it takes more than one predictor variable to predict the response variable.
- For MLR, the dependent or target variable(Y) must be the continuous/real, but the predictor or independent variable may be of continuous or categorical form.
- Each feature variable must model the linear relationship with the dependent variable.
- MLR tries to fit a regression line through a multidimensional space of data-points.

Assumptions for Multiple Linear Regression:

- A **linear relationship** should exist between the Target and predictor variables.
- The regression residuals must be **normally distributed**.
- MLR assumes little or **no multicollinearity** (correlation between the independent variable) in data.

Multiple Regression Model with k Independent Variables:

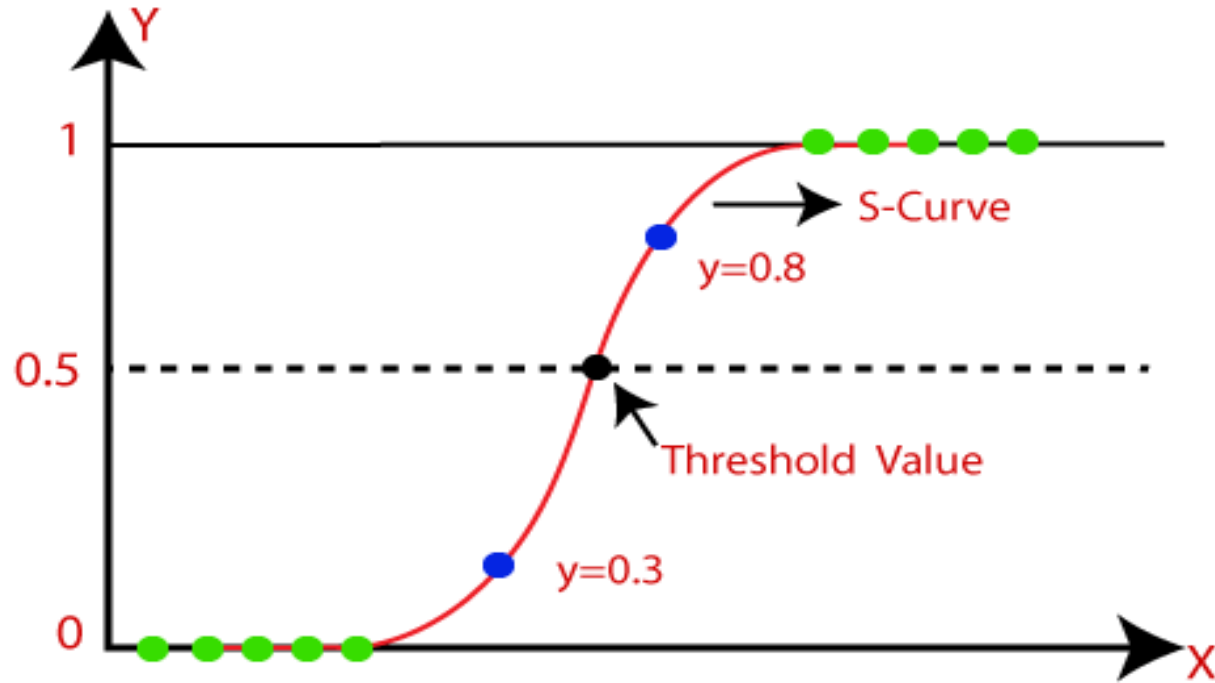
The diagram shows the equation $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k + \epsilon$. Above the equation, three labels in pink boxes are connected to specific terms by arrows: 'Y-intercept' points to β_0 , 'Population slopes' points to β_1 and β_2 , and 'Random Error' points to ϵ .

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k + \epsilon$$

Logistic Regression

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1.**
- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems.**
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.

Representation of S (sigmoid function)



$$y = S(x) = \frac{1}{1 + e^{-x}}$$
$$= \frac{e^x}{e^x + 1}$$

Logistic Regression is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm and based on the concept of probability. Here we use sigmoid function to map predictions to probabilities as given below.

$$P = \frac{1}{1 + e^{-y}}$$

Where:

$y = \beta_0 + \beta_1 x$ (in case of univariate Logistic regression)

$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \dots + \beta_n x_n$ (in case of multivariate logistic regression)

Univariate Logistic Regression means the output variable is predicted using only one predictor variable, while Multivariate Logistic Regression means output variable is predicted using multiple predictor variables.

The logistic regression function converts the values of **logits** also called **log-odds** that range from $-\infty$ to $+\infty$ to a range between **0 and 1**.

Now let us try to simplify what we said. Let **P** be the probability of occurrence of an event. So probability the event will not occur is **1-P**.

Odds is defined as the ratio of the probability of occurrence of a particular event to the probability of the event not occurring.

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

We know that logistic regression function gives us probability value. So we can write :

$$P = \frac{1}{1 + e^{-y}}$$

Now since we mentioned **log odds**, let us take the natural log of both sides of the **Odds** equation and substitute the value of **P**.

$$\ln(Odds) = \ln\left(\frac{P}{1-P}\right) = y$$

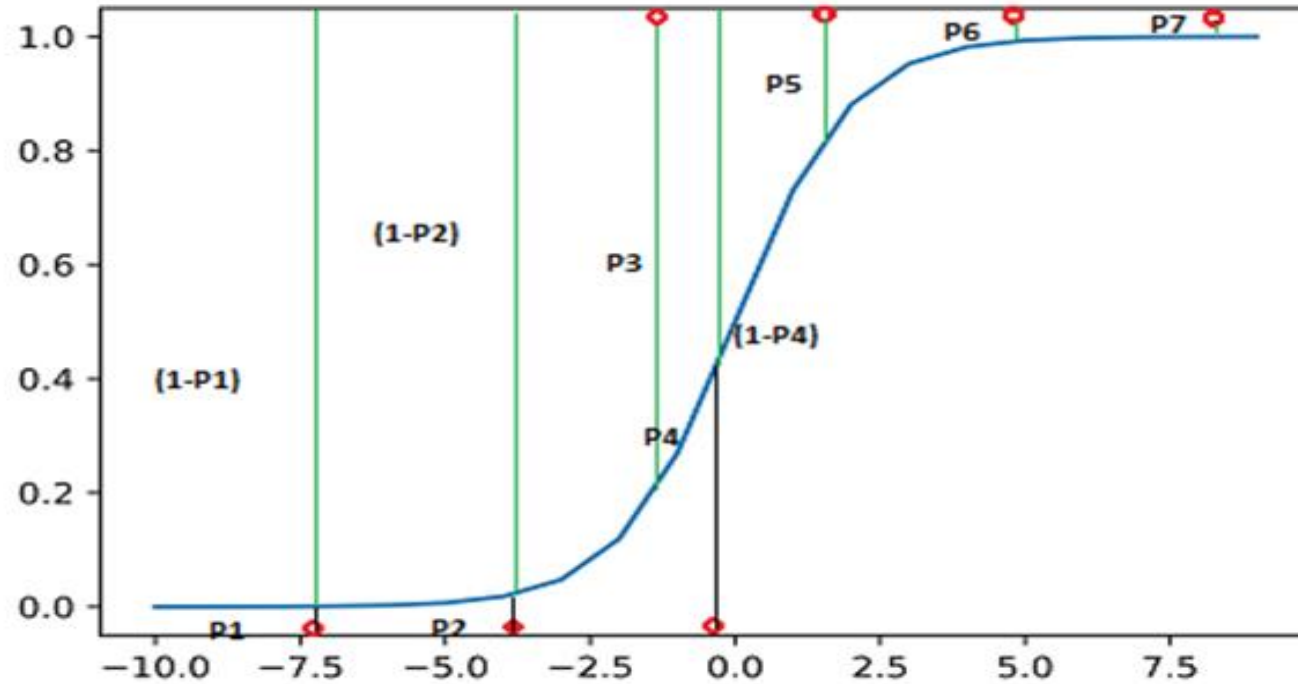
$$\ln\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 x$$

Thus we get a more simplified form of logistic regression function equation and we can say that **log odds** has linear relationship with the predictor variable **x**.

Maximum Likelihood Estimation

In order that our model predicts output variable as 0 or 1, we need to find the best fit sigmoid curve, that gives the optimum values of beta coefficients. That is we need to create an efficient boundary between the 0 and 1 values.

Now a cost function tells you how close your values are from actual. So here we need a cost function which maximizes the likelihood of getting desired output values. Such a cost function is called as **Maximum Likelihood Estimation (MLE)** function.



For points to be 0, we need the probabilities P_1 , P_2 and P_4 to be as minimum as possible and for points to be 1, we need the probabilities P_3 , P_5 , P_6 and P_7 to be as high as possible, for correct classification.

We can also say that $(1-P_1)$, $(1-P_2)$, P_3 , $(1-P_4)$, P_5 , P_6 and P_7 should be as high as possible.

The joint probability is nothing but the product of probabilities.

So the product :[$(1-P_1)*(1-P_2)* P_3*(1-P_4)*P_5*P_6*P_7$] should be maximum.

This joint probability function is nothing but our cost function which should be maximized in order to get a best fit sigmoid curve. Or we can say predicted values to be close to the actual values.

The major assumption of logistic regression

$$\log \frac{p(x_i)}{1 - p(x_i)} = \beta_0 + \beta_1 x_{i,1} + \dots + \beta_p x_{i,p} = \sum_{j=0}^p \beta_j x_{i,j}.$$

Here, we treat $x_{i,0} = 1$. We can also use vector notation to write

$$\log \frac{p(x_i; \beta)}{1 - p(x_i; \beta)} = x_i \beta.$$

Here, we view x_i as a row-vector and β as a column-vector.

The model in vector notation

$$p(x_i; \beta) = \frac{e^{x_i \beta}}{1 + e^{x_i \beta}}, \quad 1 - p(x_i; \beta) = \frac{1}{1 + e^{x_i \beta}},$$

Log likelihood for the i th observation:

$$\begin{aligned} l_i(\beta|x_i) &= (1 - y_i) \log [1 - p(x_i; \beta)] + y_i \log p(x_i; \beta) \\ &= \begin{cases} \log p(x_i; \beta) & \text{if } y_i = 1 \\ \log [1 - p(x_i; \beta)] & \text{if } y_i = 0 \end{cases} \end{aligned}$$

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, we use the concept of the threshold value, which defines the probability of either 0 or 1. Values above the threshold value tends to 1, and a value below the threshold values tends to 0.

Assumptions for Logistic Regression:

- The dependent variable must be categorical in nature.

Type of Logistic Regression:

- On the basis of the categories, Logistic Regression can be classified into three types:
- **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
- **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"
- **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Advantages & Disadvantages of Logistic Regression

- The main advantage of logistic regression is that it is much easier to set up and train than other machine learning and AI applications.
- Another advantage is that it is one of the most efficient algorithms when the different outcomes or distinctions represented by the data are linearly separable. This means that you can draw a straight line separating the results of a logistic regression calculation.
- One of the biggest attractions of logistic regression for statisticians is that it can help reveal the interrelationships between different variables and their impact on outcomes.

Logistic regression vs. linear regression

- The main difference between logistic and linear regression is that logistic regression provides a constant output, while linear regression provides a continuous output.
- In logistic regression, the outcome, or dependent variable, has only two possible values. However, in linear regression, the outcome is continuous, which means that it can have any one of an infinite number of possible values.
- Logistic regression is used when the response variable is categorical, such as yes/no, true/false and pass/fail. Linear regression is used when the response variable is continuous, such as hours, height and weight.
- For example, given data on the time a student spent studying and that student's exam scores, logistic regression and linear regression can predict different things.
- With logistic regression predictions, only specific values or categories are allowed. Therefore, logistic regression predicts whether the student passed or failed. Since linear regression predictions are continuous, such as numbers in a range, it can predict the student's test score on a scale of 0 to 100.

Confusion Matrix in Machine Learning

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an **error matrix**. Confusion matrix can be shown as given below:

		Actual Values		
		Positive	Negative	
Predicted Values	Positive	TP	FP	Type 1 Error
	Negative	FN	TN	

Understanding Confusion Matrix

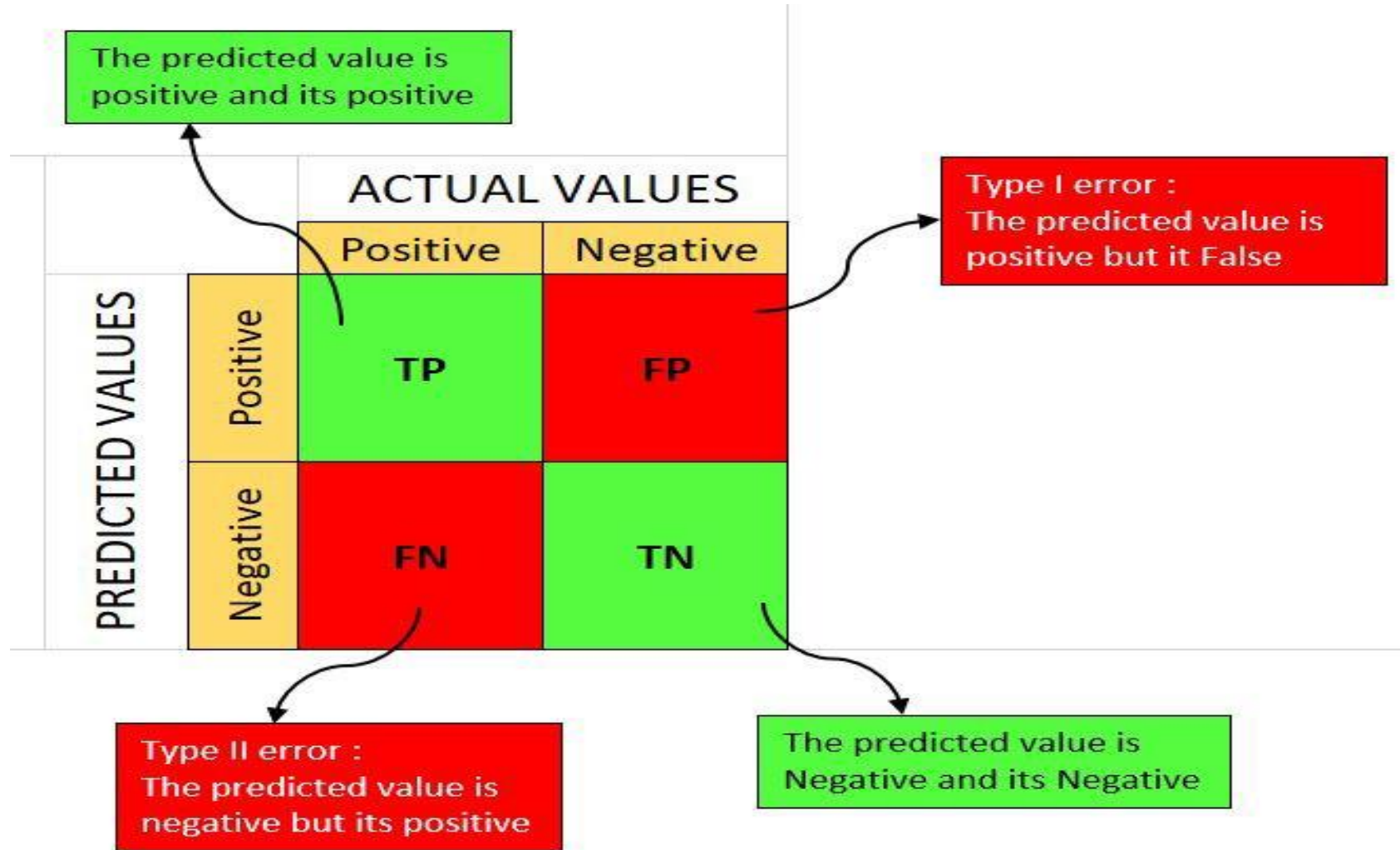
The following 4 are the basic terminology which will help us in determining the metrics we are looking for.

- **True Positives (TP)**: when the actual value is Positive and predicted is also Positive.
- **True negatives (TN)**: when the actual value is Negative and prediction is also Negative.
- **False positives (FP)**: When the actual is negative but prediction is Positive. Also known as the **Type 1 error**
- **False negatives (FN)**: When the actual is Positive but the prediction is Negative. Also known as the **Type 2 error**

A **good model** is one which has *high TP and TN rates*, while *low FP and FN rates*.

If you have an *imbalanced dataset* to work with, it's always better to use *confusion matrix* as the evaluation criteria for your machine learning model.





Understanding Confusion Matrix



Classification Measure

- A **confusion matrix** is a tabular summary of the number of **correct and incorrect predictions** made by a classifier. It is used to measure the performance of a classification model. It can be used to evaluate the performance of a classification model through the calculation of performance metrics like
 - a. Accuracy
 - b. Precision
 - c. Recall (TPR, Sensitivity)
 - d. F1-Score

Example 1. For given Classification model calculate accuracy, precision, recall and F1-score.

		PREDICTED VALUES	
		Positive (CAT)	Negative (DOG)
ACTUAL VALUES	Positive (CAT)	 TRUE POSITIVE 6 YOU ARE A CAT	 FALSE NEGATIVE 1 TYPE II ERROR YOU ARE A DOG
	Negative (DOG)	 FALSE POSITIVE 2 TYPE I ERROR YOU ARE A CAT	 TRUE NEGATIVE 11 YOU ARE NOT A CAT

Accuracy:

Accuracy simply measures how often the classifier makes the correct prediction. It's the ratio between the number of correct predictions and the total number of predictions.

		PREDICTED VALUES	
		Positive (1)	Negative (0)
ACTUAL VALUES	Positive (1)	6 TRUE POSITIVE	1 FALSE NEGATIVE
	Negative (0)	2 FALSE POSITIVE	11 TRUE NEGATIVE

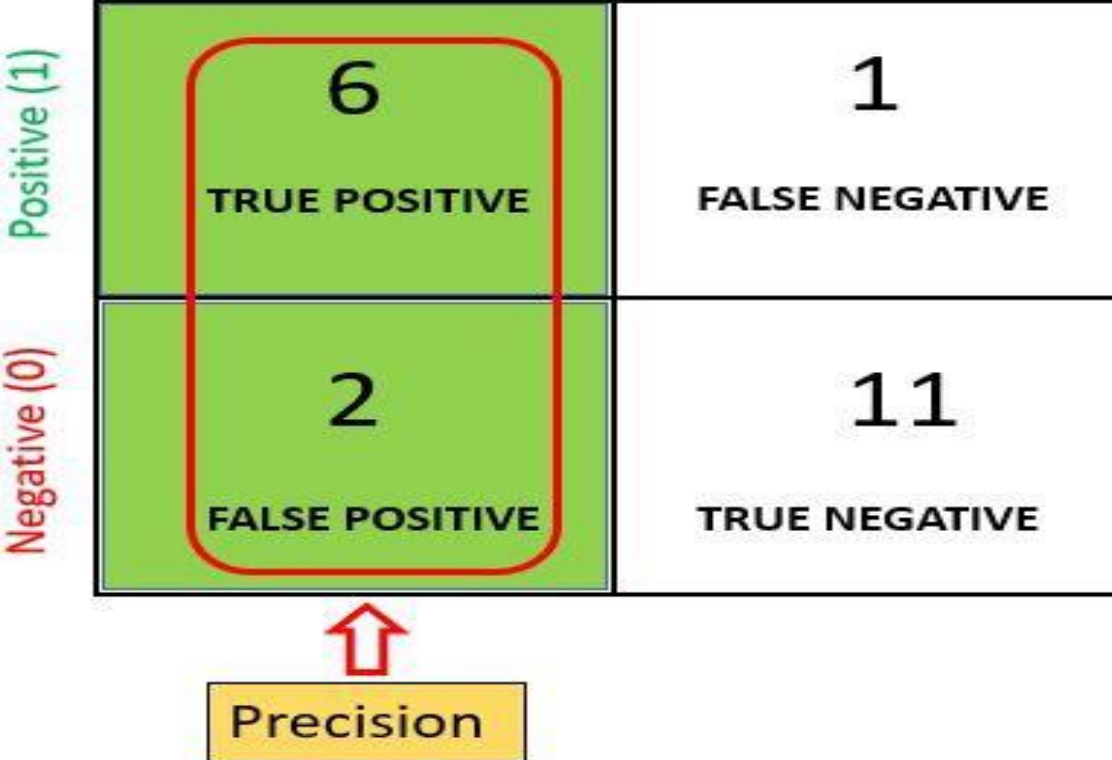
$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} = \frac{6 + 11}{6 + 11 + 2 + 1} = 85\%$$

Accuracy

Precision:

It is a measure of **correctness** that is achieved in **true prediction**. In simple words, it tells us how many predictions are *actually positive* out of all the *total positive predicted*.

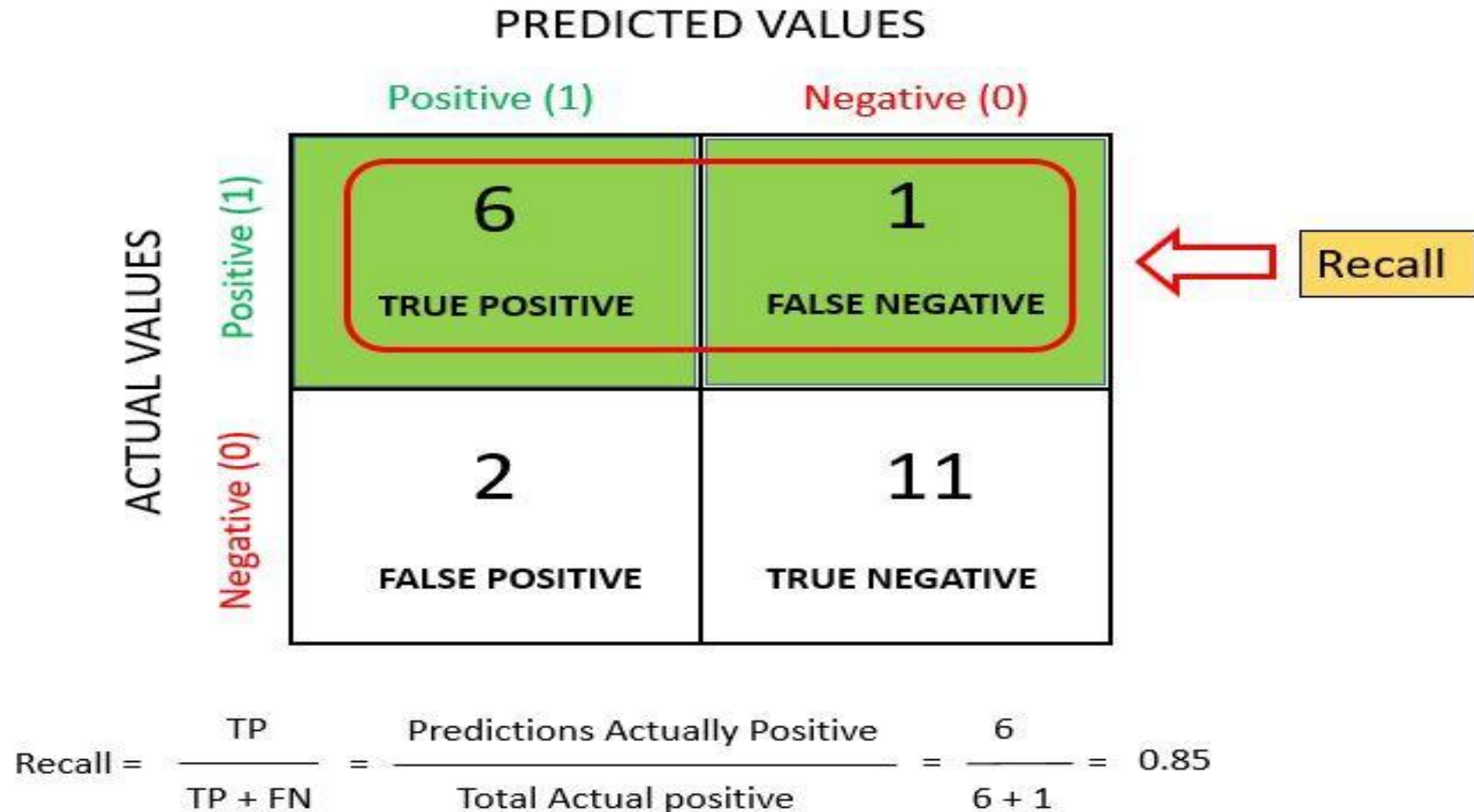
		PREDICTED VALUES	
		Positive (1)	Negative (0)
ACTUAL VALUES	Positive (1)	6 TRUE POSITIVE	1 FALSE NEGATIVE
	Negative (0)	2 FALSE POSITIVE	11 TRUE NEGATIVE



$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{Predictions Actually Positive}}{\text{Total Predicted positive}} = \frac{6}{6 + 2} = 0.75$$

Recall:

It is a measure of **actual observations** which are predicted **correctly**, i.e. how many observations of positive class are actually predicted as positive. It is also known as **Sensitivity**. ***Recall*** is a valid choice of evaluation metric when we want to capture *as many positives* as possible.



- The **F1 score** is a number between **0 and 1** and is the *harmonic mean of precision and recall*. We use harmonic mean because it is not sensitive to extremely large values, unlike simple averages.
- **F1 score** sort of maintains a **balance** between the *precision and recall* for your classifier. If your *precision is low*, the *F1 is low* and if the *recall is low* again your *F1 score is low*.

$$\text{F1-Score} = 2 * \frac{(\text{Recall} * \text{Precision})}{(\text{Recall} + \text{Precision})} = 2 * \frac{(0.85 * 0.75)}{(0.85 + 0.75)} = 0.79$$

Example 2.

- Determine the value of accuracy, error rate, precision and recall for the given confusion matrix

n=200	Predicted: No	Predicted: Yes	
Actual: No	TN=100	FP=15	115
Actual: Yes	FN=5	TP=80	85
	105	95	

ACCURACY

The no of predictions that the model got right

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) = 80 + 100 / 200 = 0.9$$

Error Rate

The no of predictions that the model predicted wrongly

$$\text{Error Rate} = (FP + FN) / (TP + TN + FP + FN) = 15 + 5 / 200 = 0.1$$

Precision

When the model predicts the positives, how often is it right?

$$\text{Precision} = TP / (TP + FP) = 80 / 95 = 0.86$$

Recall

When it's actually yes, how often does it predict yes?

$$\text{Recall} = (TP) / (TP + FN) = 80 / 85 = 0.94$$

n=200	Predicted: No	Predicted: Yes	
Actual: No	TN=100	FP=15	115
Actual: Yes	FN=5	TP=80	85
	105	95	

Prob 1. For given data calculate Recall, Precision & accuracy

y	y pred	output for threshold 0.6
0	0.5	0
1	0.9	1
0	0.7	1
1	0.7	1
1	0.3	0
0	0.4	0
1	0.5	0

Concept Learning

- Inducing general functions from specific training examples is a main issue of machine learning.
- Concept Learning in machine is acquiring the definition of a general category from given sample positive and negative training examples of the category.
- Concept Learning can be seen as a problem of searching through a predefined space of potential hypothesis for the hypothesis that best fits the training examples.
- The hypothesis space has a general-to-specific ordering of hypotheses, and the search can be efficiently organized by taking advantage of a naturally occurring structure over the hypothesis space.
- Many algorithms for concept learning organize the search through the hypothesis space by relying on a very useful structure that exists for any concept learning problem: a general-to-specific ordering of hypothesis.

Concept Learning

Example	<i>Sky</i>	<i>AirTemp</i>	<i>Humidity</i>	<i>Wind</i>	<i>Water</i>	<i>Forecast</i>	<i>EnjoySport</i>
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

- For each attribute, the hypothesis will either indicate by a "?" that any value is acceptable for this attribute,
- specify a single required value (e.g., Warm) for the attribute, or indicate by a " ϕ " that no value is acceptable.
- The most general hypothesis-that every day is a positive example-is represented by (?, ?, ?, ?, ?, ?)
- and the most specific possible hypothesis-that no day is a positive example is represented by ($\phi, \phi, \phi, \phi, \phi, \phi$)

Find S Algorithm: Finding a Maximally specific Hypothesis

Example	<i>Sky</i>	<i>AirTemp</i>	<i>Humidity</i>	<i>Wind</i>	<i>Water</i>	<i>Forecast</i>	<i>EnjoySport</i>
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

1. Initialize h to the most specific hypothesis in H

- $h_0 = \langle \phi, \phi, \phi, \phi, \phi, \phi \rangle$

2. For each positive training instance x

- For each attribute
- If the constraint is satisfied by x
- Then do nothing
- Else replace attribute in h by the next most general constraint that is satisfied by x

$$x_1 = \langle \text{Sunny}, \text{Warm}, \text{Normal}, \text{Strong}, \text{Warm}, \text{Same} \rangle$$

$$h_1 = \langle \text{Sunny}, \text{Warm}, \text{Normal}, \text{Strong}, \text{Warm}, \text{Same} \rangle$$

Example	<i>Sky</i>	<i>AirTemp</i>	<i>Humidity</i>	<i>Wind</i>	<i>Water</i>	<i>Forecast</i>	<i>EnjoySport</i>
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

- Iteration 2

$x_2 = \langle \text{Sunny, Warm, High, Strong, Warm, Same} \rangle$

$h_2 = \langle \text{Sunny, Warm, ?, Strong, Warm, Same} \rangle$

- Iteration 3: its a negative case therefore previous hypothesis is kept as it is

$h_3 = \langle \text{Sunny, Warm, ?, Strong, Warm, Same} \rangle$

- Iteration 4: Next outcome is positive, so we consider it for hypothesis generation

$x_4 = \langle \text{Sunny, Warm, High, Strong, Cool, Change} \rangle$

$h_4 = \langle \text{Sunny, Warm, ?, Strong, ?, ?} \rangle$

h_4 is the maximally specific hypothesis.

Candidate Elimination Algorithm

1. Firstly define the most specific and most generic boundaries.
2. If the first outcome is positive, we go to the most generic boundary and check the consistency. If it is consistent we retain it otherwise we write the next general hypothesis. Then we go to the specific boundary.
3. If the first outcome is negative, we go to the most specific boundary and check the consistency. If it is consistent we retain it otherwise we write the next specific hypothesis. After that we go to the general boundary.
4. Eliminate the hypotheses those are not consistent with the given data.
4. After applying above step for each outcome, we will get the most specific and most generic boundaries.

Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

$S_0 = \langle \phi, \phi, \phi, \phi, \phi, \phi \rangle$ as ϕ is not matching with any attribute, So S_1 will be generated from attributes.

$S_1 = \langle \text{Sunny, Warm, Normal, Strong, Warm, Same} \rangle$

$S_2 = \langle \text{Sunny, Warm, ?, Strong, Warm, Same} \rangle$

$S_2:$ $S_3:$

$\langle \text{Sunny, Warm, ?, Strong, Warm, Same} \rangle$

S_4

$\langle \text{Sunny, Warm, ?, Strong, ?, ?} \rangle$

$G_0 = \langle ?, ?, ?, ?, ?, ? \rangle$ As first outcome is +ve So we go with generic boundary. Also as ? Will match with all, therefore next G_1 will be same. After that go to specific boundary.

$G_1 = \langle ?, ?, ?, ?, ?, ? \rangle$ As the next outcome is also +ve So we go with generic boundary. Also as ? Will match with all, therefore next G_2 will be same. Then go to the specific boundary.

$G_2 = \langle ?, ?, ?, ?, ?, ? \rangle$

$G_3:$

$\langle \text{Sunny, ?, ?, ?, ?, ?} \rangle$

$\langle \text{?, Warm, ?, ?, ?, ?} \rangle$

$\langle \text{?, ?, Normal, ?, ?, ?} \rangle$

$\langle \text{?, ?, ?, ?, Cool, ?} \rangle$

$\langle \text{?, ?, ?, ?, ?, Same} \rangle$

$G_4:$

$\langle \text{Sunny, ?, ?, ?, ?, ?} \rangle$

$\langle \text{?, Warm, ?, ?, ?, ?} \rangle$

S

⟨Sunny, Warm, ?, Strong, ?, ?⟩

⟨Sunny, ?, ?, Strong, ?, ?⟩

⟨Sunny, Warm, ?, ?, ?, ?⟩

⟨?, Warm, ?, Strong, ?, ?⟩

G

⟨Sunny, ?, ?, ?, ?, ?⟩

⟨?, Warm, ?, ?, ?, ?⟩

Bayesian Learning

Bayesian reasoning provides a probabilistic approach to inference.

Bayesian learning methods are relevant to our study of machine learning for two different reasons:

- Bayesian learning algorithms that calculate explicit probabilities for hypotheses, such as the naive Bayes classifier, are among the most practical approaches to certain types of learning problems.
- They provide a useful perspective for understanding many learning algorithms that do not explicitly manipulate probabilities.

Features of Bayesian Learning

- Each observed training example can incrementally decrease or increase the estimated probability that a hypothesis is correct This provides a more flexible approach to learning than algorithms that completely eliminate a hypothesis if it is found to be inconsistent with any single example.
- Prior knowledge can be combined with observed data to determine the final probability of a hypothesis.
- Bayesian methods can accommodate hypotheses that make probabilistic predictions.
- New instances can be classified by combining the predictions of multiple hypotheses, weighted by their probabilities.
- Even in cases where Bayesian methods prove computationally intractable, they can provide a standard of optimal decision making against which other practical methods can be measured.
- Bayesian statistics is an approach based on Bayes Theorem

Bayes Theorem

- In machine learning we are often interested in determining the best hypothesis from some space H , given the observed training data D . One way to specify what we mean by the best hypothesis is to say that we demand the most probable hypothesis, given the data D plus any initial knowledge about the prior probabilities of the various hypothesis in H .
- Bayes theorem provides a way to calculate the probability of a hypothesis based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself.
- Bayes theorem is the cornerstone of Bayesian learning methods because it provides a way to calculate the posterior probability $P(h|D)$, from the prior probability $P(h)$, together with $P(D)$ and $P(D|h)$.

Bayes theorem:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

- $P(h)$ - the initial probability that hypothesis h holds, before we have observed the training data.
- $P(D)$ - the prior probability that training data D will be observed (i.e., the probability of D given no knowledge about which hypothesis holds).
- $P(D|h)$ - the probability of observing data D given some world in which hypothesis h holds.
- $P(h|D)$ - the posterior probability of h , because it reflects our confidence that h holds after we have seen the training data D .
- **MAP**- In many learning scenarios, the learner considers some set of candidate hypotheses H and is interested in finding the most probable hypothesis $h \in H$ given the observed data D . Any such maximally probable hypothesis is called a maximum a posteriori (MAP) hypothesis.
- We can determine the MAP hypotheses by using Bayes theorem to calculate the posterior probability of each candidate hypothesis.

$$\begin{aligned}
 h_{MAP} &\equiv \operatorname{argmax}_{h \in H} P(h|D) \\
 &= \operatorname{argmax}_{h \in H} \frac{P(D|h) P(h)}{P(D)} \\
 &= \operatorname{argmax}_{h \in H} P(D|h) P(h)
 \end{aligned}$$

Example- Does Patient have Cancer or not

- The available data is from a particular laboratory test with two possible outcomes:
+(positive) and -(negative).
- We have prior knowledge that over the entire population of people only .008 have this disease.
- Furthermore, the lab test is only an imperfect indicator of the disease. The test returns a correct positive result in only 98% of the cases in which the disease is actually present and a correct negative result in only 97% of the cases in which the disease is not present. In other cases, the test returns the opposite result.
- The above situation can be summarized by the following probabilities:

$$P(cancer) = 0.008 \qquad P(\neg cancer) = 0.992$$

$$P(+|cancer) = 0.98 \qquad P(-|cancer) = 0.02$$

$$P(+|\neg cancer) = 0.03 \qquad P(-|\neg cancer) = 0.97$$

- Observe a new patient for whom the lab test returns a positive result and diagnose the patient is having cancer or not?

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

$$P(\text{cancer}|+) = P(+|\text{cancer}) * P(\text{cancer}) = 0.98 * 0.008 = 0.0078$$

$$P(\neg \text{cancer}|+) = P(+|\neg \text{cancer}) * P(\neg \text{cancer}) = 0.03 * 0.992 = 0.0298$$

- Observe a new patient for whom the lab test returns a negative result and diagnose the patient is having cancer or not using Bayes rule?

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

$$P(cancer|-) = P(-|cancer) * P(cancer) = 0.02 * 0.008 = 0.00016$$

$$P(\neg cancer|-) = P(-|\neg cancer) * P(\neg cancer) = 0.97 * 0.992 = 0.96224$$

Bayes Theorem and Concept Learning

Brute-Force based concept learning

- Consider a case of concept learning problem :
- H = Hypothesis space define over an instance space X
- X = instance space
- $C: X \rightarrow \{0, 1\}$

C is the target concept which is Boolean Valued function.

For the above we can design a straight forward concept learning algorithm to output h_{MAP}

BRUTE-FORCE MAP LEARNING algorithm

1. For each hypothesis h in H , calculate the posterior probability

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

2. Output the hypothesis h_{MAP} with the highest posterior probability

$$h_{MAP} = \operatorname{argmax}_{h \in H} P(h|D)$$

We may choose the probability distributions $P(h)$ and $P(D/h)$ in any way we wish, to describe our prior knowledge about the learning task. Here let us choose them to be consistent with the following assumptions:

1. The training data D is noise free (i.e., $d_i = c(x_i)$).
2. The target concept c is contained in the hypothesis space H
3. We have no a priori reason to believe that any hypothesis is more probable than any other.

Given these assumptions, what values should we specify for $P(h)$? Given no prior knowledge that one hypothesis is more likely than another, it is reasonable to assign the same prior probability to every hypothesis h in H .

$$P(h) = \frac{1}{|H|} \quad \text{for all } h \text{ in } H$$

- The probability of observing the target values $D = (d_1 \dots d_m)$ for the fixed set of instances $(x_1 \dots x_m)$ given a world in which hypothesis h holds (i.e., given a world in which h is the correct description of the target concept c). Since we assume noise-free training data, the probability of observing classification d_i given h is just 1 if $d_i = h(x_i)$ and 0 if $d_i \neq h(x_i)$.

$$P(D|h) = \begin{cases} 1 & \text{if } d_i = h(x_i) \text{ for all } d_i \text{ in } D \\ 0 & \text{otherwise} \end{cases}$$

In other words, the probability of data D given hypothesis h is 1 if D is consistent with h , and 0 otherwise

Let us consider the first step of this algorithm, which uses Bayes theorem to compute the posterior probability $P(h/D)$ of each hypothesis h given the observed training data D .

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

First consider the case where h is inconsistent with the training data D . $P(D/h)$ to be 0 when h is inconsistent with D , we have

$$P(h|D) = \frac{0 \cdot P(h)}{P(D)} = 0 \text{ if } h \text{ is inconsistent with } D$$

The posterior probability of a hypothesis inconsistent with D is zero. Now consider the case where h is consistent with D . $P(D/h)$ to be 1 when h is consistent with D , we have

$$\begin{aligned} P(h|D) &= \frac{1 \cdot \frac{1}{|H|}}{P(D)} \\ &= \frac{1 \cdot \frac{1}{|H|}}{\frac{|V_{SH,D}|}{|H|}} \\ &= \frac{1}{|V_{SH,D}|} \text{ if } h \text{ is consistent with } D \end{aligned}$$

where V_{SH} is the subset of hypotheses from H that are consistent with D i.e., V_{SH} is the version space of H with respect to D

- To summarize, Bayes theorem implies that the posterior probability $P(h/D)$ under our assumed $P(h)$ and $P(D/h)$ is

$$P(h|D) = \begin{cases} \frac{1}{|VS_{H,D}|} & \text{if } h \text{ is consistent with } D \\ 0 & \text{otherwise} \end{cases}$$

Bayes Optimal Classifier

- So far we have considered the question "what is the most probable *hypothesis* given the training data?" In fact, the question that is often of most significance is the closely related question "what is the most probable *classification* of the new instance given the training data?"
- It may seem that this second question can be answered by simply applying the MAP hypothesis to the new instance, we need to check whether it is the only best way and if not what could be the other option.

- Consider a hypothesis space containing three hypotheses, h_1 , h_2 , and h_3 . Suppose that the posterior probabilities of these hypotheses given the training data are .4, **.3**, and **.3** respectively.

$$P(h_1/D)=0.4$$

$$P(h_2/D)=0.3$$

$$P(h_3/D)=0.3$$

From the above information we could identify that

- h_1 is the MAP hypothesis
- Suppose a new instance x is encountered, which is classified positive by **h_1** , but negative by h_2 and h_3 . Taking all hypotheses into account, the probability that x is positive is .4 (the probability associated with h_1), and the probability that it is negative is therefore .6. **The most probable classification (negative) in this case is different from the classification generated by the MAP hypothesis.**

- In general, the most probable classification of the new instance is obtained by combining the predictions of all hypotheses, weighted by their posterior probabilities.
- If the possible classification of the new example can take on any value v_j from some set V , then the probability $P(v_j/D)$ that the correct classification for the new instance is v_j

$$P(v_j|D) = \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

The optimal classification of the new instance is the value v_j , for which $P(v_j/D)$ is maximum.

Bayes optimal Classification :

$$\operatorname{argmax}_{v_j \in V} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D)$$

Bayes optimal classification method maximizes the probability that the new instance is classified correctly, given the available data, hypothesis space, and prior probabilities over the hypothesis.

Example

- Let the set of possible classifications of the new instance is $V = (+, -)$ and posterior probabilities of these hypotheses given the training data are .4, .3, and .3 respectively.
- The possible classification of the new example can be computed as follows:

$$P(h_1|D) = .4, P(\ominus|h_1) = 0, P(\oplus|h_1) = 1$$

$$P(h_2|D) = .3, P(\ominus|h_2) = 1, P(\oplus|h_2) = 0$$

$$P(h_3|D) = .3, P(\ominus|h_3) = 1, P(\oplus|h_3) = 0$$

Therefore

$$\sum_{h_i \in H} P(\oplus|h_i)P(h_i|D) = .4$$

$$\sum_{h_i \in H} P(\ominus|h_i)P(h_i|D) = .6$$

and

$$\operatorname{argmax}_{v_j \in \{\oplus, \ominus\}} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D) = \ominus$$

Naive Bayes Classifier

- One highly practical Bayesian learning method is the naive Bayes learner, often called the **naive Bayes classifier**.
- Naive Bayes algorithm is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems.
- The naive Bayes classifier applies to learning tasks where each instance \mathbf{x} is described by a conjunction of attribute values and where the target function $f(\mathbf{x})$ can take on any value from some finite set V .
- A set of training examples of the target function is provided, and a new instance is presented, described by the tuple of attribute values **($a_1, a_2 \dots a_n$)**.
- The learner is asked to predict the target value, or classification, for this new instance.

- The Bayesian approach to classifying the new instance is to assign the most probable target value, VMAP given the attribute values $(a_1, a_2 \dots a_n)$ that describe the instance.

$$v_{MAP} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j | a_1, a_2 \dots a_n)$$

- We can use Bayes theorem to rewrite this expression as

$$\begin{aligned} v_{MAP} &= \underset{v_j \in V}{\operatorname{argmax}} \frac{P(a_1, a_2 \dots a_n | v_j) P(v_j)}{P(a_1, a_2 \dots a_n)} \\ &= \underset{v_j \in V}{\operatorname{argmax}} P(a_1, a_2 \dots a_n | v_j) P(v_j) \end{aligned}$$

- It is easy to estimate each of the $P(v_j)$ simply by counting the frequency with which each target value v_j occurs in the training data. However, estimating the different $P(a_1, a_2 \dots a_n, / v_j)$ terms is not feasible unless we have a very, very large set of training data. The problem is that the number of these terms is equal to the number of possible instances times the number of possible target values. Therefore, we need to see every instance in the instance space many times in order to obtain reliable estimates.

Naive Bayes Classifier:

- The naive Bayes classifier is based on the simplifying assumption that the attribute values are conditionally independent given the target value.
- In other words, the assumption is that given the target value of the instance, the probability of observing the conjunction a_1, a_2, \dots, a_n , is just the product of the probabilities for the individual attributes:

$$P(a_1, a_2, \dots, a_n | v_j) = \prod_i P(a_i | v_j)$$

- Naive Bayes Classification: $v_{NB} = \operatorname{argmax}_{v_j \in V} P(v_j) \prod_i P(a_i | v_j)$

where v_{NB} denotes the target value output by the naive Bayes classifier.

To summarize, the naive Bayes learning method involves a learning step in which the various $P(v_j)$ and $P(a_i | v_j)$ terms are estimated, based on their frequencies over the training data. The set of these estimates corresponds to the learned hypothesis and used to classify new instance.

Example: Consider a fictional dataset that describes the weather conditions for playing tennis. Test it on a new instance **today=(sunny, cool, high, strong)**

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

1. Calculate prior probabilities $P(v_j)$ of target values(v_j)

- Here $V=\{\text{Yes, No}\}$
- Total instances in a data set =14
- No. of times target value is yes =9
- No. of times target value is No =5
- So, $P(\text{Play tennis}=\text{Yes}) = 9/14 = 0.64$
 $P(\text{Play tennis}=\text{No}) = 5/14 = 0.36$

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

2. Calculate Current or conditional probability of individual attribute $P(a_i/v_j)$

- Here we have 4 attributes $a=\{\text{outlook, temperature, Humidity, Wind}\}$
- So calculate $P(\text{outlook}=\text{sunny}/\text{play tennis}=\text{Yes})$
- $P(\text{outlook}=\text{sunny}/\text{play tennis}=\text{No})$
- $P(\text{outlook}=\text{overcast}/\text{play tennis}=\text{Yes})$
- $P(\text{outlook}=\text{overcast}/\text{play tennis}=\text{No})$
- $P(\text{outlook}=\text{Rain}/\text{play tennis}=\text{Yes})$
- $P(\text{outlook}=\text{Rain}/\text{play tennis}=\text{No})$

Similarly calculate conditional probability for all attributes

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Outlook	Y	N
sunny	2 / 9	3 / 5
overcast	4 / 9	0
rain	3 / 9	2 / 5
Temperature		
hot	2 / 9	2 / 5
mild	4 / 9	2 / 5
cool	3 / 9	1 / 5

Humidity	Y	N
high	3 / 9	4 / 5
normal	6 / 9	1 / 5
Windy		
Strong	3 / 9	3 / 5
Weak	6 / 9	2 / 5

Calculate the v_{NB} for classifying the new instance

New instance is **today=(sunny, cool, high, strong)**

$$v_{NB} = \operatorname{argmax}_{v_j \in \{yes, no\}} P(v_j) \prod_i P(a_i | v_j)$$

$$= \operatorname{argmax}_{v_j \in \{yes, no\}} P(v_j) \quad P(Outlook = sunny | v_j) P(Temperature = cool | v_j)$$

$$\cdot P(Humidity = high | v_j) P(Wind = strong | v_j)$$

$$v_{NB}(yes) = P(yes) P(sunny|yes) P(cool|yes) P(high|yes) P(strong|yes) = .0053$$

$$v_{NB}(no) = P(no) P(sunny|no) P(cool|no) P(high|no) P(strong|no) = .0206$$

$$v_{NB}(yes) = \frac{v_{NB}(yes)}{v_{NB}(yes) + v_{NB}(no)} = 0.205$$

$$v_{NB}(no) = \frac{v_{NB}(no)}{v_{NB}(yes) + v_{NB}(no)} = 0.795$$

As $v_{NB}(yes) < v_{NB}(no)$ so final classification is “**no**”

Bayesian Belief Network

- The naive Bayes classifier, which assumes that *all* the variables are conditionally independent given the value of the target variable.
- This assumption dramatically reduces the complexity of learning the target function. When it is met, the naive Bayes classifier outputs the optimal Bayes classification. However, in many cases this conditional independence assumption is clearly overly restrictive.
- Bayesian belief networks allows conditional independence assumptions that apply to *subsets* of the variables. Thus, Bayesian belief networks provide an intermediate approach that is less constraining than the global assumption of conditional independence made by the naive Bayes classifier, but more tractable than avoiding conditional independence assumptions altogether.

- A Bayesian belief network describes the probability distribution over a set of variables.
- Consider an arbitrary set of random variables $Y_1 \dots Y_n$ where each variable Y_i can take on the set of possible values $V(Y_i)$.
- We define the *joint space* of the set of variables Y to be the cross product $V(Y_1) \times V(Y_2) \times \dots \times V(Y_n)$.
- The probability distribution over this joint space is called the *joint probability distribution*.
- The joint probability distribution specifies the probability for each of the possible variable bindings for the tuple $(Y_1 \dots Y_n)$.

Conditional Independence

- Let X , Y , and Z be three discrete-valued random variables. We say that X is *conditionally independent* of Y given Z if the probability distribution governing X is independent of the value of Y given a value for Z ; that is, if

$$(\forall x_i, y_j, z_k) \ P(X = x_i | Y = y_j, Z = z_k) = P(X = x_i | Z = z_k)$$

where $x_i \in V(X)$, $y_j \in V(Y)$, and $z_k \in V(Z)$.

We commonly write the above expression in abbreviated form as $P(X/Y, Z) = P(X/Z)$.

This definition of conditional independence can be extended to sets of variables as well.

We say that the set of variables $X_1 \dots X_i$ is conditionally independent of the set of variables $Y_1 \dots Y_m$ given the set of variables $Z_1 \dots Z_n$, if

$$P(X_1 \dots X_i | Y_1 \dots Y_m, Z_1 \dots Z_n) = P(X_1 \dots X_i | Z_1 \dots Z_n)$$

- There is a relation between this definition and our use of conditional independence in the definition of the naive Bayes classifier
- The naive Bayes classifier assumes that the instance attribute A_1 is conditionally independent of instance attribute A_2 given the target value V . This allows the naive Bayes classifier to calculate $P(A_1, A_2|V)$ in

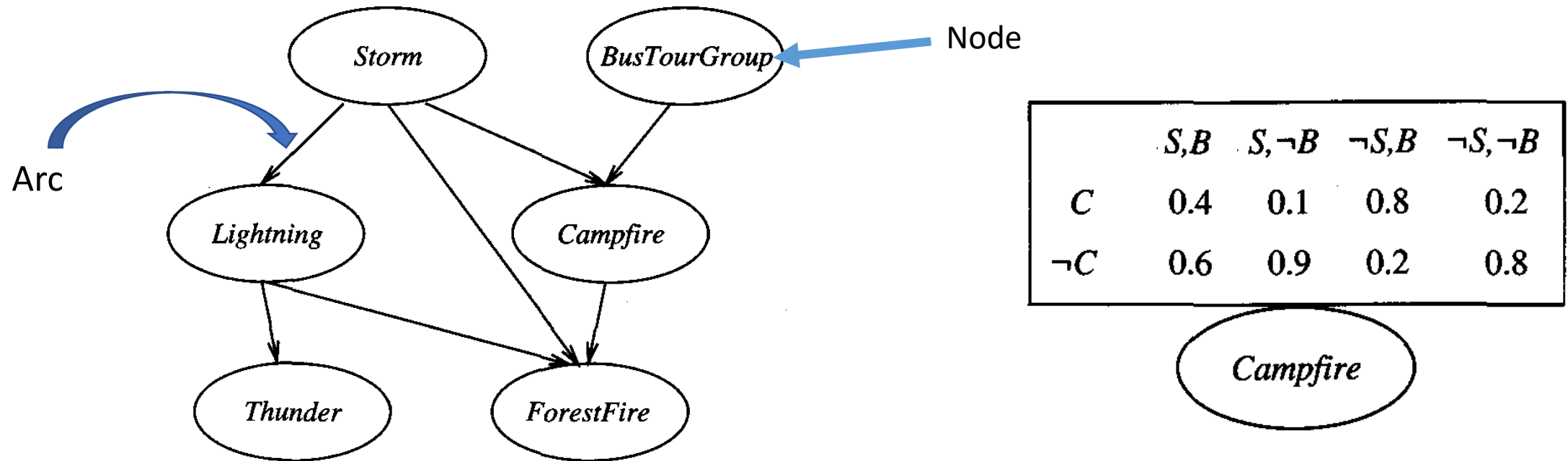
$$v_{NB} = \operatorname{argmax}_{v_j \in V} P(v_j) \prod_i P(a_i|v_j)$$

as follows

$$\begin{aligned} P(A_1, A_2|V) &= P(A_1|A_2, V)P(A_2|V) \text{ ----(Product rule of Probability)} \\ &= P(A_1|V)P(A_2|V) \end{aligned}$$

Representation of Bayesian Network

- A **Bayesian belief network** (Bayesian network for short) represents the joint probability distribution for a set of variables
- For example, the Bayesian network in given Figure represents the joint probability distribution over the boolean variables **Storm**, **Lightning**, **Thunder**, **ForestFire**, **Campfire**, and **BusTourGroup**. In general,



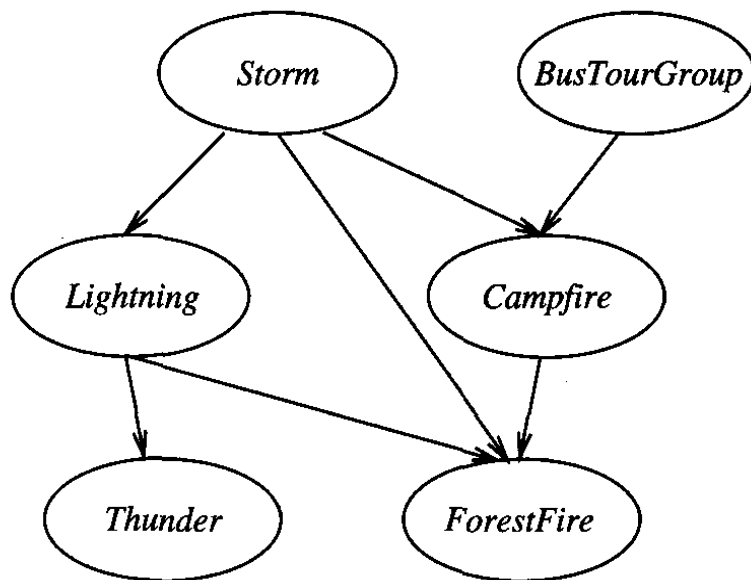
- A Bayesian network represents the joint probability distribution by specifying a set of conditional independence assumptions represented by a **directed acyclic graph(DAG)**, together with sets of local **conditional probabilities(CPT)**

- For each variable two types of information is given which are
 - Each variable in the joint space is represented by a node in the Bayesian network. For each variable two types of information are specified. First, the network arcs represent the assertion that the variable is conditionally independent of its non descendants in the network given its immediate predecessors in the network.
 - Second, a conditional probability table is given for each variable, describing the probability distribution for that variable given the values of its immediate predecessors.
- The joint probability for any desired assignment of values (y_1, \dots, y_n) to the tuple of network variables ($Y_1 \dots Y_n$) can be computed by the formula

$$P(y_1, \dots, y_n) = \prod_{i=1}^n P(y_i | \text{Parents}(Y_i))$$

- where $\text{Parents}(Y_i)$ denotes the set of immediate predecessors of Y_i in the network. Note the values of $P(y_i | \text{Parents}(y_i))$ are precisely the values stored in the conditional probability table associated with node Y_i .

Illustration



	S, B	$S, \neg B$	$\neg S, B$	$\neg S, \neg B$
C	0.4	0.1	0.8	0.2
$\neg C$	0.6	0.9	0.2	0.8

Campfire

Consider the node *Campfire*. The network nodes and arcs represent the assertion that *Campfire* is conditionally independent of its non descendants *Lightning* and *Thunder*, given its immediate parents *Storm* and *BusTourGroup*. This means that once we know the value of the variables *Storm* and *BusTourGroup*, the variables *Lightening* and *Thunder* provide no additional information about *Campfire*. The right side of the figure shows the conditional probability table associated with the variable *Campfire*. The top left entry in this table, for example, expresses the assertion that

$$P(\text{Campfire} = \text{True} / \text{Storm} = \text{True}, \text{BusTourGroup} = \text{True}) = 0.4$$

Inference

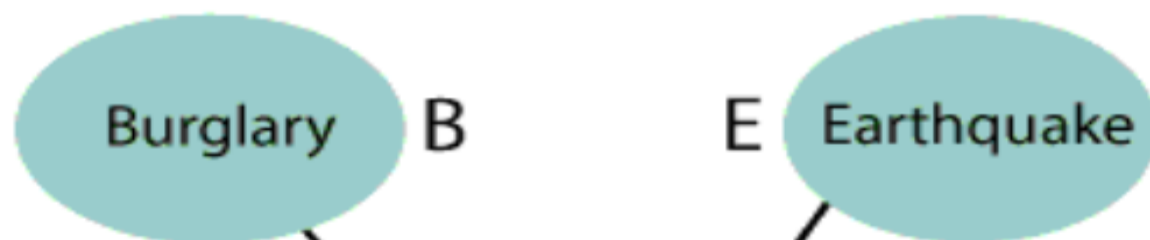
- A Bayesian network is used to infer the value of some target variable (e.g., *Forest Fire*) given the observed values of the other variables. As we are dealing with random variables the target variable can't be a single determined value.
- Therefore we wish to infer is the probability distribution for the target variable, which specifies the probability that it will take on each of its possible values given the observed values of the other variables.
- This inference step can be straightforward if values for all of the other variables in the network are known exactly.
- In the more general case we may wish to infer the probability distribution for some variable (e.g., *Forest Fire*) given observed values for only a subset of the other variables (e.g., *Thunder* and *Bus Tour Group* may be the only observed values available).
- In general, a Bayesian network can be used to compute the probability distribution for any subset of network variables given the values or distributions for any subset of the remaining variables.

Example:

Harry installed a new burglar alarm at his home to detect burglary. The alarm reliably responds at detecting a burglary but also responds for minor earthquakes. Harry has two neighbors David and Sophia, who have taken a responsibility to inform Harry at work when they hear the alarm. David always calls Harry when he hears the alarm, but sometimes he got confused with the phone ringing and calls at that time too. On the other hand, Sophia likes to listen to high music, so sometimes she misses to hear the alarm. Here we would like to compute the probability of Burglary Alarm.

Problem: Calculate the probability that alarm has sounded, but there is neither a burglary, nor an earthquake occurred, and David and Sophia both called the Harry.

T	0.002
F	0.998



T	0.001
F	0.999

B	E	P(A=T)	P(A=F)
T	T	0.94	0.06
T	F	0.95	0.05
F	T	0.69	0.31
F	F	0.001	0.999

A	P(D=T)	P(D=F)
T	0.91	0.09
F	0.05	0.95



A	P(S=T)	P(S=F)
T	0.75	0.25
F	0.02	0.98

Solution

From the formula of joint distribution, we can write the problem statement in the form of probability distribution:

- $P(S, D, A, \neg B, \neg E) =$
- $P(S|A) * P(D|A) * P(A|\neg B \wedge \neg E) * P(\neg B) * P(\neg E).$

$$= 0.75 * 0.91 * 0.001 * 0.998 * 0.999$$

$$= \mathbf{0.00068045}.$$

- Hence, a Bayesian network can answer any query about the domain by using Joint distribution.

