**Important questions for mid-II**

**Long answer questions**

1. Explain Naïve Bayes classifier

Ans:

* 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 mainly used in *text classification* that includes a high-dimensional training dataset.
* It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

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 bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.
* **Bayes**: It is called Bayes because it depends on the principle of [Bayes' Theorem](https://www.javatpoint.com/bayes-theorem-in-artifical-intelligence).

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:

Naïve Bayes Classifier Algorithm

**Where,**

**P(A|B) is Posterior probability**: Probability of hypothesis A on the observed event B.

**P(B|A) is Likelihood probability**: Probability of the evidence given that the probability of a hypothesis is true.

**P(A) is Prior Probability**: Probability of hypothesis before observing the evidence.

**P(B) is Marginal Probability**: Probability of Evidence.

|  |  |  |
| --- | --- | --- |
|  | **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 |

## Working of Naïve Bayes' Classifier:

Working of Naïve Bayes' Classifier can be understood with the help of the below example:

Suppose we have a dataset of **weather conditions** and corresponding target variable "**Play**". So, using this dataset we need to decide that 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 below steps:

1. Convert the given dataset into frequency tables.
2. Generate Likelihood table by finding the probabilities of given features.
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, first consider the below dataset:

**Frequency table for the Weather Conditions:**

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

**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 |  |

**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.**

### **Advantages of Naïve Bayes Classifier:**

* Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
* 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 of Naïve Bayes Classifier:**

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

1. **Explain K-NN with an example**

Ans:

# **K-Nearest Neighbour (KNN) Algorithm for Machine Learning**

* K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
* K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
* K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
* K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
* 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, then it classifies that data into a category that is much similar to the new data.
* **(Don’t write in assignment)(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 dog’s images and based on the most similar features it will put it in either cat or dog category.)

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

* **Step-1:** Select the K number of the neighbours.
* **Step-2:** Calculate the Euclidean distance of **K number of neighbours.**
* **Step-3:** Take the K nearest neighbours as per the calculated Euclidean distance.
* **Step-4:** Among these K neighbours, 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 neighbour is maximum.
* **Step-6:** Our model is ready.

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



* Firstly, we will choose the number of neighbours, so we will choose the k=5.
* Next, we will calculate the **Euclidean distance** between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:



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



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

How to select the value of K in the K-NN Algorithm?

Below are some points to remember while selecting 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 the 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 some time.
* The computation cost is high because of calculating the distance between the data points for all the training samples.

1. **Explain sequential covering algorithm**

Ans:

Sequential Covering is a popular algorithm based on Rule-Based Classification used for learning a disjunctive set of rules. The basic idea here is to learn one rule, remove the data that it covers, then repeat the same process. In this process, in this way, it covers all the rules involved with it in a sequential manner during the training phase.

**Algorithm Involved:**

**Sequential\_covering (Target\_attribute, Attributes, Examples, Threshold):**

Learned\_rules = {}

Rule = Learn-One-Rule(Target\_attribute, Attributes, Examples)

while Performance(Rule, Examples) > Threshold :

Learned\_rules = Learned\_rules + Rule

Examples = Examples - {examples correctly classified by Rule}

Rule = Learn-One-Rule(Target\_attribute, Attributes, Examples)

Learned\_rules = sort Learned\_rules according to performance over Examples

return Learned\_rules

The Sequential Learning algorithm takes care of to some extent, the low coverage problem in the Learn-One-Rule algorithm covering all the rules in a sequential manner.

**Working on the Algorithm:**

The algorithm involves a set of ‘ordered rules’ or ‘list of decisions’ to be made.

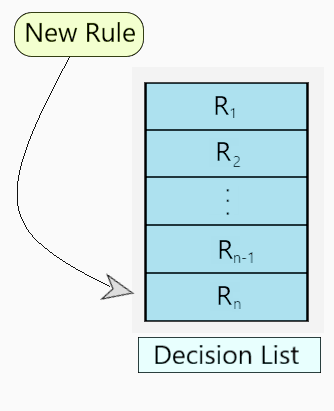
***Step 1 –****create an empty decision list, ‘R’.****Step 2 – ‘****Learn-One-Rule’ Algorithm: It extracts the best rule for a particular class ‘y’.*

**General Form of Rule: ri : (condition1, … conditioni ) -> yi**

*In the beginning,****Step 2.a –****if all training examples ∈ class ‘y’, then it’s classified as****positive example****.****Step 2.b –****else if all training examples ∉ class ‘y’, then it’s classified as****negative example****.*

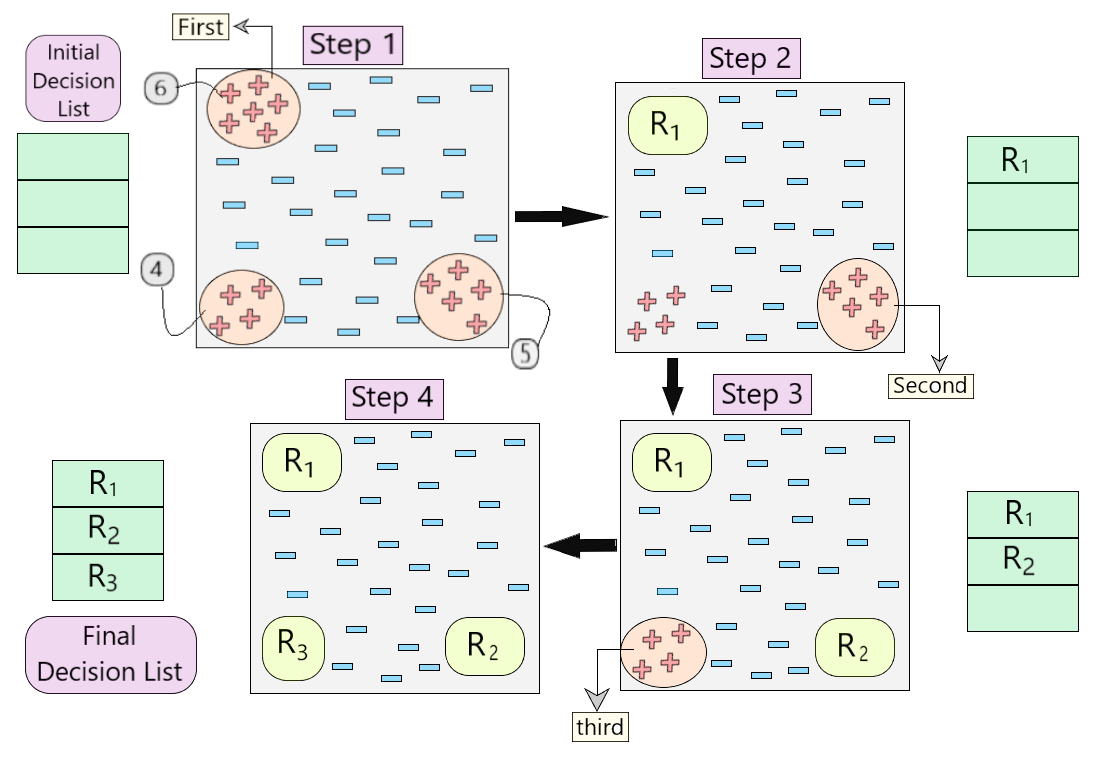
***Step 3 –****The rule becomes****‘desirable’****when it covers a majority of the positive examples.****Step 4 –****When this rule is obtained, delete all the training data associated with that rule.  
(i.e., when the rule is applied to the dataset, it covers most of the training data, and has to be removed)*

***Step 5 –****The new rule is added to the bottom of decision list, ‘R’. (Fig.3)*



***Fig 3: Decision List ‘R’***

Below, is a visual representation describing the working of the algorithm.



***Fig 4: Visual Representation of working of the algorithm***

* Let us understand step by step how the algorithm is working in the example shown in Fig.4.
* First, we created an empty decision list. During Step 1, we see that there are three sets of positive examples present in the dataset. So, as per the algorithm, we consider the one with maximum no of positive example. *(6, as shown in Step 1 of Fig 4)*
* Once we cover these 6 positive examples, we get our first rule R1, which is then pushed into the decision list and those positive examples are removed from the dataset. *(as shown in Step 2 of Fig 4)*
* Now, we take the next majority of positive examples *(5, as shown in Step 2 of Fig 4)*and follow the same process until we get rule R2. (Same for R3)
* In the end, we obtain our final decision list with all the desirable rules.

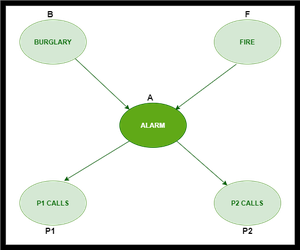
1. **Explain Bayesian belief networks.**

Basic Understanding of 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.

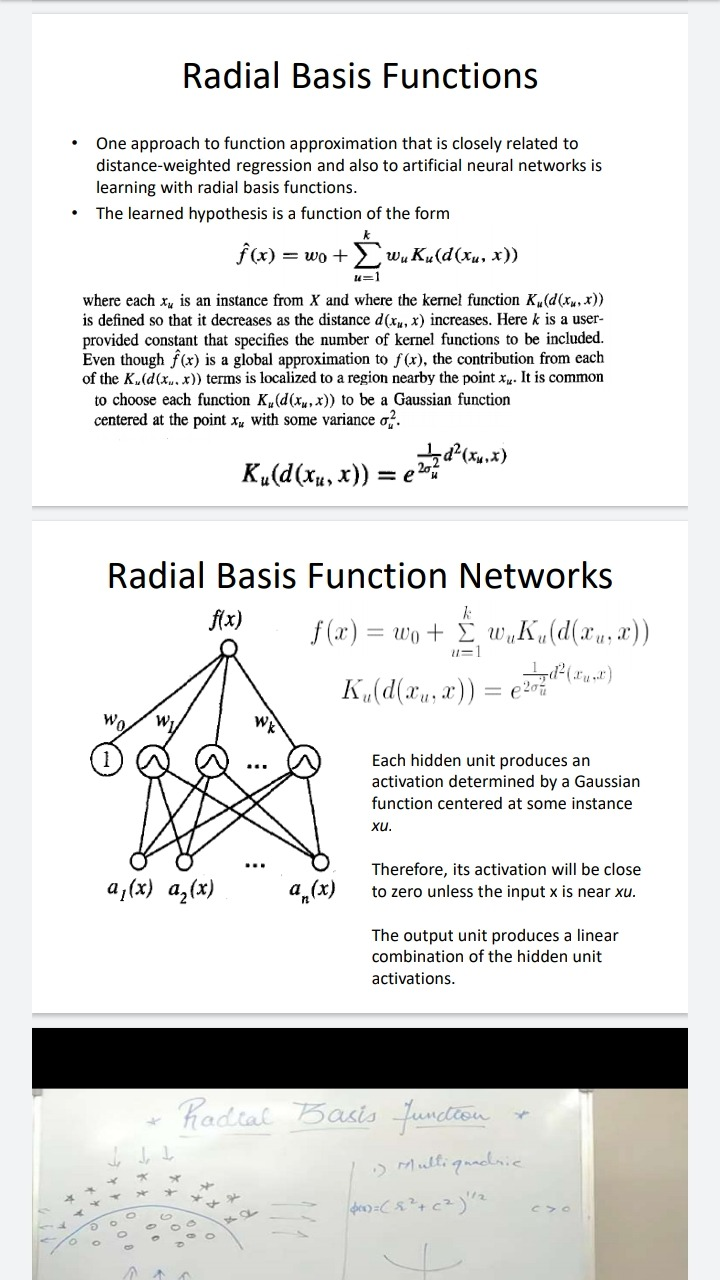
(Note: A classifier assigns data in a collection to desired categories.)

* Consider this example:



* In the above figure, we have 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, there are few drawbacks in this case, as sometimes ‘P1’ may forget to call the person ‘gfg’, even after hearing the alarm, as he has a tendency to forget things, quick.  Similarly, ‘P2’, sometimes fails to call the person ‘gfg’, as he is only able to hear the alarm, from a certain distance.

1. **Analyse the usage of Radial basis functions.**



1. **Explain FOIL algorithm**

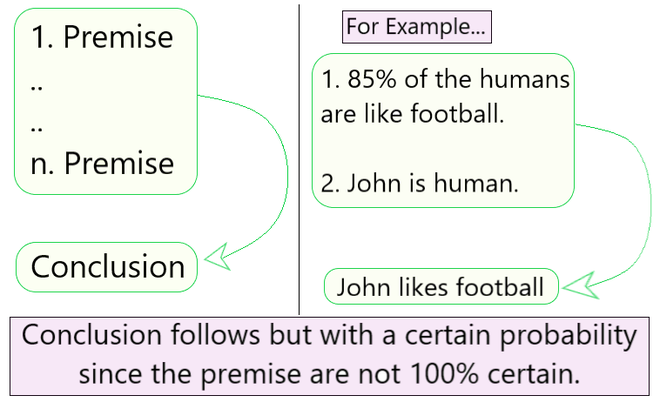
Ans:

**First Order Inductive Learner (FOIL)**

In machine learning, first-order inductive learner (FOIL) is a rule-based learning algorithm. It is a natural extension of SEQUENTIAL-COVERING and LEARN-ONE-RULE algorithms. It follows a Greedy approach.

**Inductive Learning:**

Inductive learning analyzing and understanding the evidence and then using it to determine the outcome. It is based on Inductive Logic.



***Fig 1: Inductive Logic***

**Algorithm Involved**

**FOIL (Target predicate, predicates, examples)**

• Pos ← positive examples

• Neg ← negative examples

• Learned rules ← {}

• while Pos, do

**//Learn a NewRule**

**–** NewRule ← the rule that predicts target-predicate with no preconditions

**–** NewRuleNeg ← Neg

**–** while NewRuleNeg, do

Add a new literal to specialize NewRule

1. Candidate\_literals ← generate candidates for newRule based on Predicates

2. Best\_literal ←

**argmaxL∈Candidate literalsFoil\_Gain(L,NewRule)**

3. add Best\_literal to NewRule preconditions

4. NewRuleNeg ← subset of NewRuleNeg that satisfies NewRule preconditions

**–** Learned rules ← Learned rules + NewRule

**–** Pos ← Pos − {members of Pos covered by NewRule}

• Return Learned rules

**Working of the Algorithm:**

In the algorithm, the inner loop is used to generate a new best rule. Let us consider an example and understand the step-by-step working of the algorithm.

Say we are trying to predict the **Target-predicate-** *GrandDaughter(x,y)*.

We perform the following steps: **[Refer Fig 2]**

**Step 1 -** **NewRule = GrandDaughter(x,y)**

**Step 2 -**

**2.a -** Generate the candidate\_literals.

*(Female(x), Female(y), Father(x,y), Father(y.x),*

*Father(x,z), Father(z,x), Father(y,z), Father(z,y)*)

**2.b -** Generate the respective candidate literal negations.

*(¬Female(x), ¬Female(y), ¬Father(x,y), ¬Father(y.x),*

*¬Father(x,z), ¬Father(z,x), ¬Father(y,z), ¬Father(z,y))*

**Step 3 -** FOIL might greedily select Father(x,y) as most promising, then

**NewRule = GrandDaughter(x,y) ← Father(y,z)** **[Greedy approach]**

**Step 4 -** Foil now considers all the literals from the previous step as well as:

*(Female(z), Father(z,w), Father(w,z), etc.)* and their negations.

**Step 5 -** Foil might select Father(z,x), and on the next step Female(y) leading to

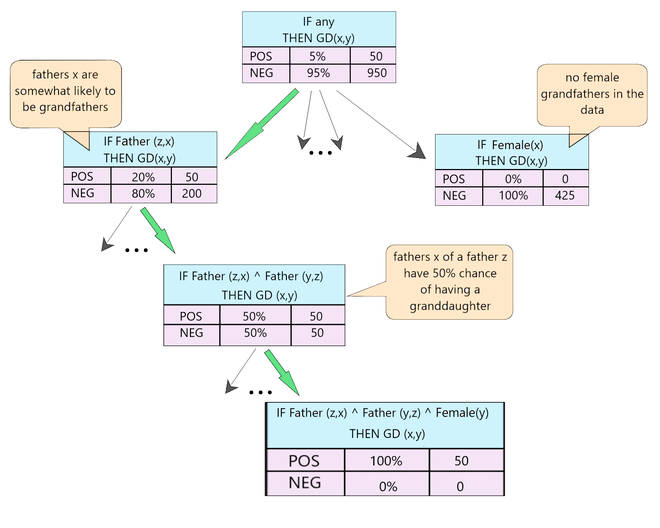
**NewRule = GrandDaughter (x,y) ← Father(y,z) ∧ Father(z,x) ∧ Female(y)**

**Step 6 -** If this greedy approach covers only positive examples it terminates

the search for further better results.

FOIL now **removes all positive examples** **covered by this new rule.**

If more are left then the outer while loop continues.



***Fig 2: FOIL Example***

**FOIL: Performance Evaluation Measure**

The performance of a new rule is not defined by its entropy measure (like the *PERFORMANCE* method in Learn-One-Rule algorithm).

FOIL uses a gain algorithm to determine which new specialized rule to opt. Each rule’s utility is estimated by the number of bits required to encode all the positive bindings. **[Eq.1]**

where,

**L is the candidate literal to add to rule R**

**p0 =** number of positive bindings of R

**n0 =** number of negative bindings of R

**p1 =** number of positive binding of R + L

**n1 =** number of negative bindings of R + L

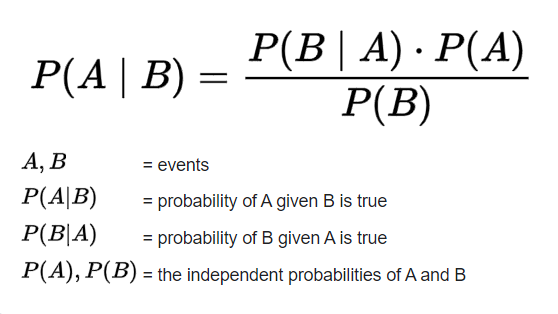
**t =** number of positive bindings of R also covered by R + L

**Short answer questions**

1. What is Bayes theorem?

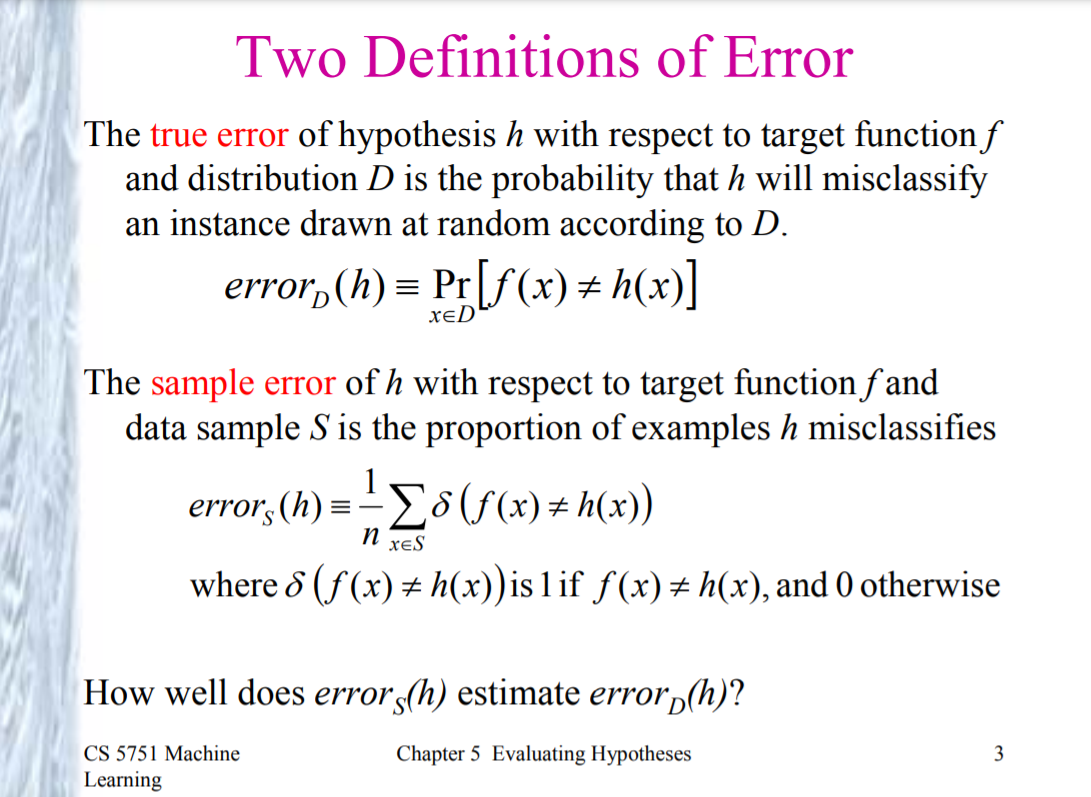
Ans:

It is a mathematical formula for determining [conditional probability](https://www.investopedia.com/terms/c/conditional_probability.asp). Conditional probability is the likelihood of an outcome occurring, based on a previous outcome occurring. Bayes' theorem provides a way to revise existing predictions or theories (update probabilities) given new or additional evidence.



1. Define “the true error of hypothesis”.

Ans:



1. Explain Gaussian function in few words.

Ans:

Gaussian functions are widely used in [statistics](https://en.wikipedia.org/wiki/Statistics) to describe the [normal distributions](https://en.wikipedia.org/wiki/Normal_distribution), in [signal processing](https://en.wikipedia.org/wiki/Signal_processing) to define [Gaussian filters](https://en.wikipedia.org/wiki/Gaussian_filter), in [image processing](https://en.wikipedia.org/wiki/Image_processing) where two-dimensional Gaussians are used for [Gaussian blurs](https://en.wikipedia.org/wiki/Gaussian_blur), and in mathematics to solve [heat equations](https://en.wikipedia.org/wiki/Heat_equation) and [diffusion equations](https://en.wikipedia.org/wiki/Diffusion_equation) and to define the [Weierstrass transform](https://en.wikipedia.org/wiki/Weierstrass_transform" \o "Weierstrass transform).

1. Interpret “instance-based learning”.

Ans:

**Instance-based learning** (sometimes called memory-based learning) is a family of learning algorithms that, instead of performing explicit generalization, compares new problem instances with instances seen in training, which have been stored in memory, and actions according to the previous analysis.

1. What is Horn clause in first order logic?

Ans:

A [clause](https://mathworld.wolfram.com/Clause.html) (i.e., a [disjunction](https://mathworld.wolfram.com/Disjunction.html) of [literals](https://mathworld.wolfram.com/Literal.html)) is called a Horn clause if it contains at most one [positive literal](https://mathworld.wolfram.com/PositiveLiteral.html). Horn clauses are usually written as

|  |
| --- |
| L_1,...,L_n=>L(=¬L_1 v ... v ¬L_n v L) |

or

|  |
| --- |
| L_1,...,L_n=>(=¬L_1 v ... v ¬L_n), |

where n>=0 and L is the only [positive literal](https://mathworld.wolfram.com/PositiveLiteral.html).

A [definite clause](https://mathworld.wolfram.com/DefiniteClause.html) is a Horn clause that has [exactly one](https://mathworld.wolfram.com/ExactlyOne.html) [positive literal](https://mathworld.wolfram.com/PositiveLiteral.html). A Horn clause without a [positive literal](https://mathworld.wolfram.com/PositiveLiteral.html) is called a [goal](https://mathworld.wolfram.com/Goal.html).

Horn clauses express a subset of statements of [first-order logic](https://mathworld.wolfram.com/First-OrderLogic.html). Programming language Prolog is built on top of Horn clauses. Prolog programs are comprised of [definite clauses](https://mathworld.wolfram.com/DefiniteClause.html) and any question in Prolog is a [goal](https://mathworld.wolfram.com/Goal.html).

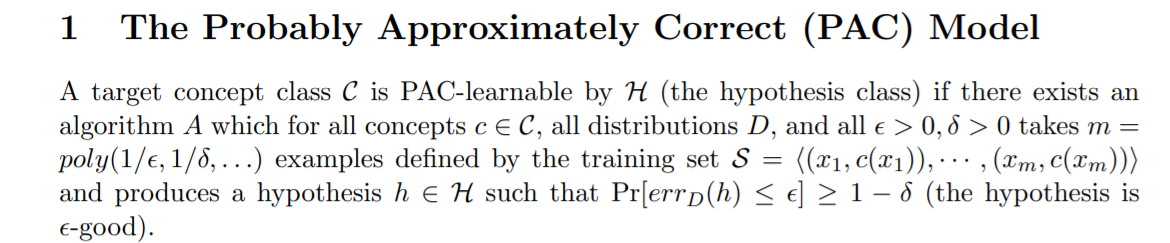
1. Define maximum likelihood hypothesis.

Ans:

The technique of **maximum likelihood** (ML) is a method to: (1) estimate the parameters of a model; and (2) test **hypotheses** about those parameters. ... The model that describes the distribution of the variables will have certain unknown quantities in it. These are called parameters.

1. When a concept C is said to be PAC-learnable?

Ans:



1. Interpret “curse of dimensionality” in K-NN
2. Explain case-based reasoning in few words.

Ans:

In **case-based reasoning**, the training examples, the **cases**, are stored and accessed to solve a new problem. To get a prediction for a new example, those cases that are similar, or close to, the new example are used to predict the value of the target features of the new example.

1. What is substitution in first order logic?

Ans:

In **first**-**order logic**, every closed propositional formula that can be derived from an open propositional formula A by **substitution** is said to be a **substitution** instance of A. If A is a closed propositional formula, we count A itself as its only **substitution** instance.

1. Define least squared error hypothesis.
2. Define Minimum description length principle.

Ans:

The **minimum description length** (**MDL**) **principle** is a powerful method of inductive inference, the basis of statistical modelling, pattern recognition, and **machine learning**. It holds that the best explanation, given a limited set of observed data, is the one that permits the greatest compression of the data.

1. List radial basis functions.

Ans:

* linear radial basis function *ϕ*(*r*)=*r*, so long as *m*>1,
* (Hardy) multiquadrics radial basis function *ϕ*(*r*)= √(*r*2+*c*2), which contains another scalar parameter *c* which may be adjusted to improve the approximation, where the choice *c*=0 gives the previous example,
* Gaussian kernel *ϕ*(*r*)=exp(-c2r2), which also contains another scalar parameter *c*≠0 which may be adjusted to adapt the approximation, or finally
* (Hardy) inverse multiquadrics radial basis function *ϕ*(*r*)=1/√(*r*2+*c*2), which contains another scalar parameter *c*≠0 which provides further flexibility.

1. What is induction?

Ans:

**Induction** is the process of inferring general rules from specific data and is the primary task of **machine learning**. ... We treat **induction** and abduction as two distinct reasoning tasks, but have demonstrated that each can be of direct service to the other in developing AI systems for solving real-world problems.

1. What is term in first order logic?

Ans:

**First**-**order logic** is symbolized reasoning in which each sentence, or statement, is broken down into a subject and a predicate. The predicate modifies or defines the properties of the subject. ... **First**-**order logic** is also known as **first**-**order** predicate calculus or **first**-**order** functional calculus.