MODULE - 3.

Classification vs Prediction.

Classification:

→ predicts categorical class labels (discrete or nominal)

→ classifies data (constructs a model) based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data.

Prediction

→ models continuous-valued functions i.e predicts unknown or missing values.

Classification - A two-step-process

1) Model Construction: describing a set of predetermined classes

* Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute.

* The set of tuples used for model construction is the training set.

* The model is represented as classification rules, decision trees or mathematical formulae.

2) Model Usage: for classifying future or unknown objects.

* Estimate accuracy of the model.

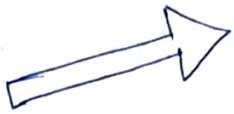
→ The known label of a test sample is compared with the classified result from the model.

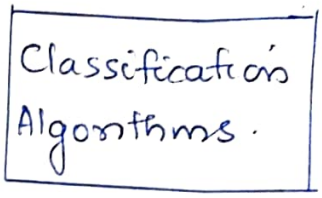
→ Accuracy rate is the percentage of test set samples that are correctly classified by the model.

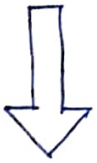
→ Test set is completely independent of training set, otherwise overfitting will occur.

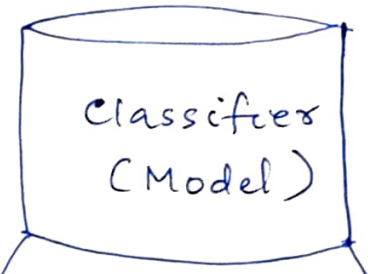
* If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known.

eg: Process (1) : Model Construction.





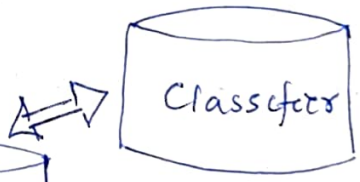




Name	Rank.	Years	Tenured.
Mike	AP	3	no
Mary	AP	7	yes.
Bill	Prof	2	yes.
Jim	Asso. Prof	.7	yes
Dave	A P	6	no.
Anne	Asso. Prof	3	no.

IF rank = 'professor'
OR years > 6
THEN Tenured = 'yes'.

Process(2): Using the Model for Prediction.



Name	Rank	Years	Tenured
Tom	AP	2	no
Merlisa	Asso. Prof	7	no
George	Prof	5	yes
Joseph	AP	7	yes.

Jeff, Professor, 4

Tenured ?

Yes.

Supervised vs Unsupervised Learning.

→ Supervised Learning (classification)

Supervision: The training data (observations, measurements etc) are accompanied by labels indicating the class of observations.

New data is classified based on the training set.

→ Unsupervised Learning: (Clustering)

The class labels of training data are not known

Given a set of observations, measurements etc with the aim of establishing the existence of classes or clusters in the data.

Issues Regarding classification and Prediction:

a) Preparing Data for classification and Prediction.

(i) Data cleaning: Preprocess the data. inorder to reduce noise. and handle missing values.

ii) relevance analysis: Remove redundant and irrelevant attributes.

eg: Correlation analysis can be used to ~~find~~ identify whether 2 attributes are statistically related. Attribute Subset selection can be used to find reduced set of attributes.

iii) data transformation and reduction.

- normalize data so that it lies within the range -1.0 to 1.0 or 0.0 to 1.0.
- data can be transformed by generalizing to higher level concepts.

eg: numeric values for attribute income can be generealized to discrete ranges. such as low, medium and high.

b) Comparing Classification and Prediction Models.

(i) Accuracy:

accuracy of a classifier refers to the ability of a given classifier to correctly predict class label of new or previously unseen data.

accuracy of a predictor: refers to how well a predictor can guess the value of the predicted attribute for new or previously unseen data

ii) Speed.

- the computational costs involved in generating and using the classifier or predictor.

iii) Robustness

- ability of the classifier or predictor to make correct predictions given noisy data or data with missing values.

iv) Scalability.

- ability to construct the classifier or predictor efficiently given large amounts of data

v) Interpretability.

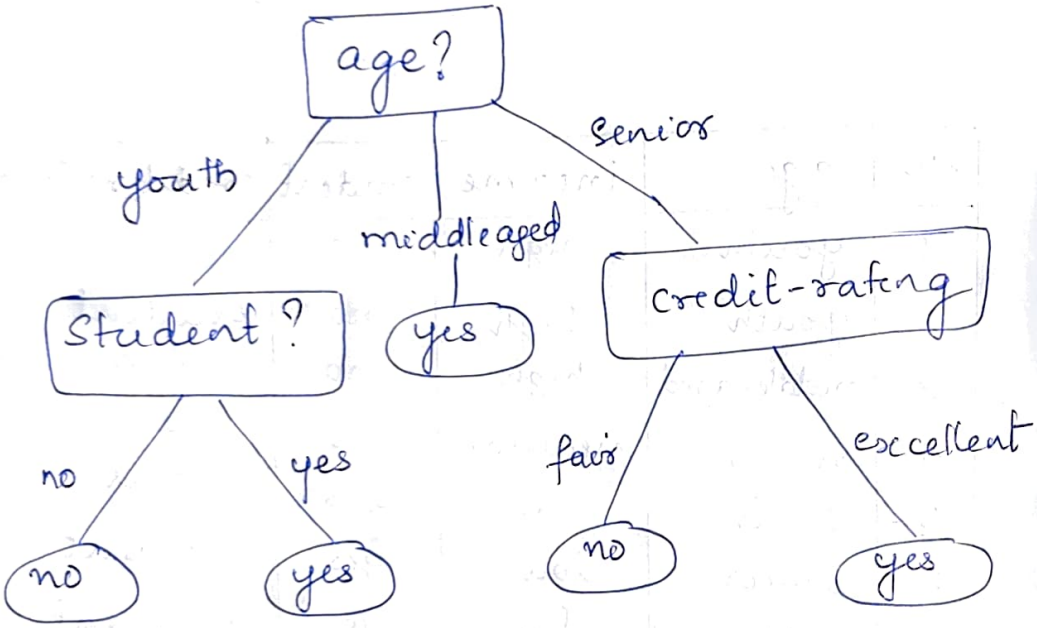
- level of understanding & insight provided by the model.

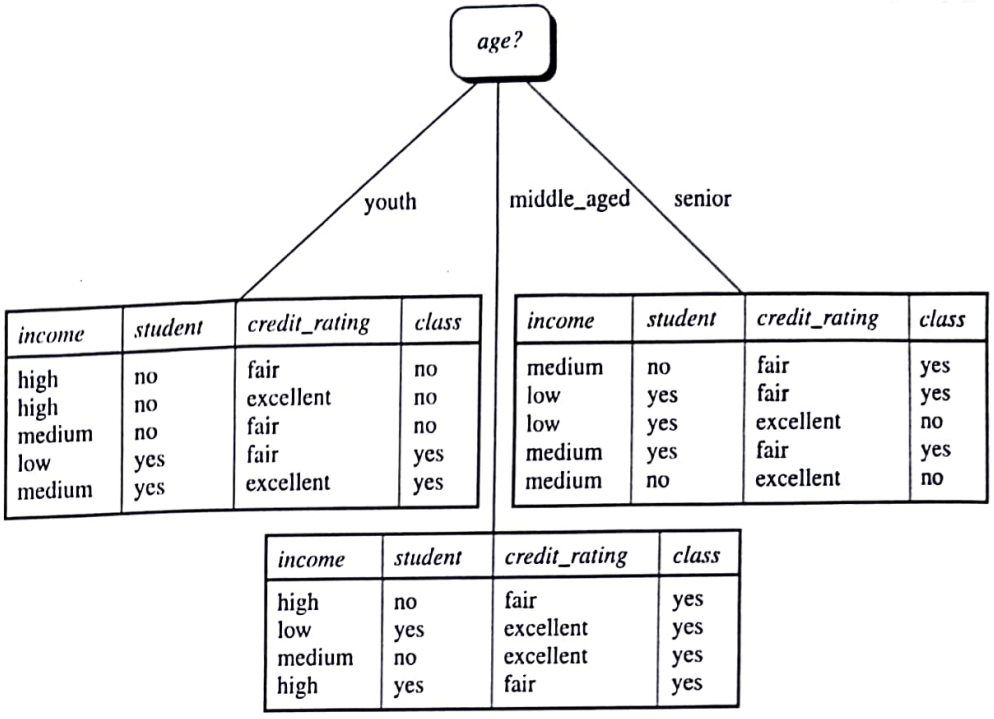
Decision Tree Induction.

- Decision Tree Induction is the learning of decision trees from class labelled training tuples
- A decision tree is a flowchart like structure
  - → where each internal node denotes a test on an attribute
  - → each branch represents an outcome of the test
  - → each leaf node holds a class label.

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle-aged	high	no	fair	yes.
4	senior	medium	no	fair	yes.
5	senior	low	yes	fair	yes.
6	senior	low	yes	excellent	no
7	middle-aged	low	yes	excellent	yes.
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes.
11	youth	medium	yes	excellent	yes.
12	middleaged	medium	no	excellent	yes.
13	middle-aged	high	yes	fair	yes.
14	Senior	medium	no	excellent	no.

A decision tree for the concept buys_computer, indicating whether a customer at AllElectronics is likely to purchase a computer. Each internal (non leaf node) represents a test on an attribute. Each leaf node represents a class (either buys_computer = yes or buys_computer = no)





income	student	credit_rating	class
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	student	credit_rating	class
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	student	credit_rating	class
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

The attribute *age* has the highest information gain and therefore becomes the splitting attribute at the root node of the decision tree. Branches are grown for each outcome of *age*. The tuples are shown partitioned accordingly.

Decision Tree – ID3.

- ID3 is a basic algorithm for learning Decision trees.
- Given a training set of samples, the algorithms for building decision trees performs search in the space of decision trees
  - The construction of the tree is top-down and the approach is greedy.
  - "Which attribute should be tested next? Which attribute gives us more information?
  - SELECT THE BEST ATTRIBUTE !!!!!!
  - A descendant node is then created for each possible value of this attribute and

Fundamental Question

examples are partitioned according to this value.

- The process is repeated for each successor node until all the examples are classified correctly or there are no attributes left.

Which attribute is the Best ???

- A statistical property called information gain measures how well a given attribute separates the training examples.
- Information gain uses the notion of entropy, commonly used in information theory.
- Information gain = expected reduction of entropy.

I Attribute Selection Measure: Information Gain

- Select the attribute with the highest information gain.

used by ID3 algorithm

- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i

estimated by $P(C_i, D/D)$

- What is Entropy ???

Entropy is a measure of the randomness in the information being processed.

The higher the entropy, the harder is to draw any conclusions from that information.

Consider a segment 'S' of a dataset having 'c' number of class labels.

Let p_i be the propotion of examples in S having i^{th} class label.

Entropy is defined as

$$\text{Entropy}(S) = \sum_{i=1}^{c} -p_i \log_2(p_i)$$

Information Gain.

Information gain tells us how important a given attribute of the feature vector is.

- This value is used to decide the ordering of attributes in the nodes of a decision tree.

(S or D) Let S be set of examples, A be a S and D are same. feature/attribute.

S_v be subset of S with A = v and Values(A) be the set of all possible values of A.

Information Gain of an attribute A relative to S, denoted by Gain(S, A) is

$$Gain(S,A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} \times Entropy(S_v)$$

ID3 Algorithm.

ID3(X, T, Attrs)

X : training examples
T : target attribute
Attrs : other attributes, Initially all attributes

Create Root node.

If all x's are +
return Root with class +

If all x's are −
return Root with class −

If Attrs is empty
return Root with class most common value of T in X

Else

A ← best attribute

decision attribute for Root ← A

For each possible value v_i of A:

- add a new branch below Root, for test $A = v_i$
- X_i ← subset of X with $A = v_i$
- If X_i is empty then

  add a new leaf with class the most common value of T in X

  else

  add the subtree generated by.

  ID3 (X_i, T, Attrs − {A})

return Root.

Class P : buys_computer = "yes"

Class N : buys_computer = "no"

(or Info(S)) $\text{Entropy}(S) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940$ bits

~~Entropy(age)~~ =

$$\text{Info}_{age}(S) = \frac{5}{14} \times \left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right) +$$

$$\frac{4}{14} \times \left(-\frac{4}{4}\log_2\frac{4}{4} - \frac{0}{4}\log_2\frac{0}{4}\right) +$$

$$\frac{5}{14} \times \left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right)$$

$= 0.694$ bits

$$Gain(S, age) = Info(D) - Info_{age}(D)$$

$$= 0.940 - 0.694$$

$$= \underline{0.246 \text{ bits}}$$

$$Info_{income}(S) = \frac{4}{14}\left(\frac{-2}{4}\log_2\left(\frac{2}{4}\right) - \frac{2}{4}\log_2\left(\frac{2}{4}\right)\right) + \frac{6}{14}\left(\frac{-4}{6}\log_2\left(\frac{4}{6}\right) - \frac{2}{6}\log_2\left(\frac{2}{6}\right)\right) + \frac{4}{14}\left(\frac{-3}{4}\log_2\left(\frac{3}{4}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right)\right)$$

$$= 0.9097$$

$$Gain(S, income) = 0.9403 - 0.9097$$

$$= \underline{0.03 \text{ bits}}$$

$$Info_{student}(S) = \frac{7}{14}\left(\frac{-6}{7}\log_2\left(\frac{6}{7}\right) - \frac{1}{7}\log_2\left(\frac{1}{7}\right)\right) + \frac{7}{14}\left(\frac{-3}{7}\log_2\left(\frac{3}{7}\right) - \frac{4}{7}\log_2\left(\frac{4}{7}\right)\right)$$

$$= 0.7878$$

$$Gain(S, student) = 0.9403 - 0.7878$$

$$= \underline{0.152 \text{ bits}}$$

$$\text{Info}_{\text{credit-rating}}(S) = \frac{6}{14}\left(-\frac{3}{6}\log_2\left(\frac{3}{6}\right) - \frac{3}{6}\log_2\left(\frac{3}{6}\right)\right) + \frac{8}{14}\left(-\frac{6}{8}\log_2\left(\frac{6}{8}\right) - \frac{2}{8}\log_2\left(\frac{2}{8}\right)\right)$$

$$= 0.892$$

$$\text{Gain}(S, \text{credit-rating}) = 0.9403 - 0.892 = 0.0483$$

Here the attribute with maximum Information gain is age. with a value of 0.246 bels So it will be chosen as the root of decision-tree.

The process is repeated for the rest of the attributes.

II Attribute Selection Measure: Gain Ratio

used by C4.5

- Information gain measure is biased towards attributes with large number of values.
- C4.5 (a successor of ID3 algorithm) uses gain ratio to overcome the problem (normalization to information gain)

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \log_2\left(\frac{|D_j|}{|D|}\right)$$

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}$$

The attribute with maximum gain ratio is selected as the splitting attribute.

eg:- To find the gain-Ratio of the attribute 'income'

$$Gain(S, income) = 0.029$$

$$Split\text{-}Info_{income}(D) = -\frac{4}{14}\log_2\left(\frac{4}{14}\right) - \frac{6}{14}\log_2\left(\frac{6}{14}\right) - \frac{4}{14}\log_2\left(\frac{4}{14}\right)$$

$$= 0.926$$

$$\therefore GainRatio(income) = \frac{Gain(D, income)}{Split\text{-}Info_{income}(D)}$$

$$= \frac{0.029}{0.926} = 0.031$$

III Attribute Selection Measure: Gini Index.

(used by CART)

If a dataset 'D' contains examples from 'n' classes, gini index is defined as

$$gini(D) = 1 - \sum_{j=1}^{n} p_j^2$$

where p_j is the relative frequency of class j in D.

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- Reduction in Impurity is defined as

$$\Delta gini(A) = gini(D) - gini_A(D)$$

- The attribute that maximizes the reduction in impurity (minimum Gini Index) is selected as the splitting attribute.

eg: $$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Now we need to find the gini index for each attribute.

eg: We start with "income"

Consider the subset

$S_1 = \{low, medium\}, \{high\}$

Gini income $\in S_1$

$$= \frac{10}{14} Gini(D_1) + \frac{4}{14} Gini(D_2)$$

$$= \frac{10}{14}\left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14}\left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right)$$

$$= \underline{0.443}$$

$S_2 = \{low, high\}, \{medium\}$

Gini income $\in S_2$

$$= \frac{8}{14} Gini(D_1) + \frac{6}{14} Gini(D_2)$$

$$= \frac{8}{14}\left[1 - \left(\frac{5}{8}\right)^2 - \left(\frac{3}{8}\right)^2\right] + \frac{6}{14}\left[1 - \left(\frac{4}{6}\right)^2 - \left(\frac{2}{6}\right)^2\right]$$

$$= \underline{0.4581}$$

$S_3 = \{high, medium\}, \{low\}$

Gini income $\in S_3$

$$\frac{10}{14}\left[1 - \left(\frac{6}{10}\right)^2 - \left(\frac{4}{10}\right)^2\right] + \frac{4}{14}\left[1 - \left(\frac{3}{4}\right)^2 - \left(\frac{1}{4}\right)^2\right]$$

$$= \underline{0.4500}$$

Take the minm value = 0.443

$\therefore \Delta$ Gini(income) = 0.459 − 0.443

= 0.016

The attribute that provides the smallest $gini_A(D)$ (or largest reduction in impurity) is chosen to split the node.

Comparison of the various Attribute Selection Measures.

1. Information gain:
   - biased towards multivalued attributes
2. Gain Ratio
   - tends to prefer unbalanced splits in which one partition is much smaller than others.
3. Gini Index
   - biased to multivalued attributes
   - has difficulty when no. of classes is large
   - tends to favor tests that result in equal sized partitions and purity in both partitions.

Bayesian Classification

- Statistical classifier.
- predicts class membership probabilities ie, the probability that a given tuple belongs to a particular class.
- It is based on Bayes' theorem.
- Naive Bayes classifier is found to be comparable in performance with decision tree and other neural network classifiers.
- Naive Bayes classifiers assume that the effect of an attribute value on a given class is independent of the values of other attributes

Assumption

- It simplifies the computation and the term "naive"

Bayes' Theorem

- Let X be a data tuple.
- In Bayesian terms, X is considered "evidence". X is described by a set of "n" attributes

- H be some hypothesis such that data tuple X belongs to some class 'C'.

- For classification problems, we want to determine P(H/X) ie the probability that hypothesis "H" holds given the "evidence" or observed data "X".

- we are looking for the probability that a tuple X belongs to a class C given the attribute description of X.

$$P(H/X) = \frac{P(X/H) \cdot P(H)}{P(X)}$$

P(H/X) - posterior probability of H conditioned on X.

P(H) - prior probability of H. (independent of X).

P(X/H) - posterior probability of X conditioned on H.

P(X) - prior probability of X

eg: Suppose X is a 35 year old customer with an income of $ 40,000. H is hypothesis that our customer will buy a computer.

P(H/X) - probability that customer X will buy a computer given the customer age & income

$P(H)$ – probability that any given customer will buy a computer regardless of age, income or any other information.

$P(X|H)$ – probability that a customer X is 35 years old and earns $40,000, given that the customer will buy a computer.

$P(X)$ – probability that a person from the set of customers is 35 years old and earns $40,000.

Naive Bayesian classification: Working

1. Let D be a training set of tuples with associated class labels. Each tuple is a n-dimensional attribute vector.

$$X = (x_1, x_2, \ldots, x_n)$$

2. Suppose that there are m classes. $C_1, C_2, \ldots, C_m$. Given a tuple X, the classifier will predict that X belongs to the class having the highest posterior probability conditioned on X.

i.e we say X belongs to a class C_i iff

$$P(C_i|X) > P(C_j|X) \text{ for } 1 \leq j \leq m,\ j \neq i$$

The class C_i for which $P(C_i|X)$ is maximized is called the maximum posterior hypothesis.

By Bayes' theorem

$$P(C_i|X) = \frac{P(X|C_i) \cdot P(C_i)}{P(X)}$$

3. $P(X)$ is a constant for all classes and so only $P(X|C_i) \cdot P(C_i)$ need to be maximized

→ If the class prior probabilities ie $P(C_i)$ are not known, it is commonly assumed as $P(C_1) = P(C_2) \ldots = P(C_m)$

→ Otherwise

$$P(C_i) = \frac{|C_i, D|}{|D|}$$

$|C_i, D|$ - no: of training tuples of class C_i in D.

4. Involves computation of $P(X|C_i)$.

Naive algorithm - assumption of class conditional independence - values of attributes are conditionally independent of one another.

$$P(X/C_i) = \prod_{k=1}^{n} P(x_k/C_i)$$

$$= P(x_1/C_i) \times P(x_2/C_i) \times \ldots P(x_n/C_i)$$

To compute $P(X/C_i)$ we consider the following:

a) If A_k is categorical, then $P(x_k/C_i)$ is no: of tuples of class C_i in D having value x_k for A_k, divided by $|C_i, D|$ no: of tuples of class C_i in D.

b) If A_k is continous-valued, it is assumed to have a gaussian distribution with a mean μ and standard deviation σ by.

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

eg:- AllElectronics Customer DataBase.

Suppose $X = (age = youth, income = medium, student = yes, credit\text{-}rating = fair)$.
to classify, tuple

By Bayes' Theorem

$$P(C_i/X) = P(X/C_i) \cdot P(C_i)$$

C_i { buys-computer / yes ; buys-computer / no.

To find $P(C_i)$

$P(C_1) = P(\text{buys\_computer} = \text{yes}) = \frac{9}{14} = 0.643$

$P(C_2) = P(\text{buys\_computer} = \text{no}) = \frac{5}{14} = 0.357$

To find $P(X|C_i)$

$P(X|C_1) = P(X|\text{buys\_computer} = \text{yes})$

$= P(\text{age}|\text{buys-computer} = \text{yes}) \times$
(age = youth)
$P(\text{income} = \text{medium}|\text{buys-computer} = \text{yes}) \times$
$P(\text{student} = \text{yes}|\text{buys-computer} = \text{yes}) \times$
$P(\text{credit-rating} = \text{fair}|\text{buys-computer} = \text{yes})$

$= \frac{2}{9} \times \frac{4}{9} \times \frac{6}{9} \times \frac{5}{9} = 0.044.$

$P(X|C_2) = P(X|\text{buys-computer} = \text{no})$

$= P(\text{age} = \text{youth}|\text{buys-computer} = \text{no}) \times$
$P(\text{income} = \text{medium}|\text{buys-computer} = \text{no}) \times$
$P(\text{student} = \text{yes}|\text{buys-computer} = \text{no}) \times$
$P(\text{credit-rating} = \text{fair}|\text{buys-computer} = \text{no}).$

$= \frac{3}{5} \times \frac{2}{5} \times \frac{1}{5} \times \frac{2}{5} = 0.192.$

$\therefore P(X|C_1) \times P(C_1)$

$= P(X|\text{buys\_computer}=\text{yes}) \cdot P(\text{buys\_computer}=\text{yes})$

$= 0.044 \times 0.643 = 0.028$

$P(X|C_2) \times P(C_2)$

$= P(X|\text{buys\_computer}=\text{no}) \cdot P(\text{buys\_computer}=\text{no})$

$= 0.0192 \times 0.357 = 0.0068$

So we conclude that X belongs to class C_1 i.e buys_computer = yes

Since $P(C_1|X) > P(C_2|X)$

→ Laplacian Correction.

We have $P(X|C_i) = P(x_1|C_i) \times P(x_2|C_i) \ldots\ldots$

If any one of $P(x_k|C_i) = 0$ then $P(X|C_i)$ will be 0.

There is a simple trick to avoid this problem. We assume that our training database D is so large that adding one to each count would only make

a negligible difference in the estimated probability value, yet could conveniently avoid the case of probability value of zero. This technique for probability estimation is called Laplacian Correction / Laplacian Estimator.

eg: Suppose that for class buys-computer = yes in some DB contains 1000 tuples.
we have 0 tuples with income = low,
990 tuples with income = medium and
10 tuples with income = high.

$$P(\text{income} = \text{low} \mid \text{buys-computer} = \text{yes}) = \frac{0}{1000} = 0$$

$$P(\text{income} = \text{medium} \mid \text{buys-computer} = \text{yes}) = \frac{990}{1000} = 0.990$$

$$P(\text{income} = \text{high} \mid \text{buys-computer} = \text{yes}) = \frac{10}{1000} = 0.010$$

Using Laplacian Correction for the 3 quantities, we pretend that we have 1 more tuple to each income-value pair

Then the probabilities become

$\frac{1}{1003} = 0.001$, $\frac{991}{1003} = 0.988$.

$\frac{11}{1003} = 0.011$