Unit-3

Evaluating Hypotheses: Estimating hypotheses Accuracy

- For estimating hypothesis accuracy, statistical methods are applied.

- **Evaluating hypotheses:**
- Whenever you form a hypothesis for a given training data set, for example, you came up with a hypothesis for the EnjoySport example where the attributes of the instances decide if a person will be able to enjoy their favorite sport or not.
- Now to test or evaluate how accurate the considered hypothesis is we use different statistical measures. Evaluating hypotheses is an important step in training the model.

- **To evaluate the hypotheses precisely focus on these points:**

➢ First, how well does this estimate the accuracy of a hypothesis across additional examples, given the observed accuracy of a hypothesis over a limited sample of data?

➢ Second, how likely is it that if one theory outperforms another across a set of data, it is more accurate in general?

➢ Third, what is the best strategy to use limited data to both learn and measure the accuracy of a hypothesis?

- **Motivation:** There are instances where the accuracy of the entire model plays a huge role in the model is adopted or not. For example, consider using a training model for Medical treatment. We need to have a high accuracy so as to depend on the information the model provides.
- When we need to learn a hypothesis and estimate its future accuracy based on a small collection of data, **we face two major challenges:**

1. **Bias in the estimation:** Initially, the observed accuracy of the learned hypothesis over training instances is a poor predictor of its accuracy over future cases.

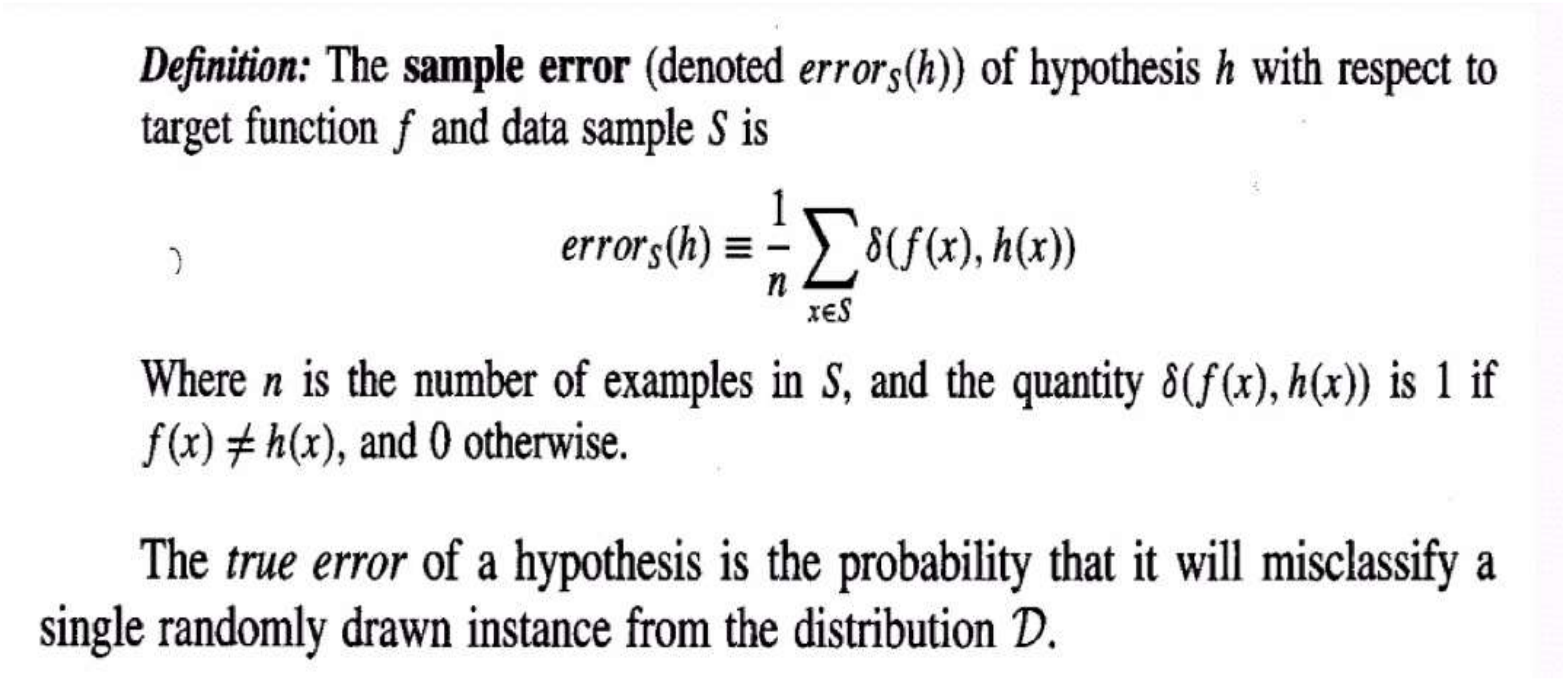
Because the learned hypothesis was generated from previous instances, future examples will likely yield a skewed estimate of hypothesis correctness.

2. Estimation variability: Depending on the nature of the particular set of test examples, even if the hypothesis accuracy is tested over an unbiased set of test instances independent of the training examples, the measurement accuracy can still differ from the true accuracy.

- The anticipated variance increases as the number of test examples decreases.

- So, we need to address:
- What is the best estimate of the accuracy of h over future instances taken from the same distribution, given a hypothesis h and a data sample containing n examples picked at random according to the distribution D?
- What is the margin of error in this estimate of accuracy?

True Error and Sample Error



Definition: The **sample error** (denoted $error_S(h)$) of hypothesis h with respect to target function f and data sample S is

$$error_S(h) \equiv \frac{1}{n} \sum_{x \in S} \delta(f(x), h(x))$$

Where n is the number of examples in S, and the quantity $\delta(f(x), h(x))$ is 1 if $f(x) \neq h(x)$, and 0 otherwise.

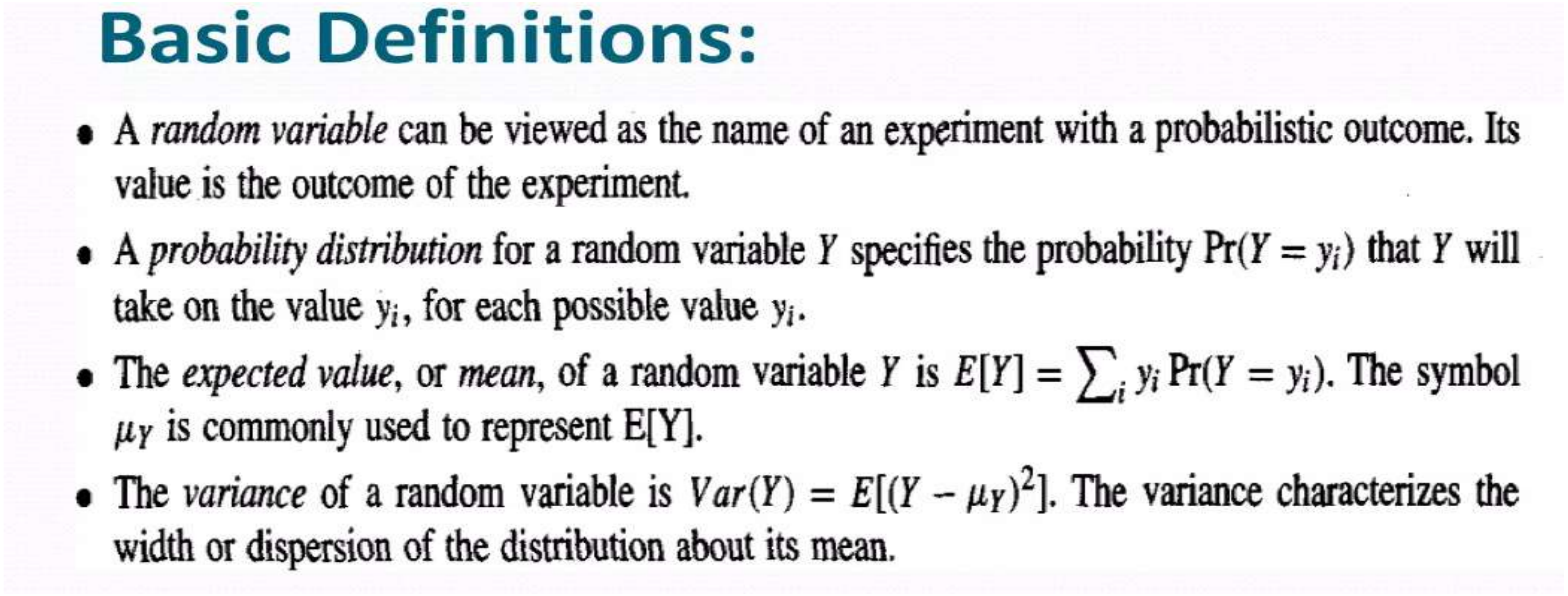
The *true error* of a hypothesis is the probability that it will misclassify a single randomly drawn instance from the distribution $\mathcal{D}$.

Definition: The **true error** (denoted $error_{\mathcal{D}}(h)$) of hypothesis h with respect to target function f and distribution $\mathcal{D}$, is the probability that h will misclassify an instance drawn at random according to $\mathcal{D}$.

$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[f(x) \neq h(x)]$$

Here the notation $\Pr_{x \in \mathcal{D}}$ denotes that the probability is taken over the instance distribution $\mathcal{D}$.

Basics of Sampling Theory



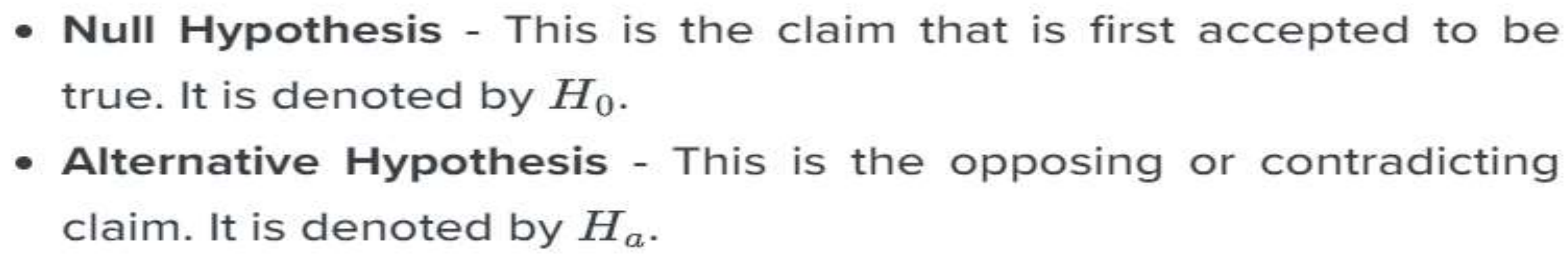
Basic Definitions:

- A *random variable* can be viewed as the name of an experiment with a probabilistic outcome. Its value is the outcome of the experiment.
- A *probability distribution* for a random variable Y specifies the probability $\Pr(Y = y_i)$ that Y will take on the value y_i, for each possible value y_i.
- The *expected value*, or *mean*, of a random variable Y is $E[Y] = \sum_i y_i \Pr(Y = y_i)$. The symbol μ_Y is commonly used to represent E[Y].
- The *variance* of a random variable is $Var(Y) = E[(Y - \mu_Y)^2]$. The variance characterizes the width or dispersion of the distribution about its mean.

- The *standard deviation* of Y is $\sqrt{Var(Y)}$. The symbol σ_Y is often used used to represent the standard deviation of Y.
- The *Binomial distribution* gives the probability of observing r heads in a series of n independent coin tosses, if the probability of heads in a single toss is p.
- The *Normal distribution* is a bell-shaped probability distribution that covers many natural phenomena.
- The *Central Limit Theorem* is a theorem stating that the sum of a large number of independent, identically distributed random variables approximately follows a Normal distribution.
- An *estimator* is a random variable Y used to estimate some parameter p of an underlying population.
- The *estimation bias* of Y as an estimator for p is the quantity $(E[Y] - p)$. An unbiased estimator is one for which the bias is zero.
- A *N% confidence interval* estimate for parameter p is an interval that includes p with probability N%.

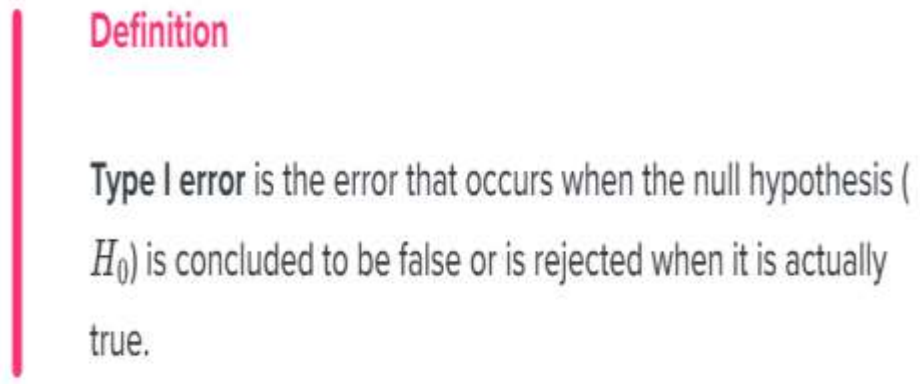
Types I & Type II Errors in Hypothesis Testing

- Hypothesis testing is used to find out if a claim is true or not but during this process of testing, some errors can occur.



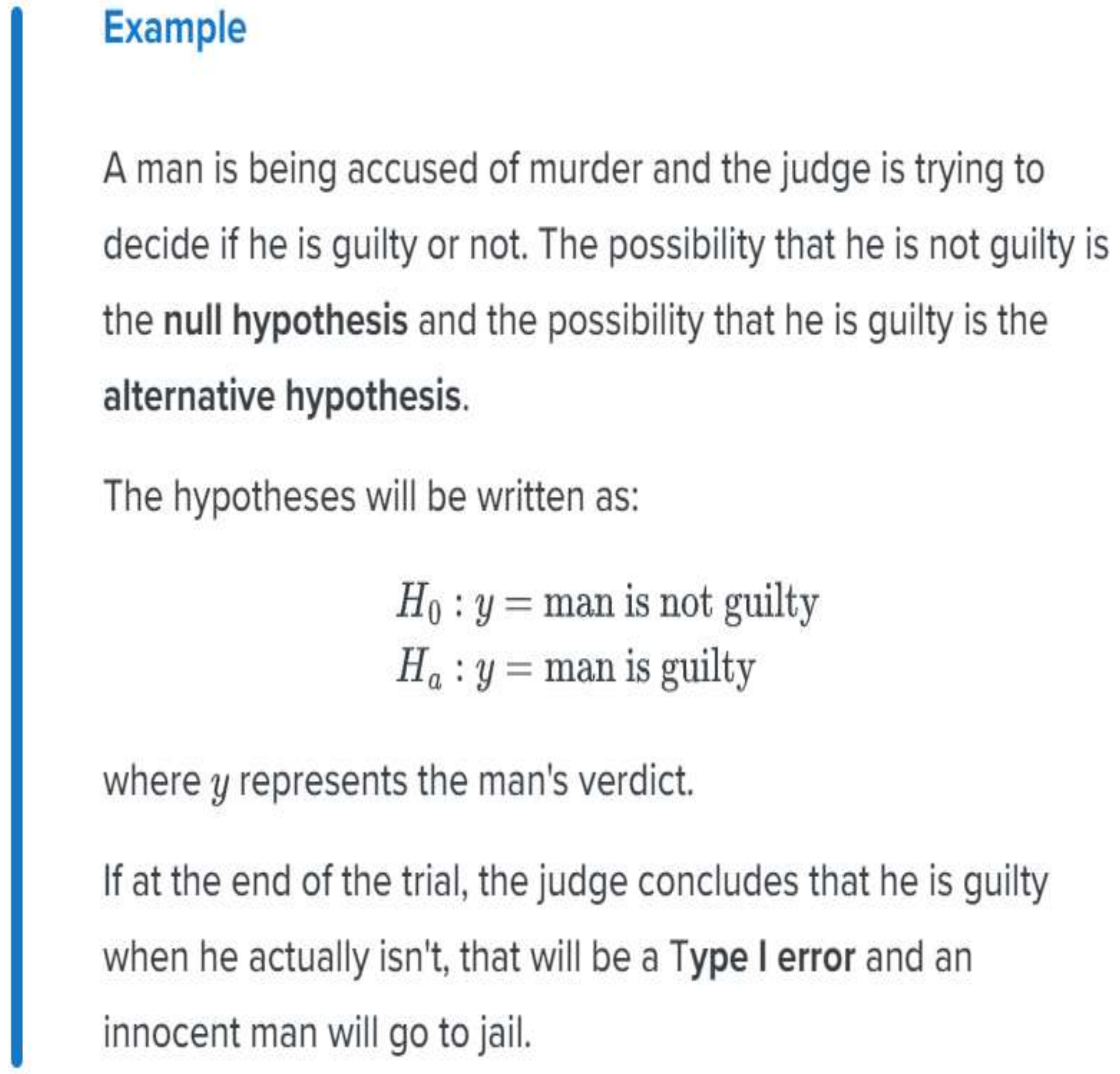
- **Null Hypothesis** - This is the claim that is first accepted to be true. It is denoted by H_0.
- **Alternative Hypothesis** - This is the opposing or contradicting claim. It is denoted by H_a.

- There are two types of errors in hypothesis testing, Type I and Type II errors.



Definition

Type I error is the error that occurs when the null hypothesis (H_0) is concluded to be false or is rejected when it is actually true.



Example

A man is being accused of murder and the judge is trying to decide if he is guilty or not. The possibility that he is not guilty is the **null hypothesis** and the possibility that he is guilty is the **alternative hypothesis**.

The hypotheses will be written as:

$$H_0 : y = \text{man is not guilty}$$
$$H_a : y = \text{man is guilty}$$

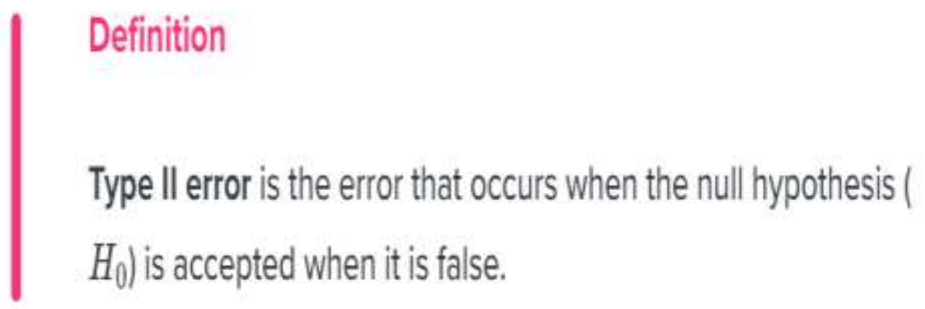
where y represents the man's verdict.

If at the end of the trial, the judge concludes that he is guilty when he actually isn't, that will be a **Type I error** and an innocent man will go to jail.

The probability of a Type I error is called the **level of significance** of the test and it is denoted by α. So, if you have $\alpha = 0.05$, it means that the level of significance of the test is 0.05.
You can also define α as the probability of rejecting a null hypothesis when it is true.

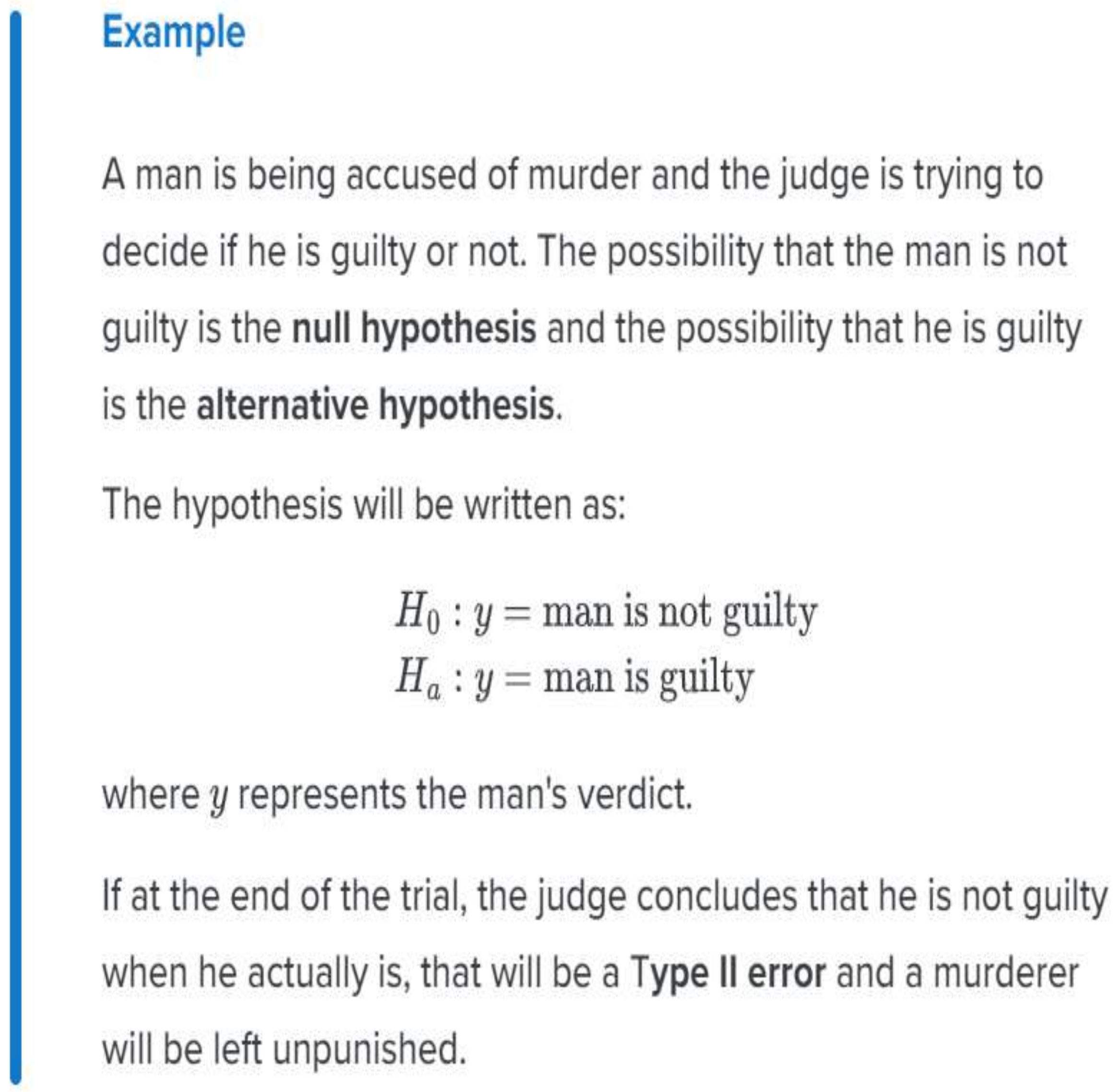
$$P(rejecting\, H_0/H_0\; is\; true) = \alpha$$

Type II Error in Hypothesis Testing



Definition

Type II error is the error that occurs when the null hypothesis (H_0) is accepted when it is false.



Example

A man is being accused of murder and the judge is trying to decide if he is guilty or not. The possibility that the man is not guilty is the **null hypothesis** and the possibility that he is guilty is the **alternative hypothesis**.

The hypothesis will be written as:

$$H_0 : y = \text{man is not guilty}$$
$$H_a : y = \text{man is guilty}$$

where y represents the man's verdict.

If at the end of the trial, the judge concludes that he is not guilty when he actually is, that will be a **Type II error** and a murderer will be left unpunished.

The probability of a Type II error is denoted by β

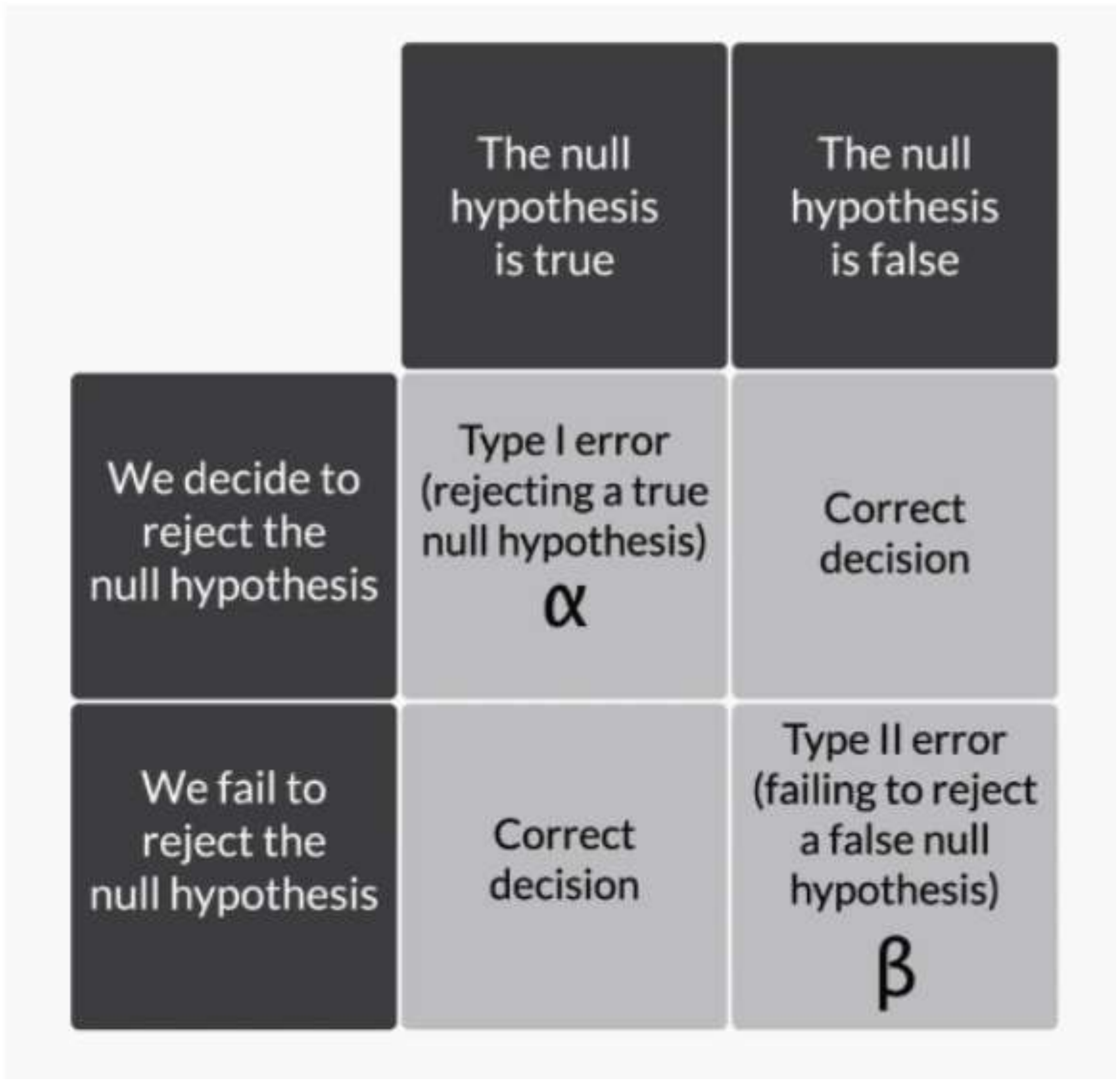
$$P(\text{accepting } H_0 : \; H_0 \text{ is false}) = \beta.$$

Either Type I or Type II error can occur in any test but the error that is more serious or significant depends on the situation.

Example

A doctor's diagnosis of a patient is to be confirmed with a test. The null hypothesis is that the patient has the disease and the alternative hypothesis is that he doesn't have the disease.

If the test concludes that the patient has the disease when he doesn't then that's a Type II error. In this case, a Type II error is much more serious than a Type I because assuring someone that they are healthy when they are not can cause serious problems.



	The null hypothesis is true	The null hypothesis is false
We decide to reject the null hypothesis	Type I error (rejecting a true null hypothesis) α	Correct decision
We fail to reject the null hypothesis	Correct decision	Type II error (failing to reject a false null hypothesis) β

How to Compare Machine Learning Algorithms

- ML algorithms can be compared on common grounds and they can be analyzed easily based on following parameters:

1. Time complexity: time complexity can be very different during training and testing.

2. Space complexity: Space complexity measures how much memory an algorithm needed to run in terms of the input size. A ML program could not be successfully run if a ML algorithm loads too much data into the working memory of a machine.

3. Sample complexity: Sample complexity measures the number of training examples needed to train the network in order to guarantee a valid generalization. For example, deep neural network has high sample complexity since lots of training data are needed to train it.

4. Bias-variance tradeoff: Different ML algorithms would have different bias-variance tradeoff. The bias errors come from the fact that a model is biased towards a specific solution or assumption.

5. Online and Offline: Online and offline learning refers to the way a machine learning software learns to update the model. Online learning means training data can be presented one at a time so that parameters can be updated immediately when new data are available.

- Offline learning, however, requires the training to start over again (re-train the whole model) when new data presented in order to update the parameters. If an algorithm is an online one, it would be efficient since the parameters used in production can be updated in real-time to reflect the effect of new data. ID3 Decision tree algorithm is an example of offline learning

6. Parallelizability: A parallel algorithm means that an algorithm can complete multiple operations at a given time.

This can be done by distributing the workloads across different workers, like processors in a single machine or multiple machines.

The nature of k-nearest neighbors (k-NN) model allows it to be easily run on multiple machine at the same time.

7. Parametricity: The concept of parametricity is widely used in the fields of statistical learning. A parametric model means the number of parameters of a model is fixed while the number of parameters of a non-parametric model grows when more data are available.

- Parametric model is very common in machine learning. Examples are linear regression, neural networks and many other ML models. k-NN and SVM (support vector machine).

Bayes Theorem

- Bayes’ Theorem is named after Reverend Thomas Bayes.
- It is used to find the probability of an event, based on prior knowledge of conditions that might be related to that event. It is a further case of conditional probability.

Bayes theorem is also known as the Bayes Rule or Bayes Law. It is used to determine the conditional probability of event A when event B has already happened. The general statement of Bayes' theorem is "The conditional probability of an event A, given the occurrence of another event B, is equal to the product of the event of B, given A and the probability of A divided by the probability of event B." i.e.

$$P(A|B) = P(B|A)P(A) / P(B)$$

where,

P(A) *and* ***P(B)*** *are the probabilities of events A and B*

P(A|B) *is the probability of event A when event B happens*

P(B|A) *is the probability of event B when A happens*

Bayes Theorem and Concept Learning

- In Machine Learning, there are situations where you need to make decisions or predictions based on incomplete or noisy information.
- Bayes Theorem allows calculating the probability of a hypothesis or event given the observed data. You can make more accurate predictions or decisions by incorporating prior knowledge and updating it with new evidence.
- The theorem is based on the concept of conditional probability. This is the probability of an event occurring, given that another event has already happened.

- In Machine Learning, the Bayes Theorem is often applied in the context of Bayesian inference. This allows you to make predictions or estimate unknown quantities based on observed data.

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

P(A|B) is the posterior probability of event A given event B

P(B|A) is the likelihood of event B given event A

P(A) is the prior probability of event A

P(B) is the prior probability of event B

- **Imagine you have a dataset of emails, where each email is labeled as either “spam” or “not spam.”**
- The objective is to construct a Machine Learning model that can autonomously classify incoming emails as either spam or non-spam, relying on the content of the messages as the basis for classification.
- To apply Bayes Theorem in this scenario, make certain assumptions.
- Assume that the words occurring in the emails are independent of each other. This means that the presence or absence of one word does not affect the presence or absence of another word.
- This simplifies the problem and allows to apply the theorem effectively.

Let’s define some terms for better understanding:

- P(Spam) represents the prior probability of an email being spam.
- P(Not Spam) represents the prior probability that an email is not spam.
- P(Word|Spam) is the probability of a word occurring in a spam email.
- P(Word|Not Spam) is the probability of a word appearing in a non-spam email.

- To classify a new email as spam or not, you can use the Bayes Theorem as follows:

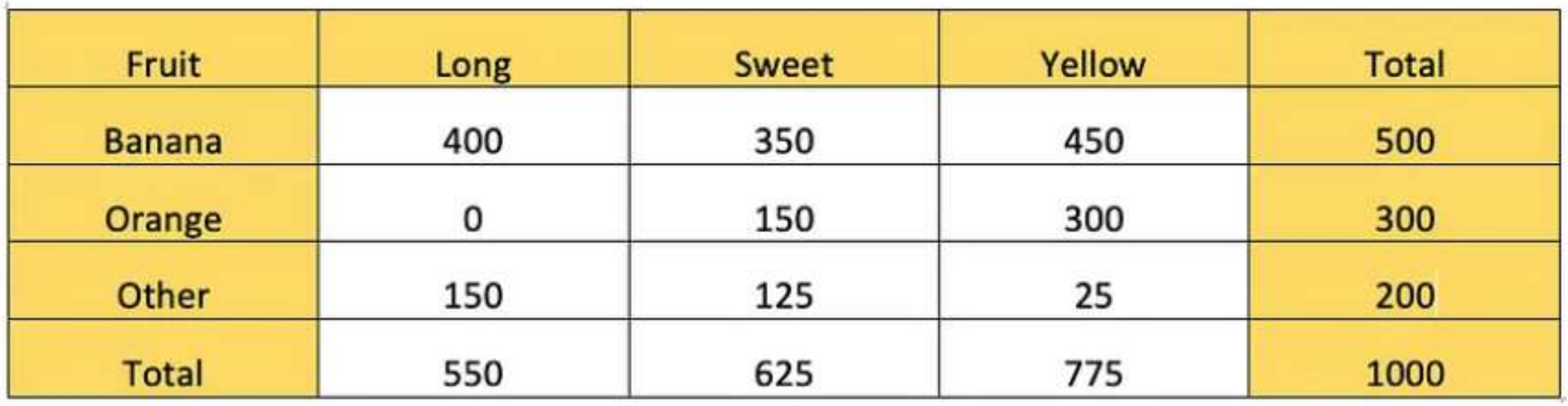
```
P(Spam|Word) = (P(Word|Spam) * P(Spam)) / P(Word)
```

- Here, P(Spam|Word) is the posterior probability of an email being spam, given the occurrence of a particular word.

Naïve Bayes Classifier Algorithm

- 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 simple and most effective Classification algorithms which helps in building the fast machine learning 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**.

- Example: There is some research about fruit characteristics from the 1000 total sample of different fruits



Fruit	Long	Sweet	Yellow	Total
Banana	400	350	450	500
Orange	0	150	300	300
Other	150	125	25	200
Total	550	625	775	1000

- If we have a fruit that has a Long, Sweet, and Yellow characteristic. What fruit is it?

- To know what fruit is it, we must compute the value of this probability.

1. **Whether that fruit is a Banana P(Banana|Long, Sweet, Yellow)**
2. **Whether that fruit is an Orange P(Orange|Long, Sweet, Yellow)**
3. **Whether that fruit is the Other fruit P(Other|Long, Sweet, Yellow)**

P(Banana|Long, Sweet, Yellow)

$$P(Banana|Long, Sweet, Yellow) = \frac{P(Long, Sweet, Yellow|Banana) * P(Banana)}{P(Long, Sweet, Yellow)}$$

$$P(Banana|Long, Sweet, Yellow) = \frac{P(Long|Banana) * P(Sweet|Banana) * P(Yellow|Banana) * P(Banana)}{P(Long) * P(Sweet) * P(Yellow)}$$

$$P(Banana|Long, Sweet, Yellow) = \frac{0.8 * 0.7 * 0.9 * 0.5}{0.55 * 0.63 * 0.78}$$

$$P(Banana|Long, Sweet, Yellow) = \frac{0.25}{0.27}$$

P(Orange|Long,Sweet,Yellow) = 0

The probability given fruit is Orange are zero because the Probability of Orange when given fruit is long are zero.

. . .

P(Other|Long, Sweet, Yellow)

$$P(Other|Long, Sweet, Yellow) = \frac{P(Long, Sweet, Yellow|Other) * P(Other)}{P(Long, Sweet, Yellow)}$$

$$P(Other|Long, Sweet, Yellow) = \frac{P(Long|Other) * P(Sweet|Other) * P(Yellow|Other) * P(Other)}{P(Long) * P(Sweet) * P(Yellow)}$$

$$P(Other|Long, Sweet, Yellow) = \frac{0.75 * 0.63 * 0.13 * 0.2}{0.55 * 0.63 * 0.78}$$

$$P(Other|Long, Sweet, Yellow) = \frac{0.01}{0.27}$$

```
P(Banana|Long, Sweet, Yellow) > P(Other|Long, Sweet, Yellow)
```

so that the given fruit will be classified into *Banana*.

Advantages

- **Incorporating Prior Knowledge-** Bayes Theorem allows to incorporate prior knowledge or beliefs into the model. This is particularly useful when you have domain expertise or existing information about the problem you are trying to solve. By incorporating prior knowledge, you can make more informed predictions and improve the accuracy of your models.
- **Handling Uncertainty**– Machine learning often deals with uncertainty, especially when working with limited data or noisy inputs. Bayes Theorem provides a principled way to reason under uncertainty by updating beliefs based on observed evidence. It allows you to quantify and propagate uncertainty throughout the model, resulting in more robust and reliable predictions.
- **Flexibility in Model Updating-** Based on Bayes Theorem, Bayesian inference lets you update your models as new data becomes available continuously. This is particularly useful in scenarios where the underlying data distribution may change over time. You can adapt your models and maintain their relevance and accuracy by updating the prior probabilities with new evidence.

- **Handling Small Data Sets-** Bayesian methods can be advantageous when data is insufficient. By incorporating prior beliefs, you can leverage existing knowledge to overcome the limitations of small datasets. The prior probabilities act as a regularization term, helping to avoid overfitting and providing more stable predictions.
- **Transparent Decision Making-** Bayes Theorem provides a transparent framework for decision-making. It allows you to express and update your beliefs based on observed evidence. This transparency is valuable for understanding model behavior, diagnosing issues, and explaining the reasoning behind predictions to stakeholders or end-users.
- **Flexibility in Model Selection-** Bayesian inference facilitates model comparison and selection. By evaluating the posterior probabilities of different models, you can quantify their relative performance and choose the most suitable one. This flexibility allows you to compare complex models and select the one that best fits the data and problem.

Bayesian Belief Network

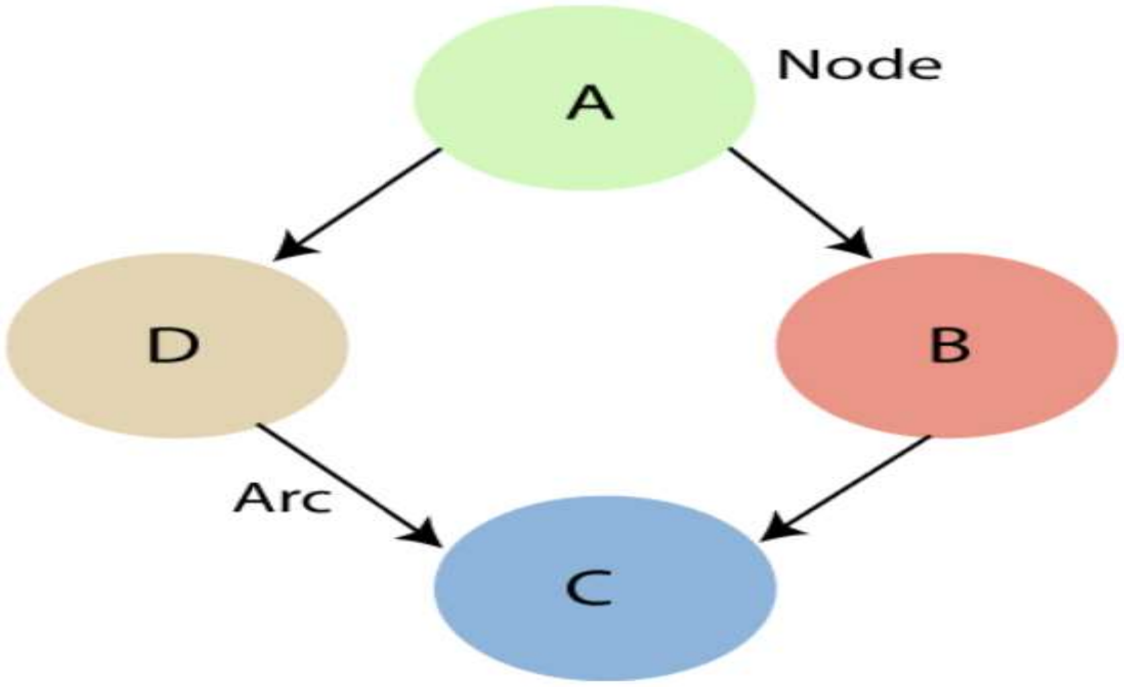
- **Bayesian Belief Network** is a graphical representation of different probabilistic relationships among random variables in a particular set.
- It is a classifier with no dependency on attributes i.e it is condition independent.
- Due to its feature of joint probability, the probability in Bayesian Belief Network is derived, based on a condition — P(attribute/parent) i.e probability of an attribute, true over parent attribute.

- Bayesian Network can be used for building models from data and experts opinions, and it consists of two parts:

  - **Directed Acyclic Graph**
  - **Table of conditional probabilities.**

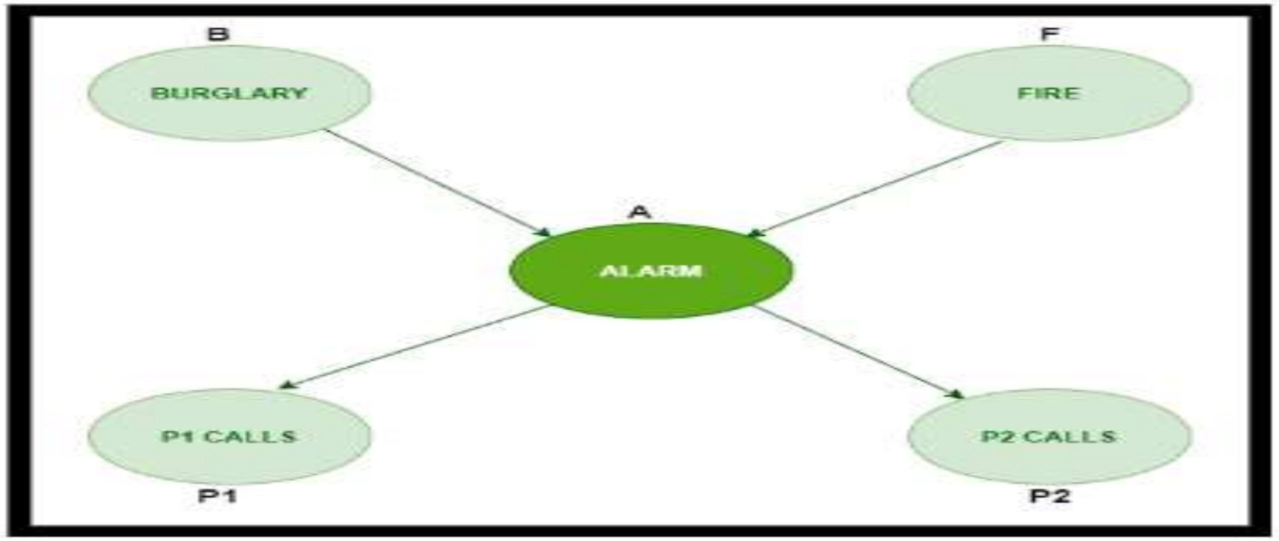
The generalized form of Bayesian network that represents and solve decision problems under uncertain knowledge is known as an **Influence diagram**.

A Bayesian network graph is made up of nodes and Arcs (directed links), where:



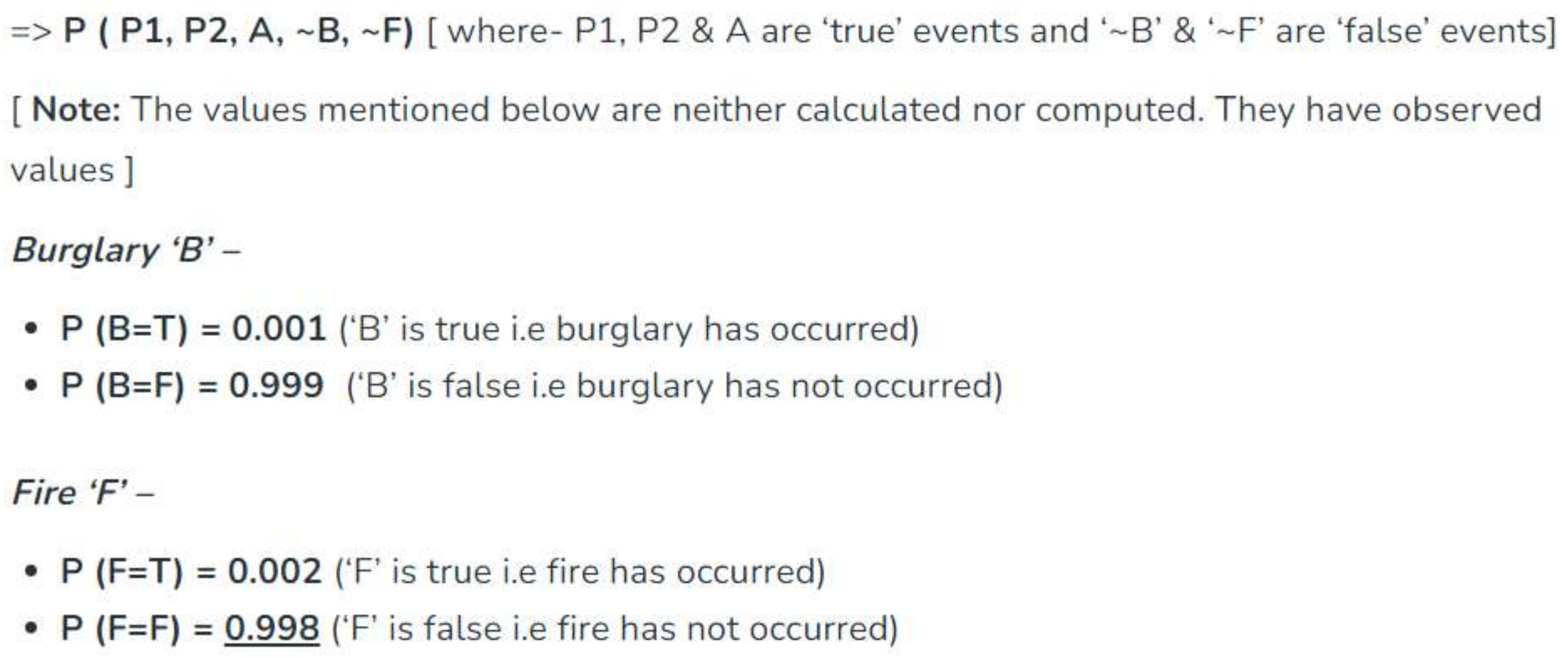
- Each **node** corresponds to the random variables, and a variable can be **continuous** or **discrete**.

- Example: In the given figure, an alarm ‘A’ – a node, say installed in a house of a person ‘gfg’, which rings upon two probabilities i.e burglary ‘B’ and fire ‘F’, which are – parent nodes of the alarm node. The alarm is the parent node of two probabilities P1 calls ‘P1’ & P2 calls ‘P2’ person nodes.
- Upon the instance of burglary and fire, ‘P1’ and ‘P2’ call person ‘gfg’, respectively. But, sometimes ‘P1’ may forget to call even after hearing the alarm, Similarly, ‘P2’, sometimes fails to call the person.



Q) Find the probability that ‘P1’ is true (P1 has called ‘gfg’), ‘P2’ is true (P2 has called ‘gfg’) when the alarm ‘A’ rang, but no burglary ‘B’ and fire ‘F’ has occurred.

Given:



=> **P (P1, P2, A, ~B, ~F)** [where- P1, P2 & A are ‘true’ events and ‘~B’ & ‘~F’ are ‘false’ events]

[**Note:** The values mentioned below are neither calculated nor computed. They have observed values]

Burglary ‘B’ –

- **P (B=T) = 0.001** (‘B’ is true i.e burglary has occurred)
- **P (B=F) = 0.999** (‘B’ is false i.e burglary has not occurred)

Fire ‘F’ –

- **P (F=T) = 0.002** (‘F’ is true i.e fire has occurred)
- **P (F=F) = 0.998** (‘F’ is false i.e fire has not occurred)

Alarm ‘A’ –

B	F	P (A=T)	P (A=F)
T	T	0.95	0.05
T	F	0.94	0.06
F	T	0.29	0.71
F	F	0.001	**0.999**

- The alarm ‘A’ node can be ‘true’ or ‘false’ (i.e may have rung or may not have rung). It has two parent nodes burglary ‘B’ and fire ‘F’ which can be ‘true’ or ‘false’ (i.e may have occurred or may not have occurred) depending upon different conditions.

Person 'P1' *–*

A	P (P1=T)	P (P1=F)
T	**0.95**	0.05
F	0.05	0.95

- The person 'P1' node can be 'true' or 'false' (i.e may have called the person 'gfg' or not) . It has a parent node, the alarm 'A', which can be 'true' or 'false' (i.e may have rung or may not have rung ,upon burglary 'B' or fire 'F').

Person ‘P2’ –

A	P (P2=T)	P (P2=F)
T	**0.80**	0.20
F	0.01	0.99

- The person ‘P2’ node can be ‘true’ or false’ (i.e may have called the person ‘gfg’ or not). It has a parent node, the alarm ‘A’, which can be ‘true’ or ‘false’ (i.e may have rung or may not have rung, upon burglary ‘B’ or fire ‘F’).

Solution: Considering the observed probabilistic scan –

With respect to the question — **P (P1, P2, A, ~B, ~F)** , we need to get the probability of 'P1'. We find it with regard to its parent node – alarm 'A'. To get the probability of 'P2', we find it with regard to its parent node — alarm 'A'.

We find the probability of alarm 'A' node with regard to '~B' & '~F' since burglary 'B' and fire 'F' are parent nodes of alarm 'A'.

From the observed probabilistic scan, we can deduce –

P (P1, P2, A, ~B, ~F)

= P (P1/A) * P (P2/A) * P (A/~B~F) * P (~B) * P (~F)

= 0.95 * 0.80 * 0.001 * 0.999 * 0.998

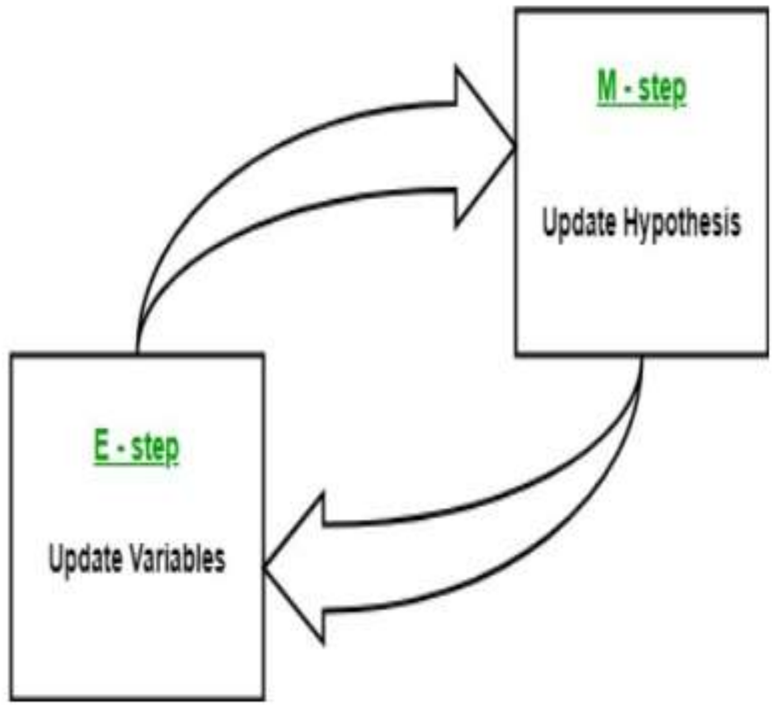
= 0.00075

Expectation-Maximization Algorithm (EM algorithm)

- In real-world machine learning applications, it is common to have many relevant features, but only a subset of them may be observable.
- When dealing with variables that are sometimes observable and sometimes not, it is indeed possible to utilize the instances when that variable is visible or observed in order to learn and make predictions for the instances where it is not observable.
- This approach is often referred to as handling missing data. By using the available instances where the variable is observable, machine learning algorithms can learn patterns and relationships from the observed data.

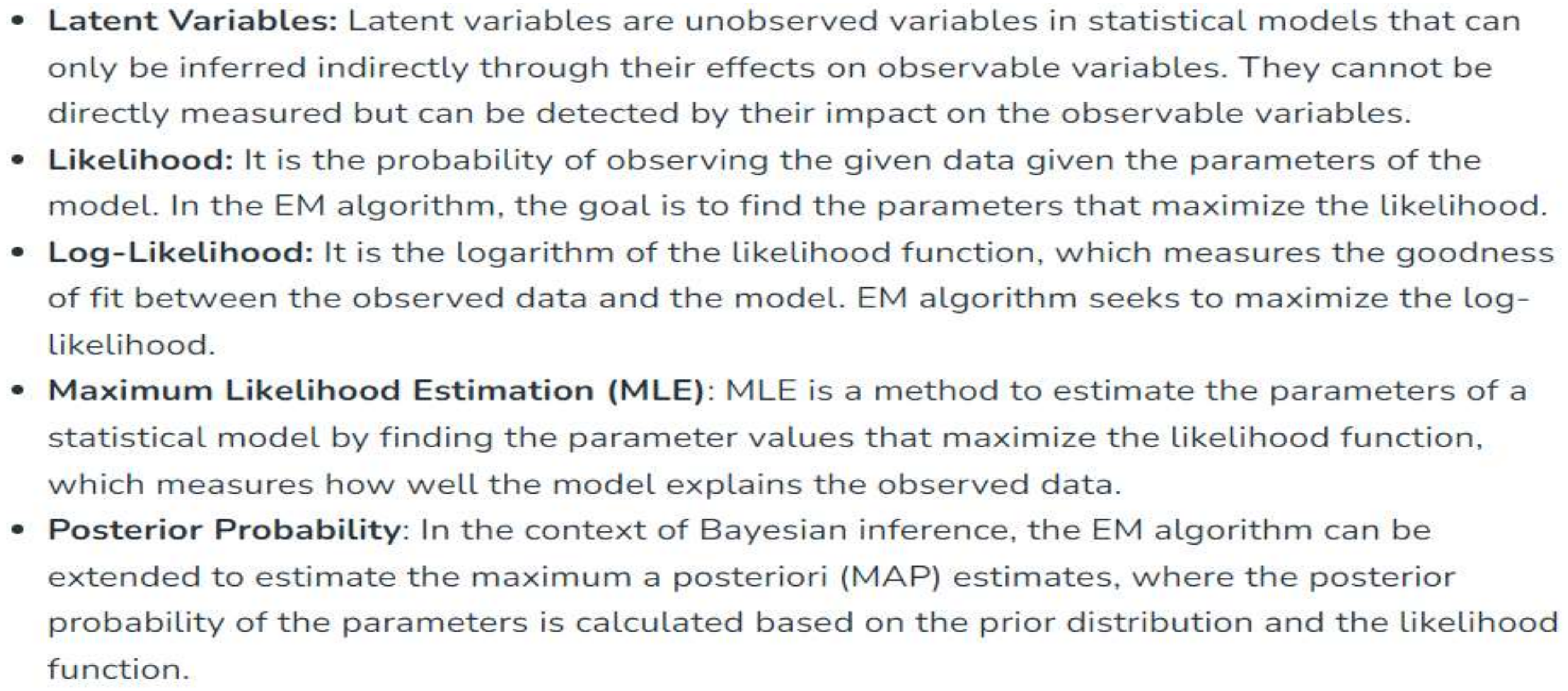
- These learned patterns can then be used to predict the values of the variable in instances where it is missing or not observable.
- The expectation-Maximization algorithm can be used to handle situations where variables are partially observable. When certain variables are observable, we can use those instances to learn and estimate their values. Then, we can predict the values of these variables in instances when it is not observable.
- EM algorithm is applicable to latent variables, which are variables that are not directly observable but are inferred from the values of other observed variables.
- **The EM algorithm serves as the foundation for many unsupervised clustering algorithms in the field of machine learning.** It provides a framework to find the local maximum likelihood parameters of a statistical model and infer latent variables in cases where data is missing or incomplete.

- It consists of an estimation step (E-step) and a maximization step (M-step), forming an iterative process to improve model fit.
- In the E step, the algorithm computes the latent variables i.e. expectation of the log-likelihood using the current parameter estimates.
- In the M step, the algorithm determines the parameters that maximize the expected log-likelihood obtained in the E step, and corresponding model parameters are updated based on the estimated latent variables.
- By iteratively repeating these steps, the EM algorithm seeks to maximize the likelihood of the observed data.
- It is commonly used for unsupervised learning tasks, such as clustering, where latent variables are inferred and has applications in various fields, including machine learning, computer vision, and natural language processing.



Expectation-Maximization in EM Algorithm

Key Terms in Expectation-Maximization (EM) Algorithm

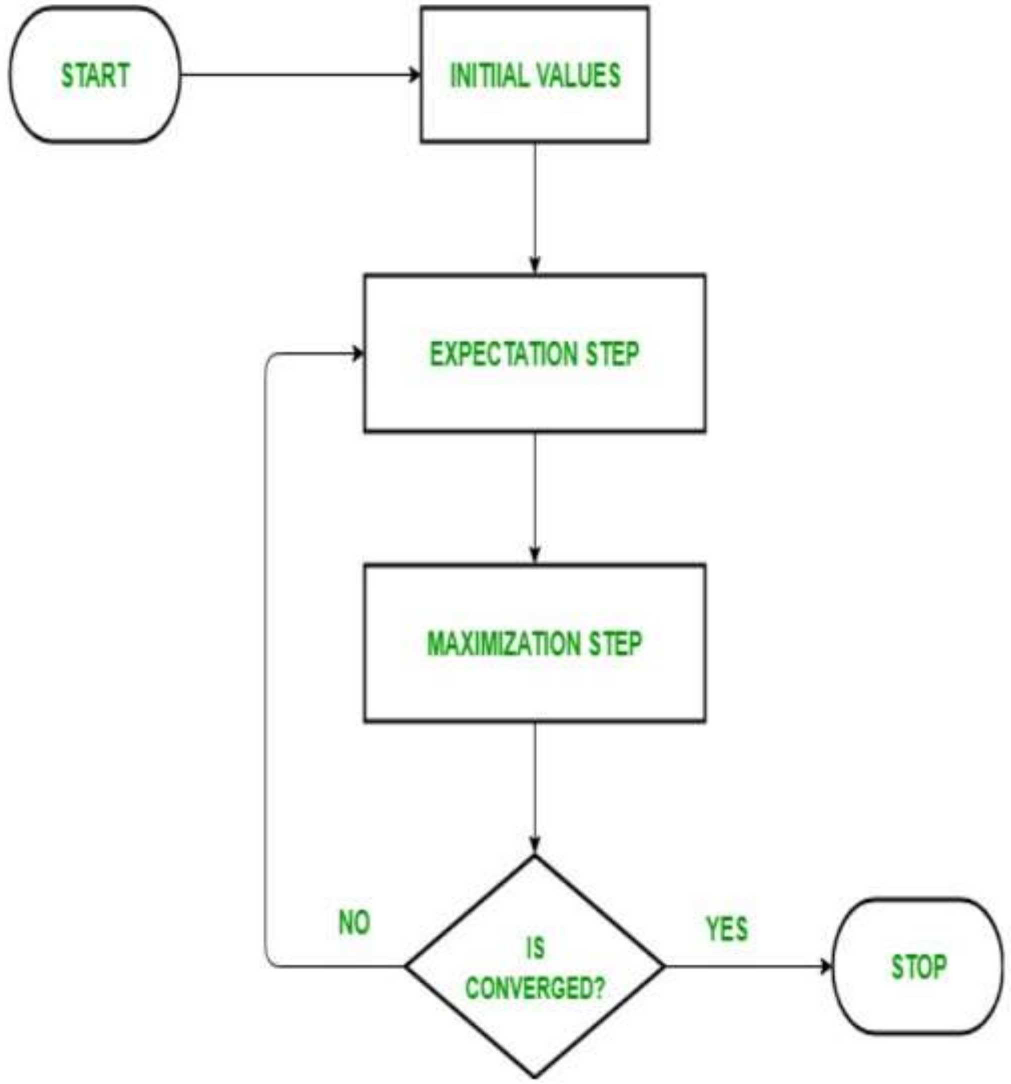


- **Latent Variables:** Latent variables are unobserved variables in statistical models that can only be inferred indirectly through their effects on observable variables. They cannot be directly measured but can be detected by their impact on the observable variables.
- **Likelihood:** It is the probability of observing the given data given the parameters of the model. In the EM algorithm, the goal is to find the parameters that maximize the likelihood.
- **Log-Likelihood:** It is the logarithm of the likelihood function, which measures the goodness of fit between the observed data and the model. EM algorithm seeks to maximize the log-likelihood.
- **Maximum Likelihood Estimation (MLE)**: MLE is a method to estimate the parameters of a statistical model by finding the parameter values that maximize the likelihood function, which measures how well the model explains the observed data.
- **Posterior Probability**: In the context of Bayesian inference, the EM algorithm can be extended to estimate the maximum a posteriori (MAP) estimates, where the posterior probability of the parameters is calculated based on the prior distribution and the likelihood function.

- **Expectation (E) Step**: The E-step of the EM algorithm computes the expected value or posterior probability of the latent variables given the observed data and current parameter estimates. It involves calculating the probabilities of each latent variable for each data point.
- **Maximization (M) Step**: The M-step of the EM algorithm updates the parameter estimates by maximizing the expected log-likelihood obtained from the E-step. It involves finding the parameter values that optimize the likelihood function, typically through numerical optimization methods.
- **Convergence:** Convergence refers to the condition when the EM algorithm has reached a stable solution. It is typically determined by checking if the change in the log-likelihood or the parameter estimates falls below a predefined threshold.

How Expectation-Maximization (EM) Algorithm Works:

- The goal of the Expectation-Maximization algorithm is to use the available observed data of the dataset to estimate the missing data and then use that data to update the values of the parameters.



1. **Initialization:**
    - Initially, a set of initial values of the parameters are considered. A set of incomplete observed data is given to the system with the assumption that the observed data comes from a specific model.

2. **E-Step (Expectation Step):** In this step, we use the observed data in order to estimate or guess the values of the missing or incomplete data. It is basically used to update the variables.
    - Compute the posterior probability or responsibility of each latent variable given the observed data and current parameter estimates.
    - Estimate the missing or incomplete data values using the current parameter estimates.
    - Compute the log-likelihood of the observed data based on the current parameter estimates and estimated missing data.

3. **M-step (Maximization Step):** In this step, we use the complete data generated in the preceding “Expectation” – step in order to update the values of the parameters. It is basically used to update the hypothesis.
    - Update the parameters of the model by maximizing the expected complete data log-likelihood obtained from the E-step.
    - This typically involves solving optimization problems to find the parameter values that maximize the log-likelihood.
    - The specific optimization technique used depends on the nature of the problem and the model being used.

4. **Convergence**: In this step, it is checked whether the values are converging or not, if yes, then stop otherwise repeat *step-2* and *step-3* i.e. “Expectation” – step and “Maximization” – step until the convergence occurs.
    - Check for convergence by comparing the change in log-likelihood or the parameter values between iterations.
    - If the change is below a predefined threshold, stop and consider the algorithm converged.
    - Otherwise, go back to the E-step and repeat the process until convergence is achieved.