# DWDM SESSIONAL - II IMPORTANT QUESTIONS

## UNIT –3

1. Decision tree algorithm with an example.

- A **decision tree** is a flowchart-like tree structure, where

  Each internal node (non-leaf node) denotes a test on an attribute

  Each branch represents an outcome of the test

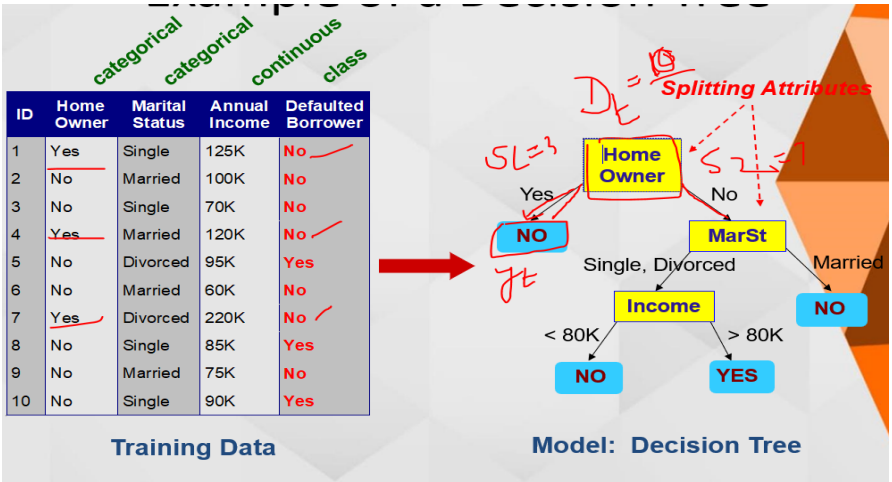
  Each leaf node (or terminal node) holds a class label

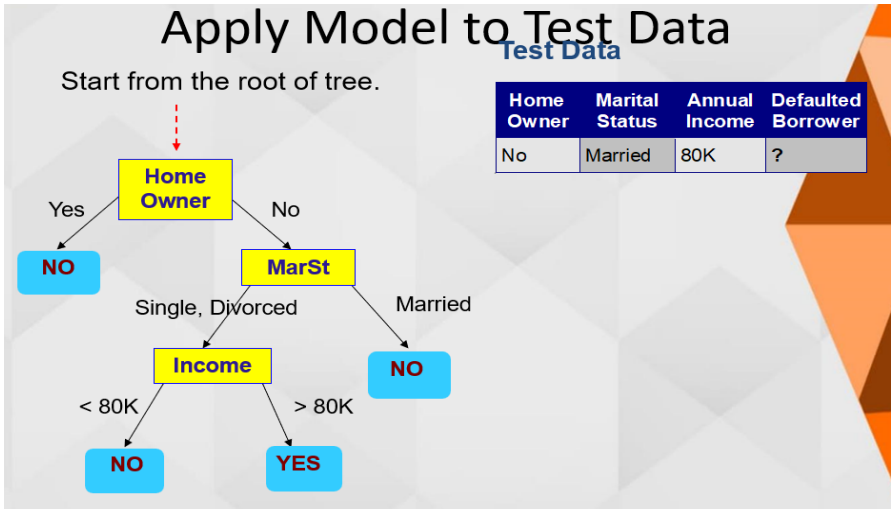
  - Can be represented by logical formulas

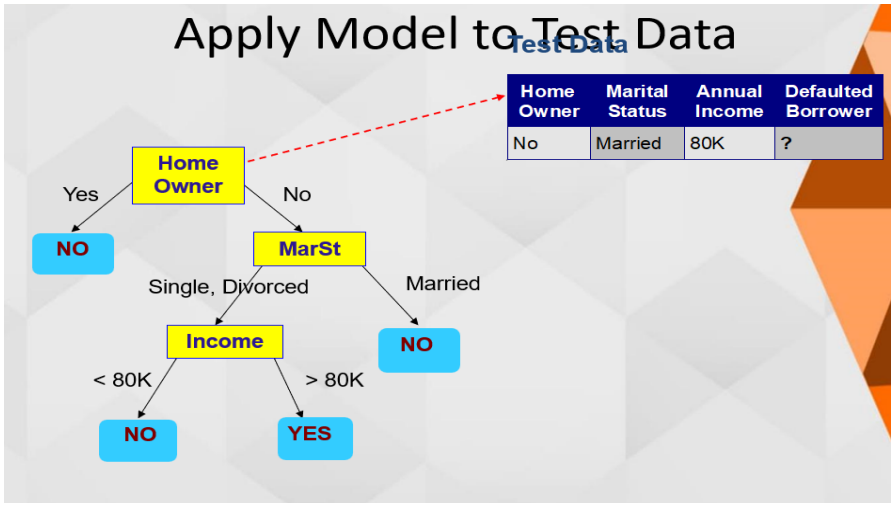
### How does the Decision Tree algorithm Work?

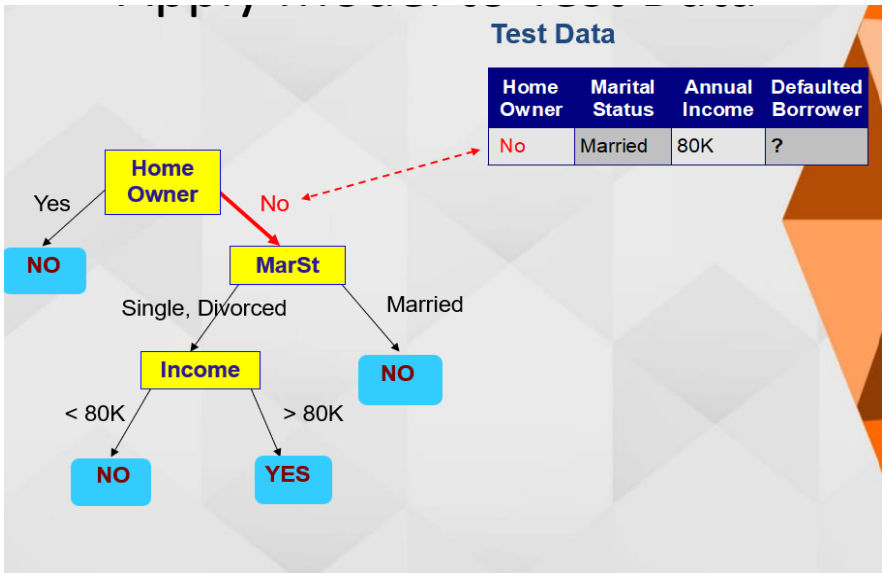
- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:**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

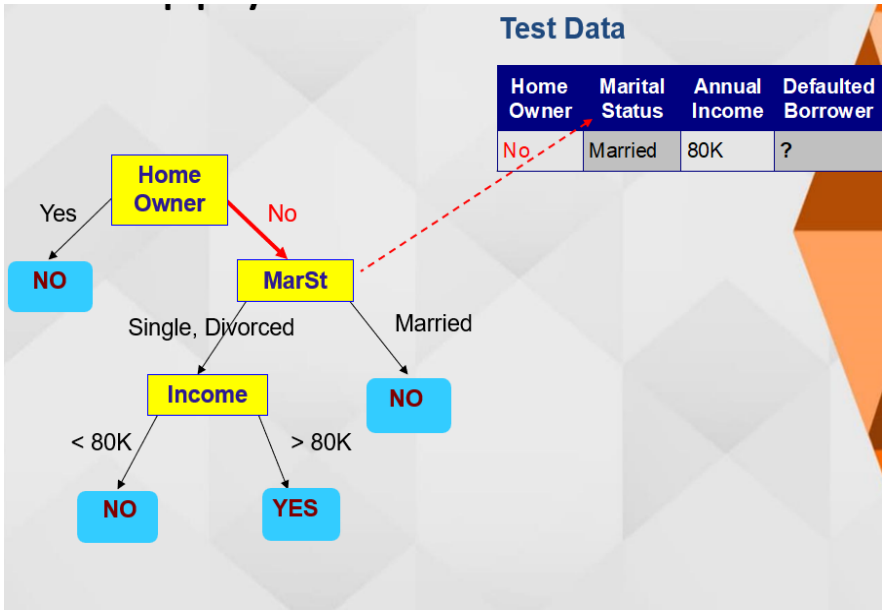
Example of a Decision Tree:

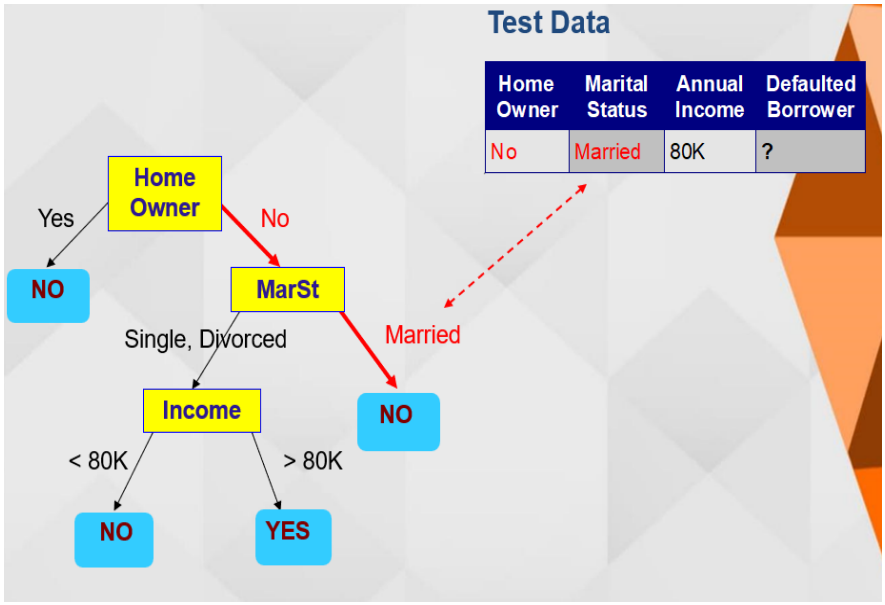


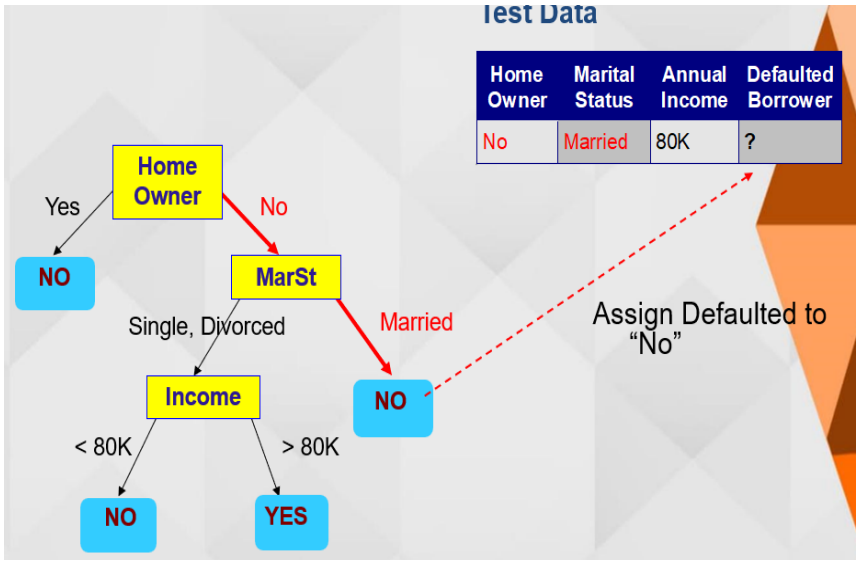

Apply Model to Test Data
Start from the root of tree.
Test Data
Home Owner
Marital Status
Annual Income
Defaulted Borrower
No
Married
80K
?
Home Owner
Yes
No
NO
MarSt
Single, Divorced
Married
Income
NO
< 80K
> 80K
NO
YES


Apply Model to Test Data
Test Data
Home Owner
Marital Status
Annual Income
Defaulted Borrower
No
Married
80K
?
Home Owner
Yes
No
NO
MarSt
Single, Divorced
Married
Income
NO
< 80K
> 80K
NO
YES


Test Data
Home Owner
Marital Status
Annual Income
Defaulted Borrower
No
Married
80K
?
Home Owner
Yes
No
NO
MarSt
Single, Divorced
Married
Income
NO
< 80K
> 80K
NO
YES


Test Data
Home Owner
Marital Status
Annual Income
Defaulted Borrower
No
Married
80K
?
Home Owner
Yes
No
NO
MarSt
Single, Divorced
Married
Income
NO
< 80K
> 80K
NO
YES


Test Data
Home Owner
Marital Status
Annual Income
Defaulted Borrower
No
Married
80K
?
Home Owner
Yes
No
NO
MarSt
Single, Divorced
Married
Income
NO
< 80K
> 80K
NO
YES


Test Data
Home Owner
Marital Status
Annual Income
Defaulted Borrower
No
Married
80K
?
Home Owner
Yes
No
NO
MarSt
Single, Divorced
Married
Income
NO
< 80K
> 80K
NO
YES
Assign Defaulted to "No"

## 2. Attribute Selection Measures (Information Gain,Gain Ratio,Gini index)

**Attribute Selection Measures:**

**Information Gain :**

Information gain is used for deciding the best features/attributes that render maximum data about a class. It follows the method of entropy while aiming at reducing the level of entropy, starting from the root node to the leaf nodes.

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i),$$

where $p_i$ is the nonzero probability that an arbitrary tuple in $D$ belongs to class $C_i$ and is estimated by $|C_{i,D}|/|D|$.

- **entropy of attribute A with values {$a_1$,$a_2$,…,$a_v$}**

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j).$$

- **information gained by branching on attribute A**

$$Gain(A) = Info(D) - Info_A(D).$$

**Gain ratio** – The information gain measure is biased approaching tests with several results. It can select attributes having a high number of values. For instance, consider an attribute that facilitates as a unique identifier, including product ID.

**Gini index** – The Gini index can be used in CART. The Gini index calculates the impurity of D, a data partition or collection of training tuples, as

$$\text{Gini(D)}=1-\sum_{i=1}^{m} p_{2i}$$

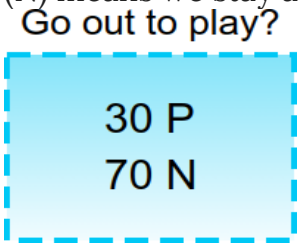
- **If a data set *T* contains examples from *n* classes, gini index, *gini*(*T*) is defined as**

**where $p_j$ is the relative frequency of class *j* in *T*.**

- **If a data set *T* is split into two subsets $T_1$ and $T_2$ with sizes $N_1$ and $N_2$ respectively, the *gini* index of the split data contains examples from *n* classes, the *gini* index *gini*(*T*) is defined as**

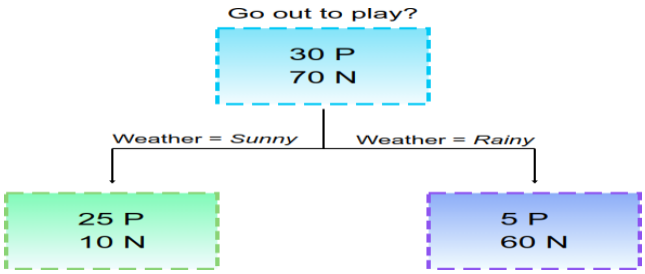
$$gini_{split}(T) = \frac{N_1}{N} gini(T_1) + \frac{N_2}{N} gini(T_2)$$

Let's take an example.
Suppose we have a dataset of our last 100 days which records if we go outside to play or not. Positive (P) means we do go outside, while Negative (N) means we stay at home to study Data Mining.




The Entropy of our initial dataset is

$$H = -(0.3 * \log_2 0.3 + 0.7 * \log_2 0.7)$$
$$\approx 0.88$$




The Entropies of the resulting 2 sub-datasets:

$$H_{Weather=Sunny} = -(\frac{25}{35} * \log_2 \frac{25}{35} + \frac{10}{35} * \log_2 \frac{10}{35}) \approx 0.86$$

$$H_{Weather=Rainy} = -(\frac{5}{65} * \log_2 \frac{5}{65} + \frac{60}{65} * \log_2 \frac{60}{65}) \approx 0.39$$

To illustrate, the Information Gain using Weather is:

$$\begin{aligned} IG_{Weather} &= H - (\sum \frac{|D_j|}{|D|} * H_j) \\ &= H - (\frac{35}{100} H_{Weather=Sunny} + \frac{65}{100} H_{Weather=Rainy}) \\ &\approx 0.88 - 0.55 \\ &\approx 0.33 \end{aligned}$$

The Gain Ratio is:

$$GainRatio = \frac{\text{Information Gain}}{\text{Intrinsic Information}}$$

Plug it to the above example:

$$\begin{aligned}\text{Gain Ratio}_{\text{Weather}} &\approx \frac{0.33}{0.93} \\ &\approx 0.35\end{aligned}$$

The Gini of a dataset is:

$$\text{Gini} = 1 - (\sum p_i^2)$$

where is the proportion of a label.

The Gini of the above original dataset is:

$$\begin{aligned}\text{Gini}(D) &= 1 - (0.3^2 + 0.7^2) \\ &= 0.42\end{aligned}$$

The *Gini* of a split is computed by:

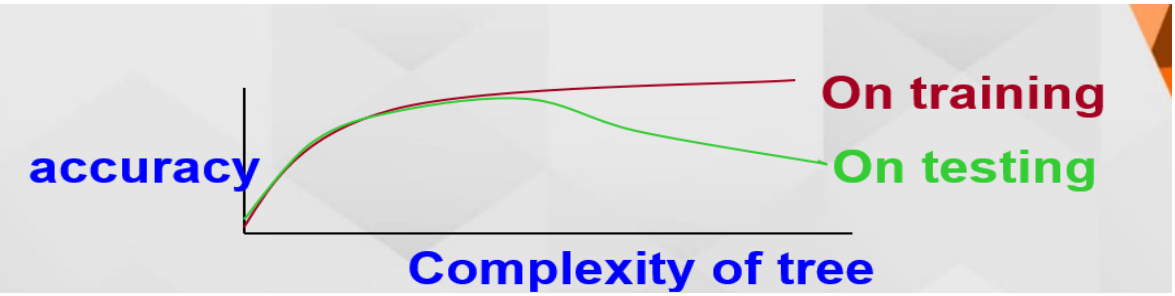
$$\begin{aligned}\text{Gini}_{\text{split = Weather}} &= \frac{35}{100} * \text{Gini}_{\text{Sunny}} + \frac{65}{100} * \text{Gini}_{\text{Rainy}} \\ &\approx 0.35 * 0.41 + 0.65 * 0.14 \\ &\approx 0.2345\end{aligned}$$

**3. Model Overfitting(Tree Pruning) in Classification**

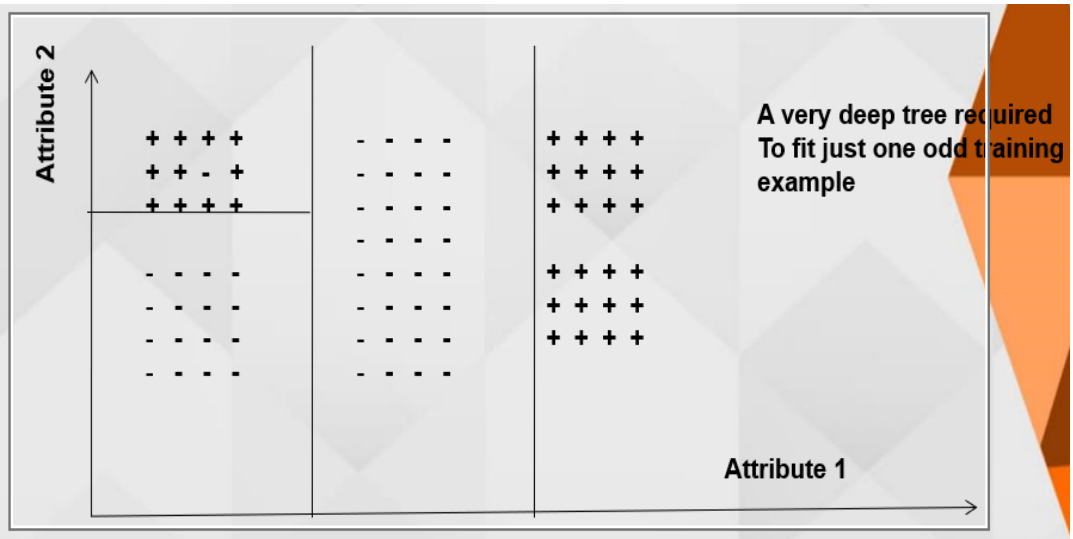
Overfitting the Data

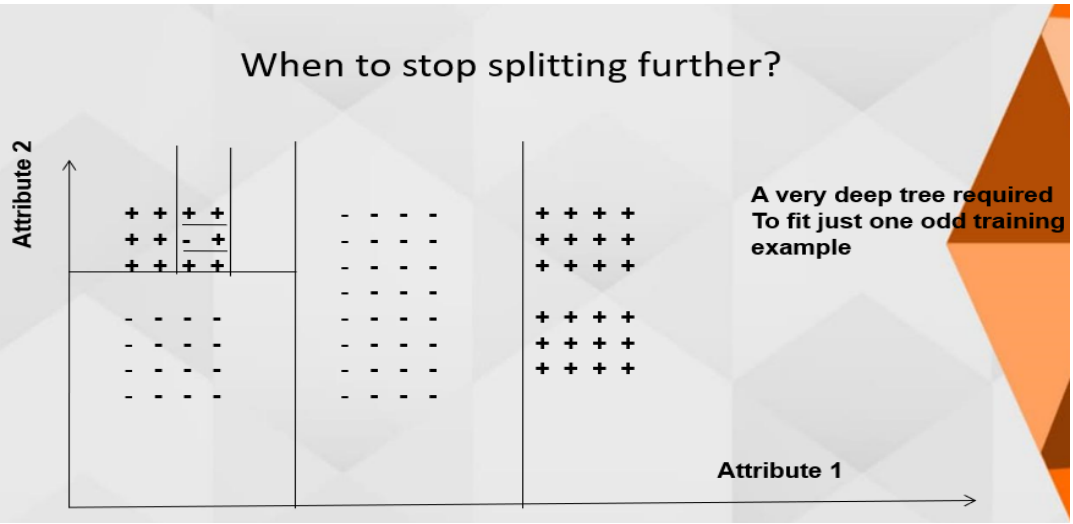
- Learning a tree that classifies the training data perfectly may not lead to the tree with the best generalization performance.
  - There may be noise in the training data the tree is fitting
  - The algorithm might be making decisions based on very little data

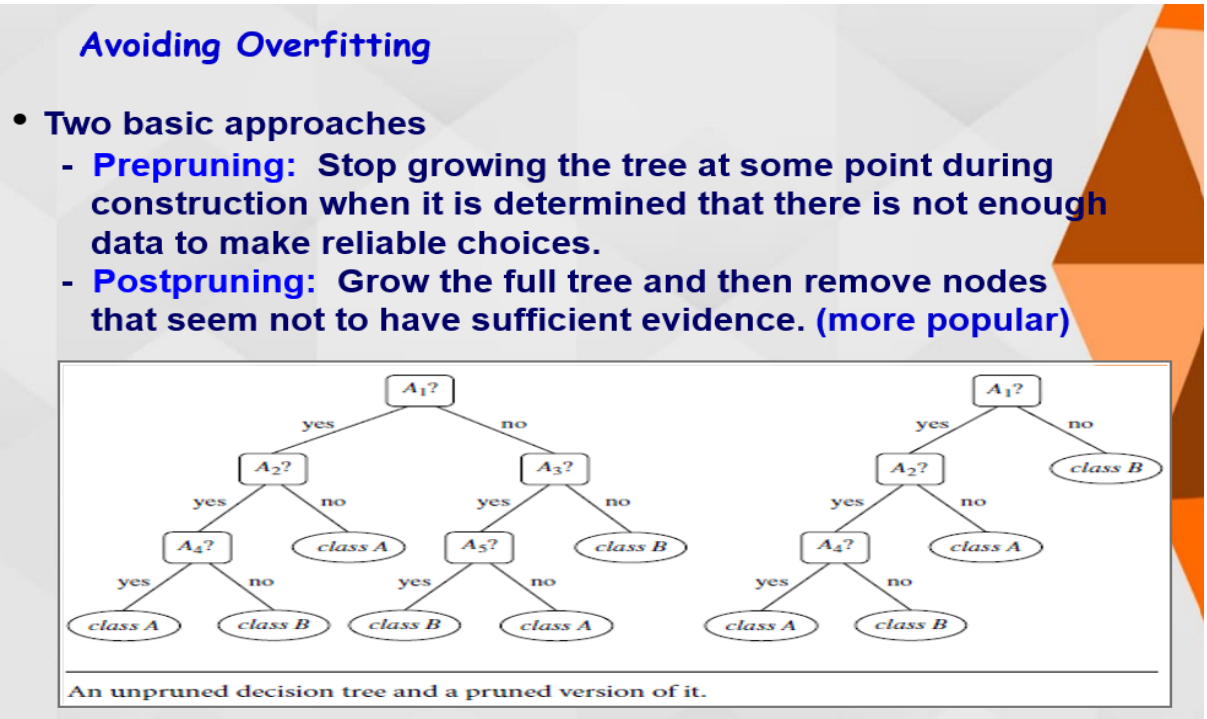
A hypothesis *h* is said to overfit the training data if the is another hypothesis, h', such that *h* has smaller error than *h'* on the training data but *h* has larger error on the test data than *h'*




Overfitting






**4. Any Training Data Set will be given; You have to classify the data set according to Decision tree algorithm (or ID3 algorithm).**

Let us take an example of the last 10 days weather dataset with attributes outlook, temperature, wind, and humidity. The outcome variable will be playing cricket or not. We will use the ID3 algorithm to build the decision tree.

| Day | Outlook | Temperature | Humidity | Wind | Play crick |
|---|---|---|---|---|---|
| 1 | Sunny | Hot | High | Weak | No |
| 2 | Sunny | Hot | High | Strong | No |
| 3 | Overcast | Hot | High | Weak | Yes |
| 4 | Rain | Mild | High | Weak | Yes |
| 5 | Rain | Cool | Normal | Weak | Yes |
| 6 | Rain | Cool | Normal | Strong | No |
| 7 | Overcast | Cool | Normal | Strong | Yes |
| 8 | Sunny | Mild | High | Weak | No |
| 9 | Sunny | Cool | Normal | Weak | Yes |
| 10 | Rain | Mild | Normal | Weak | Yes |

| Day | Outlook | Temperature | Humidity | Wind | Play crick |
|---|---|---|---|---|---|
| 11 | Sunny | Mild | Normal | Strong | Yes |
| 12 | Overcast | Mild | High | Strong | Yes |
| 13 | Overcast | Hot | Normal | Weak | Yes |
| 14 | Rain | Mild | High | Strong | No |

**Step1:** The first step will be to create a root node.
**Step2:** If all results are yes, then the leaf node "yes" will be returned else the leaf node "no" will be returned.
**Step3:** Find out the Entropy of all observations and entropy with attribute "x" that is E(S) and E(S, x).
**Step4:** Find out the information gain and select the attribute with high information gain.
**Step5:** Repeat the above steps until all attributes are covered.
**Calculation of Entropy:**

Yes No

9 5

$$Entropy(S) = \sum_{x \in X} p(x) \log_2 \frac{1}{p(x)}$$

$$Entropy(S) = -\left(\frac{9}{14}\right)\log_2\left(\frac{9}{14}\right) - \left(\frac{5}{14}\right)\log_2\left(\frac{5}{14}\right)$$

$$= 0.940$$

If entropy is zero, it means that all members belong to the same class and if entropy is one then it means that half of the tuples belong to one class and one of them belong to other class. 0.94 means fair distribution.

Find the information gain attribute which gives maximum information gain.

**For Example** "Wind", it takes two values: Strong and Weak, therefore, x = {Strong, Weak}.

$$IG(S, Wind) = H(S) - \sum_{i=0}^{n} P(x) * H(x)$$

Find out H(x), P(x) for x =weak and x= strong. H(S) is already calculated above.

Weak= 8

Strong= 8

$$P(S_{weak}) = \frac{Number\ of\ Weak}{Total}$$
$$= \frac{8}{14}$$
$$P(S_{strong}) = \frac{Number\ of\ Strong}{Total}$$
$$= \frac{6}{14}$$

For “weak” wind, 6 of them say “Yes” to play cricket and 2 of them say “No”. So entropy will be:

$$Entropy(S_{weak}) = -\left(\frac{6}{8}\right)\log_2\left(\frac{6}{8}\right) - \left(\frac{2}{8}\right)\log_2\left(\frac{2}{8}\right)$$
$$= 0.811$$

For “strong” wind, 3 said “No” to play cricket and 3 said “Yes”.

$$Entropy(S_{strong}) = -\left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right) - \left(\frac{3}{6}\right)\log_2\left(\frac{3}{6}\right)$$
$$= 1.000$$

This shows perfect randomness as half items belong to one class and the remaining half belong to others.

**Calculate the information gain,**

$$IG(S, Wind) = H(S) - \sum_{i=0}^{n} P(x) * H(x)$$
$$IG(S, Wind) = H(S) - P(S_{weak}) * H(S_{weak}) - P(S_{strong}) * H(S_{strong})$$
$$= 0.940 - \left(\frac{8}{14}\right)(0.811) - \left(\frac{6}{14}\right)(1.00)$$
$$= 0.048$$

**Similarly the information gain for other attributes is:**

$$IG(S, Outlook) = 0.246$$
$$IG(S, Temperature) = 0.029$$
$$IG(S, Humidity) = 0.151$$

The attribute outlook has the **highest information gain** of 0.246, thus it is chosen as root.
Overcast has 3 values: Sunny, Overcast and Rain. Overcast with play cricket is always “Yes”. So it ends up with a leaf node, “yes”. For the other values “Sunny” and “Rain”.

**Table for Outlook as “Sunny” will be:**

| Temperature | Humidity | Wind | Golf |
|---|---|---|---|
| Hot | High | Weak | No |
| Hot | High | Strong | No |
| Mild | High | Weak | No |
| Cool | Normal | Weak | Yes |
| Mild | Normal | Strong | Yes |

**Entropy for "Outlook" "Sunny" is:**

$$H(S_{sunny}) = \left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) = 0.96$$

**Information gain for attributes with respect to Sunny is:**

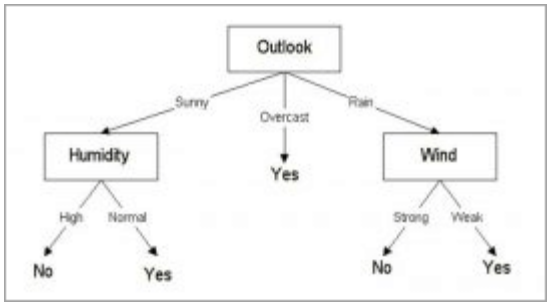
$$IG(S_{sunny}, Humidity) = 0.96$$

$$IG(S_{sunny}, Temperature) = 0.57$$

$$IG(S_{sunny}, Wind) = 0.019$$

The information gain for humidity is highest, therefore it is chosen as the next node. Similarly, Entropy is calculated for Rain. **Wind gives the highest information gain**.

**The decision tree would look like below:**




**5. Explain Scalability for Decision Tree Induction**

Scalable Decision Tree Induction Methods in Data Mining

- SLIQ (EDBT'96 — Mehta et al.)
  - builds an index for each attribute and only class list and the current attribute list reside in memory
- SPRINT (VLDB'96 — J. Shafer et al.)
  - constructs an attribute list data structure
- PUBLIC (VLDB'98 — Rastogi & Shim)
  - integrates tree splitting and tree pruning: stop growing the tree earlier
- RainForest (VLDB'98 — Gehrke, Ramakrishnan & Ganti)
  - separates the scalability aspects from the criteria that determine the quality of the tree

- builds an AVC-list (attribute, value, class label)
- BOAT (PODS'99 – Gehrke, Ganti, Ramakrishnan & Loh)

-Uses bootstrapping to create several small samples.

## Scalability Framework for RainForest

- Separates the scalability aspects from the criteria that determine the quality of the tree
- Builds an AVC-list: **AVC (Attribute, Value, Class_label)**
- **AVC-set** (of an attribute *X*)
  - Projection of training dataset onto the attribute *X* and class label where counts of individual class label are aggregated
- **AVC-group** (of a node *n*)
  - Set of AVC-sets of all predictor attributes at the node *n*

| age | income | student | credit_rating | buys_computer |
|---|---|---|---|---|
| <=30 | high | no | fair | no |
| <=30 | high | no | excellent | no |
| 31…40 | high | no | fair | yes |
| >40 | medium | no | fair | yes |
| >40 | low | yes | fair | yes |
| >40 | low | yes | excellent | no |
| 31…40 | low | yes | excellent | yes |
| <=30 | medium | no | fair | no |
| <=30 | low | yes | fair | yes |
| >40 | medium | yes | fair | yes |
| <=30 | medium | yes | excellent | yes |
| 31…40 | medium | no | excellent | yes |
| 31…40 | high | yes | fair | yes |
| >40 | medium | no | excellent | no |

| *age* | *buys_computer* yes | no |
|---|---|---|
| youth | 2 | 3 |
| middle_aged | 4 | 0 |
| senior | 3 | 2 |

| *income* | *buys_computer* yes | no |
|---|---|---|
| low | 3 | 1 |
| medium | 4 | 2 |
| high | 2 | 2 |

| *student* | *buys_computer* yes | no |
|---|---|---|
| yes | 6 | 1 |
| no | 3 | 4 |

| *credit_ratting* | *buys_computer* yes | no |
|---|---|---|
| fair | 6 | 2 |
| excellent | 3 | 3 |

The use of data structures to hold aggregate information regarding the training data (e.g., these AVC-sets describing Table 8.1's data) are one approach to improving the scalability of decision tree induction.

## UNIT– 4

1. Define Association and explain market basket analysis in detail.

## Association Analysis:

Association rule mining finds interesting association or correlation relationships among a large set of data items.

## Market basket analysis:

The example of association rule mining is market basket analysis.
This process analyzes customer buying habits by finding associations between the different items that customers place in their "shopping baskets" as shown in figure a. The discovery of such
associations can help retailers develop marketing strategies by gaining insight into which items are
frequently purchased together by customers. For instance, if customers are buying milk, how likely are
they to also buy bread on the same trip to the supermarket? Such information can lead to increased
sales by helping retailers to do selective marketing and plan their shelf space. For instance, placing
milk and bread within close proximity may further encourage the sale of these items together within
single visits to the store.

- Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction
- **Market-Basket transactions**

## Market-Basket transactions

| TID | Items |
|---|---|
| 1 | Bread, Milk |
| 2 | Bread, Sugar, Flour, Eggs |
| 3 | Milk, Sugar, Flour, Coke |
| 4 | Bread, Milk, Sugar, Flour |
| 5 | Bread, Milk, Sugar, Coke |

**Example of Association Rules**

{Sugar} → {Flour},
{Milk, Bread} → {Eggs,Coke},
{Flour, Bread} → {Milk},

Implication means co-occurrence, not causality!

## Definition: Frequent Itemset

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Sugar}
  - k-itemset
    - An itemset that contains k items

    **E.g.,** {Milk, Bread, Sugar} – 3 itemset

    The null set is an itemset that doesnot contain any items.
- **Support count (σ)**
  - Frequency of occurrence of an itemset(No. of transactions that contain a particular itemset.)
  - E.g. σ({Milk, Bread, Sugar}) = 2
- **Support**
  - Fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Sugar}) = 2/5
- **Frequent Itemset**
  - An itemset whose support is greater than or equal to a *minsup* threshold

| TID | Items |
|---|---|
| 1 | Bread, Milk |
| 2 | Bread, Sugar, Flour, Eggs |
| 3 | Milk, Sugar, Flour, Coke |
| 4 | Bread, Milk, Sugar, Flour |
| 5 | Bread, Milk, Sugar, Coke |

- **Association Rule:** refer to the probability of customer purchasing one product when he purchases some other product.

An implication expression of the form X → Y, where X and Y are itemsets

Example:
{Milk, Sugar} → {Flour}

| TID | Items |
|---|---|
| 1 | Bread, Milk |
| 2 | Bread, Sugar, Flour, Eggs |
| 3 | Milk, Sugar, Flour, Coke |
| 4 | Bread, Milk, Sugar, Flour |
| 5 | Bread, Milk, Sugar, Coke |

Example:

$$\{\text{Milk}, \text{Sugar}\} \Rightarrow \{\text{Flour}\}$$

$$s = \frac{\sigma(\text{Milk}, \text{Sugar}, \text{Flour})}{|\ |} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk}, \text{Sugar}, \text{Flour})}{\sigma(\text{Milk}, \text{Sugar})} = \frac{2}{3} = 0.67$$

l Rule Evaluation Metrics

Support (s)

- Fraction of transactions that contain both X and Y

Support, s (X → Y) = σ(XUY)

N

Confidence (c)

Measures how often items in Y
appear in transactions that contain X

Confidence,c(X->Y) = $\frac{\sigma(XUY)}{\sigma(X)}$.

2. **Define Frequent itemset, Support, Confidence in detail**.

**Association Mining** searches for **frequent items** in the data-set.

In frequent mining usually the interesting associations and correlations between item sets in transactional and relational databases are found. In short, Frequent Mining shows which items appear together in a transaction or relation.

- **Support :** It is one of the measure of interestingness. This tells about usefulness and certainty of rules. **5% Support** means total 5% of transactions in database follow the rule.

```
Support(A -> B) = Support_count(A ∪ B)
```

- **Confidence:** A confidence of 60% means that 60% of the customers who purchased a milk and bread also bought butter.

```
Confidence(A -> B) = Support_count(A ∪ B) / Support_count(A)
```

**Example On finding Frequent Itemsets –** Consider the given dataset with given transactions.

| TransactionId | Items |
|---|---|
| 1 | {A,C,D} |
| 2 | {B,C,D} |
| 3 | {A,B,C,D} |
| 4 | {B,D} |
| 5 | {A,B,C,D} |

- Lets say minimum support count is 3
- Relation hold is maximal frequent => closed => frequent

***1-frequent:*** *{A} = 3; // not closed due to {A, C} and not maximal {B} = 4; // not closed due to {B, D} and no maximal {C} = 4; // not closed due to {C, D} not maximal {D} = 5; // closed item-set since not immediate super-set has same count. Not maximal*
***2-frequent:*** *{A, B} = 2 // not frequent because support count < minimum support count so ignore {A, C} = 3 // not closed due to {A, C, D} {A, D} = 3 // not closed due to {A, C, D} {B, C} = 3 // not closed due to {B, C, D} {B, D} = 4 // closed but not maximal due to {B, C, D} {C, D} = 4 // closed but not maximal due to {B, C, D}*
***3-frequent:*** *{A, B, C} = 2 // ignore not frequent because support count < minimum support count {A, B, D} = 2 // ignore not frequent because support count < minimum support count {A, C, D} = 3 // maximal frequent {B, C, D} = 3 // maximal frequent*
***4-frequent:*** *{A, B, C, D} = 2 //ignore not frequent </*

3. **Apriori algorithm with an example. ( join and prune steps also)**

> Apriori algorithm refers to the algorithm which is used to calculate the association rules between objects. It means how two or more objects are related to one another. In other words, we can say that the apriori algorithm is an association rule leaning that analyzes that people who bought product A also bought product B.

The given three components comprise the apriori algorithm.

1. Support
2. Confidence
3. Lift

Let's take an example to understand this concept.

We have already discussed above; you need a huge database containing a large no of transactions. Suppose you have 4000 customers transactions in a Big Bazar. You have to calculate the Support, Confidence, and Lift for two products, and you may say Biscuits and Chocolate. This is because customers frequently buy these two items together.

Out of 4000 transactions, 400 contain Biscuits, whereas 600 contain Chocolate, and these 600 transactions include a 200 that includes Biscuits and chocolates. Using this data, we will find out the support, confidence, and lift.

## Support

Support refers to the default popularity of any product. You find the support as a quotient of the division of the number of transactions comprising that product by the total number of transactions. Hence, we get

Support (Biscuits) = (Transactions relating biscuits) / (Total transactions)

= 400/4000 = 10 percent.

## Confidence

Confidence refers to the possibility that the customers bought both biscuits and chocolates together. So, you need to divide the number of transactions that comprise both biscuits and chocolates by the total number of transactions to get the confidence.

Hence,

Confidence = (Transactions relating both biscuits and Chocolate) / (Total transactions involving Biscuits)

= 200/400

= 50 percent.

It means that 50 percent of customers who bought biscuits bought chocolates also.

## Lift

Consider the above example; lift refers to the increase in the ratio of the sale of chocolates when you sell biscuits. The mathematical equations of lift are given below.

Lift = (Confidence (Biscuits - chocolates)/ (Support (Biscuits)

= 50/10 = 5

It means that the probability of people buying both biscuits and chocolates together is five times more than that of purchasing the biscuits alone. If the lift value is below one, it requires that the people are unlikely to buy both the items together. Larger the value, the better is the combination.

# How does the Apriori Algorithm work in Data Mining?

We will understand this algorithm with the help of an example

Consider a Big Bazar scenario where the product set is P = {Rice, Pulse, Oil, Milk, Apple}. The database comprises six transactions where 1 represents the presence of the product and 0 represents the absence of the product.

| Transaction ID | Rice | Pulse | Oil Milk | Apple | |
|---|---|---|---|---|---|
| t1 | 1 | 1 | 1 | 0 | 0 |
| t2 | 0 | 1 | 1 | 1 | 0 |
| t3 | 0 | 0 | 0 | 1 | 1 |
| t4 | 1 | 1 | 0 | 1 | 1 |
| t5 | 1 | 1 | 1 | 0 | 0 |
| t6 | 1 | 1 | 1 | 1 | 1 |

The Apriori Algorithm makes the given assumptions

- All subsets of a frequent itemset must be frequent.
- The subsets of an infrequent item set must be infrequent.
- Fix a threshold support level. In our case, we have fixed it at 50 percent.

**Step 1**

Make a frequency table of all the products that appear in all the transactions. Now, short the frequency table to add only those products with a threshold support level of over 50 percent. We find the given frequency table.

| Product | Frequency (Number of transactions) |
|---|---|
| Rice (R) | 4 |
| Pulse(P) | 5 |
| Oil(O) | 4 |
| Milk(M) | 4 |

The above table indicated the products frequently bought by the customers.

**Step 2**

Create pairs of products such as RP, RO, RM, PO, PM, OM. You will get the given frequency table.

| Itemset | Frequency (Number of transactions) |
|---|---|
| RP | 4 |
| RO | 3 |
| RM | 2 |

| | |
|---|---|
| PO | 4 |
| PM | 3 |
| OM | 2 |

**Step 3**

Implementing the same threshold support of 50 percent and consider the products that are more than 50 percent. In our case, it is more than 3

Thus, we get RP, RO, PO, and PM

**Step 4**

Now, look for a set of three products that the customers buy together. We get the given combination.

1. RP and RO give RPO
2. PO and PM give POM

**Step 5**

Calculate the frequency of the two itemsets, and you will get the given frequency table.

| Itemset | Frequency (Number of transactions) |
|---|---|
| RPO | 4 |
| POM | 3 |

If you implement the threshold assumption, you can figure out that the customers' set of three products is RPO.

**4)Frequent Pattern(FP) growth algorithm with an example**

The FP-Growth Algorithm proposed by ***Han in***

The FP-Growth Algorithm is an alternative way to find frequent item sets without using candidate generations, thus improving performance. For so much, it uses a divide-and-

conquer strategy. The core of this method is the usage of a special data structure named frequent-pattern tree (FP-tree), which retains the item set association information.

**EXAMPLE:**

Support threshold=50%, Confidence= 60%

**Table 1:**

| Transaction | List of items |
|---|---|
| T1 | I1,I2,I3 |
| T2 | I2,I3,I4 |
| T3 | I4,I5 |
| T4 | I1,I2,I4 |
| T5 | I1,I2,I3,I5 |
| T6 | I1,I2,I3,I4 |

**Solution:** Support threshold=50% => 0.5*6= 3 => min_sup=3

**Table 2: Count of each item**

| Item | Count |
|---|---|
| I1 | 4 |
| I2 | 5 |
| I3 | 4 |
| I4 | 4 |

| | |
|---|---|
| I5 | 2 |

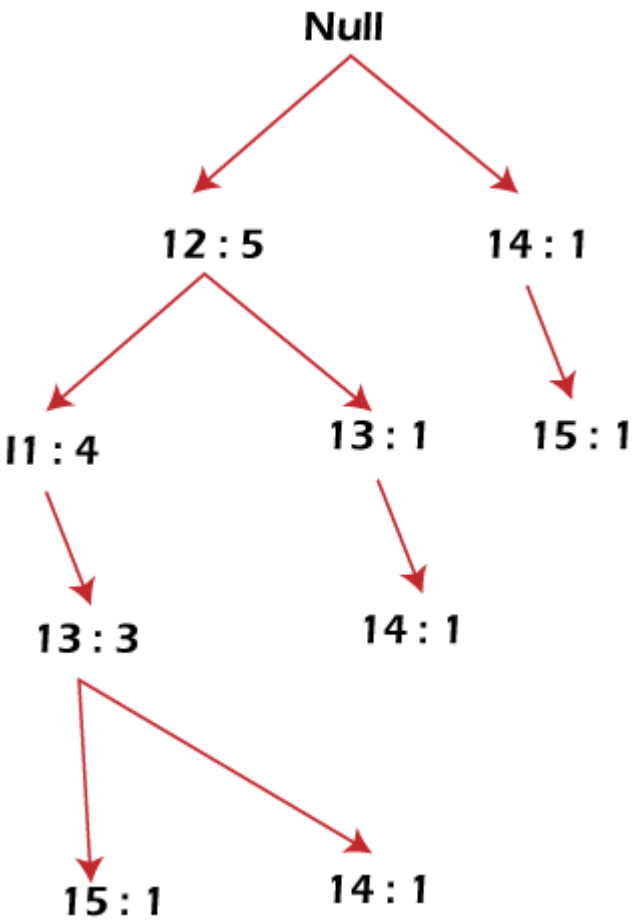
**Table 3: Sort the itemset in descending order.**

| Item | Count |
|---|---|
| I2 | 5 |
| I1 | 4 |
| I3 | 4 |
| I4 | 4 |

**Build FP Tree**

**Let's build the FP tree in the following steps, such as:**

1. Considering the root node null.
2. The first scan of Transaction T1: I1, I2, I3 contains three items {I1:1}, {I2:1}, {I3:1}, where I2 is linked as a child, I1 is linked to I2 and I3 is linked to I1.
3. T2: I2, I3, and I4 contain I2, I3, and I4, where I2 is linked to root, I3 is linked to I2 and I4 is linked to I3. But this branch would share the I2 node as common as it is already used in T1.
4. Increment the count of I2 by 1, and I3 is linked as a child to I2, and I4 is linked as a child to I3. The count is {I2:2}, {I3:1}, {I4:1}.
5. T3: I4, I5. Similarly, a new branch with I5 is linked to I4 as a child is created.
6. T4: I1, I2, I4. The sequence will be I2, I1, and I4. I2 is already linked to the root node. Hence it will be incremented by 1. Similarly I1 will be incremented by 1 as it is already linked with I2 in T1, thus {I2:3}, {I1:2}, {I4:1}.
7. T5:I1, I2, I3, I5. The sequence will be I2, I1, I3, and I5. Thus {I2:4}, {I1:3}, {I3:2}, {I5:1}.
8. T6: I1, I2, I3, I4. The sequence will be I2, I1, I3, and I4. Thus {I2:5}, {I1:4}, {I3:3}, {I4 1}.

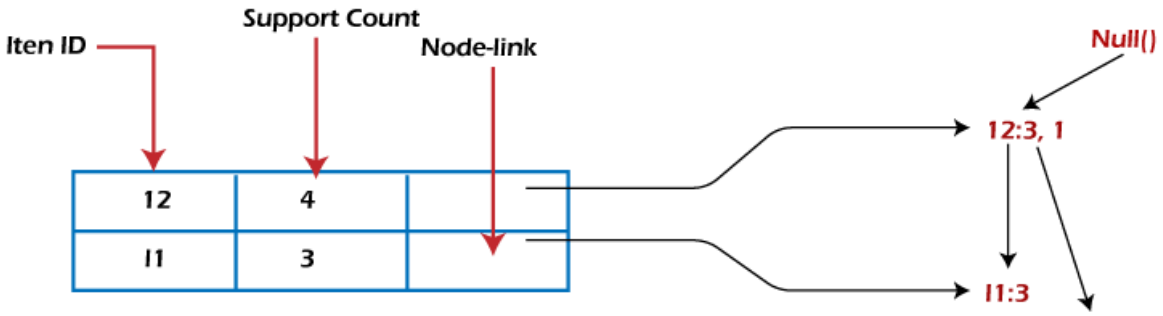



**Mining of FP-tree is summarized below:**

1. The lowest node item, I5, is not considered as it does not have a min support count. Hence it is deleted.
2. The next lower node is I4. I4 occurs in 2 branches , {I2,I1,I3:,I41},{I2,I3,I4:1}. Therefore considering I4 as suffix the prefix paths will be {I2, I1, I3:1}, {I2, I3: 1} this forms the conditional pattern base.
3. The conditional pattern base is considered a transaction database, and an FP tree is constructed. This will contain {I2:2, I3:2}, I1 is not considered as it does not meet the min support count.
4. This path will generate all combinations of frequent patterns : {I2,I4:2},{I3,I4:2},{I2,I3,I4:2}
5. For I3, the prefix path would be: {I2,I1:3},{I2:1}, this will generate a 2 node FP-tree : {I2:4, I1:3} and frequent patterns are generated: {I2,I3:4}, {I1:I3:3}, {I2,I1,I3:3}.
6. For I1, the prefix path would be: {I2:4} this will generate a single node FP-tree: {I2:4} and frequent patterns are generated: {I2, I1:4}.

| Item | Conditional Pattern Base | Conditional FP-tree | Frequent Patterns G |
|---|---|---|---|
| I4 | {I2,I1,I3:1},{I2,I3:1} | {I2:2, I3:2} | {I2,I4:2},{I3,I4:2},{I2,I3,I4 |
| I3 | {I2,I1:3},{I2:1} | {I2:4, I1:3} | {I2,I3:4}, {I1:I3:3}, {I2,I1, |
| I1 | {I2:4} | {I2:4} | {I2,I1:4} |

The diagram given below depicts the conditional FP tree associated with the conditional node I3.




**5)Compact Representation of Frequent Item Sets.**

**Compact Representation of Frequent Itemset**

- **What happens when you have a large market basket data with over a hundred items?**
- The number of frequent itemsets grows exponentially and this in turn creates an issue with storage and it is for this purpose that alternative representations have been derived which reduce the initial set but can be used to generate all other frequent itemsets.
- The Maximal and Closed Frequent Itemsets are two such representations that are subsets of the larger frequent itemset**.**

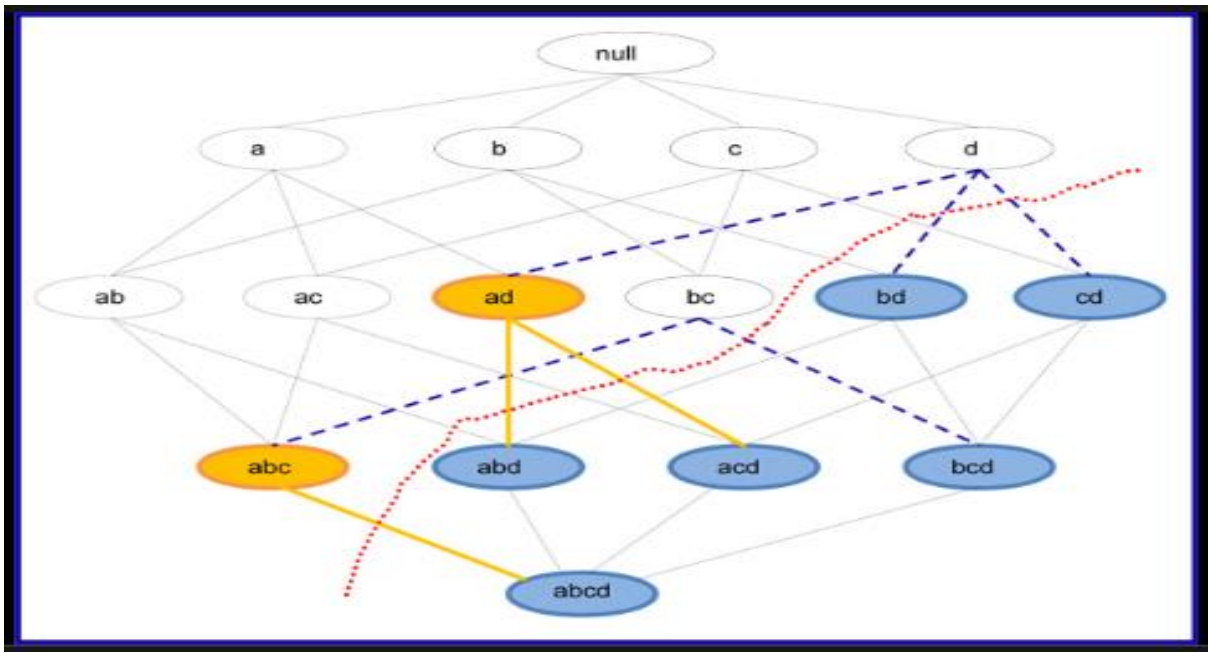
**Maximal Frequent Itemset**

***Definition***

- **It is a frequent itemset for which none of its immediate supersets are frequent.**

***Identification***

- **Examine the frequent itemsets that appear at the border between the infrequent and frequent itemsets.**
- **Identify all of its immediate supersets.**
- **If none of the immediate supersets are frequent, the itemset is maximal frequent.**



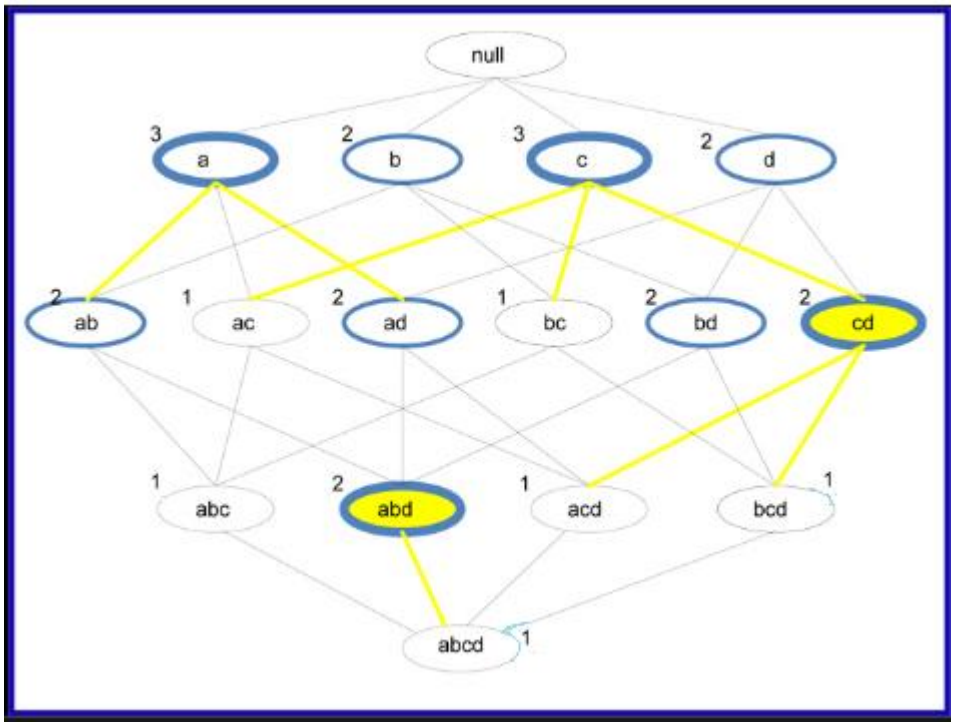

**Closed Frequent Itemset**

***Definition:***

- **An itemset is closed in a data set if there exists no superset that has the same support count as this original itemset**
- **It is a frequent itemset that is both closed and its support is greater than or equal to minsup..**

***Identification***

- First identify all frequent itemsets.
- Then from this group find those that are closed by checking to see if there exists a superset that has the same support as the frequent itemset, if there is, the itemset is disqualified, but if none can be found, the itemset is closed.




**6)Any Transaction data will be given. You have to apply apriori and FP growth algorithms.**

**Refer questions 3,4…**

**UNIT– 5**

1. **Define Cluster and importance of cluster analysis.**

**Cluster: a collection of data objects**

a. **Similar to one another within the same cluster**
b. **Dissimilar to the objects in other clusters**

Cluster Analysis is the process to find similar groups of objects in order to form clusters. It is an unsupervised machine learning-based algorithm that acts on unlabelled data. A group of data points would comprise together to form a cluster in which all the objects would belong to the same group.

# Importance Of Cluster Analysis:

- It is widely used in image processing, data analysis, and pattern recognition.
- It helps marketers to find the distinct groups in their customer base and they can characterize their customer groups by using purchasing patterns.
- It can be used in the field of biology, by deriving animal and plant taxonomies and identifying genes with the same capabilities.

- It also helps in information discovery by classifying documents on the web.

## 2)Types of Clustering.

**Partitioning Method:** It is used to make partitions on the data in order to form clusters. If "n" partitions are done on "p" objects of the database then each partition is represented by a cluster and n < p. The two conditions which need to be satisfied with this Partitioning Clustering Method are:

- One objective should only belong to only one group.
- There should be no group without even a single purpose.

**Hierarchical Method:** In this method, a hierarchical decomposition of the given set of data objects is created. We can classify hierarchical methods and will be able to know the purpose of classification on the basis of how the hierarchical decomposition is formed. There are two types of approaches for the creation of hierarchical decomposition, they are:

- **Agglomerative Approach**
- **Divisive Approach**

**Density-Based Method:** The density-based method mainly focuses on density. In this method, the given cluster will keep on growing continuously as long as the density in the neighbourhood exceeds some threshold, i.e, for each data point within a given cluster.

**Grid-Based Method:** In the Grid-Based method a grid is formed using the object together,i.e, the object space is quantized into a finite number of cells that form a grid structure. One of the major advantages of the grid-based method is fast processing time and it is dependent only on the number of cells in each dimension in the quantized space. The processing time for this method is much faster so it can save time.

**Model-Based Method:** In the model-based method, all the clusters are hypothesized in order to find the data which is best suited for the model. The clustering of the density function is used to locate the clusters for a given model

**Constraint-Based Method:** The constraint-based clustering method is performed by the incorporation of application or user-oriented constraints. A constraint refers to the user expectation or the properties of the desired clustering results. .

## 3)K- means clustering algorithm with an example.

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

# How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:

**Step-1:** Select the number K to decide the number of clusters.

**Step-2:** Select random K points or centroids. (It can be other from the input dataset).

**Step-3:** Assign each data point to their closest centroid, which will form the predefined K clusters.

**Step-4:** Calculate the variance and place a new centroid of each cluster.

**Step-5:** Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

**Step-6:** If any reassignment occurs, then go to step-4 else go to FINISH.

**Step-7**: The model is ready.

**Example**

**Suppose that the data mining task is to cluster the following eight points with (x, y) representing location) into three clusters.**

**A1(2, 10), A2(2, 5), A3(8, 4), B1(5, 8), B2(7, 5), B3(6, 4), C1(1, 2), C2(4, 9)**

**The distance function is Euclidean distance. Suppose initially we assign A1, B1 and C1 as the center of each cluster, respectively. Use the k-means algorithm to show only**

**i) The three cluster centers after the first round execution and**

**ii) The final three clusters.**

The resultant values and three cluster centers after the first round execution are shown in the tabular form.

Table 7.4.1 The Resultant Values and Center after 1st Round

| | Points | (2, 10) Mean1 | (5, 8) Mean2 | (1, 2) Mean3 | Cluster |
|---|---|---|---|---|---|
| A1 | (2, 10) | 0 | 3.60 | 8.06 | 1 |
| A2 | (2, 5) | 5 | 4.24 | 3.16 | 3 |
| A3 | (8, 4) | 8.48 | 5 | 7.28 | 2 |
| B1 | (5, 8) | 3.60 | 0 | 7.21 | 2 |
| B2 | (7, 5) | 7.07 | 3.60 | 6.70 | 2 |
| B3 | (6, 4) | 7.21 | 4.12 | 5.38 | 2 |
| C1 | (1, 2) | 8.06 | 7.21 | 0 | 3 |
| C2 | (4, 9) | 2.23 | 1.41 | 7.61 | 2 |

2) The final three clusters are,

| Cluster 1 | Cluster 2 | Cluster 3 |
|---|---|---|
| A1 (2, 10) | A3(8, 4) | A2(2, 5) |
| | B1(5, 8) | C1(1, 2) |
| | B2(7, 5) | |
| | B3(6, 4) | |
| | C2(4, 9) | |

New center of cluster1 = (2, 10)

New center of cluster 2 = $\left[\frac{(8+5+7+6+4)}{5}, \frac{(4+8+5+4+9)}{5}\right]$ = (6, 6)

New center of cluster 3 = $\left[\frac{(2+1)}{2}, \frac{(5+2)}{2}\right]$ = (1.5, 3.5).

**5)Example problems on K- means clustering.**

Cluster the following eight points (with (x, y) representing locations) into three clusters:

A1(2, 10), A2(2, 5), A3(8, 4), A4(5, 8), A5(7, 5), A6(6, 4), A7(1, 2), A8(4, 9)

Initial cluster centers are: A1(2, 10), A4(5, 8) and A7(1, 2).

The distance function between two points a = (x1, y1) and b = (x2, y2) is defined as-

$$P(a, b) = |x2 – x1| + |y2 – y1|$$

Use K-Means Algorithm to find the three cluster centers after the second iteration.

## Solution-

We follow the above discussed K-Means Clustering Algorithm-

### Iteration-01:

- We calculate the distance of each point from each of the center of the three clusters.
- The distance is calculated by using the given distance function.

The following illustration shows the calculation of distance between point A1(2, 10) and each of the center of the three clusters-

### Calculating Distance Between A1(2, 10) and C1(2, 10)-

P(A1, C1)

= |x2 – x1| + |y2 – y1|

= |2 – 2| + |10 – 10|

= 0

### Calculating Distance Between A1(2, 10) and C2(5, 8)-

P(A1, C2)

= |x2 – x1| + |y2 – y1|

= |5 – 2| + |8 – 10|

= 3 + 2

= 5

### Calculating Distance Between A1(2, 10) and C3(1, 2)-

P(A1, C3)

= |x2 – x1| + |y2 – y1|

= |1 – 2| + |2 – 10|

= 1 + 8

= 9

In the similar manner, we calculate the distance of other points from each of the center of the three clusters.

Next,

- We draw a table showing all the results.
- Using the table, we decide which point belongs to which cluster.
- The given point belongs to that cluster whose center is nearest to it.

| Given Points | Distance from center (2, 10) of Cluster-01 | Distance from center (5, 8) of Cluster-02 | Distance from center (1, 2) of Cluster-03 | Point belongs to Cluster |
|---|---|---|---|---|
| A1(2, 10) | 0 | 5 | 9 | C1 |
| A2(2, 5) | 5 | 6 | 4 | C3 |
| A3(8, 4) | 12 | 7 | 9 | C2 |
| A4(5, 8) | 5 | 0 | 10 | C2 |
| A5(7, 5) | 10 | 5 | 9 | C2 |
| A6(6, 4) | 10 | 5 | 7 | C2 |
| A7(1, 2) | 9 | 10 | 0 | C3 |

| | | | | |
|---|---|---|---|---|
| A8(4, 9) | 3 | 2 | 10 | C2 |

From here, New clusters are-

*Cluster-01:*

First cluster contains points-

- A1(2, 10)

*Cluster-02:*

Second cluster contains points-

- A3(8, 4)
- A4(5, 8)
- A5(7, 5)
- A6(6, 4)
- A8(4, 9)

*Cluster-03:*

Third cluster contains points-

- A2(2, 5)
- A7(1, 2)

Now,

- We re-compute the new cluster clusters.
- The new cluster center is computed by taking mean of all the points contained in that cluster.

*For Cluster-01:*

- We have only one point A1(2, 10) in Cluster-01.
- So, cluster center remains the same.

*For Cluster-02:*

Center of Cluster-02

= ((8 + 5 + 7 + 6 + 4)/5, (4 + 8 + 5 + 4 + 9)/5)

= (6, 6)

*For Cluster-03:*

Center of Cluster-03

= ((2 + 1)/2, (5 + 2)/2)

= (1.5, 3.5)

This is completion of Iteration-01.

## Iteration-02:

- We calculate the distance of each point from each of the center of the three clusters.
- The distance is calculated by using the given distance function.

The following illustration shows the calculation of distance between point A1(2, 10) and each of the center of the three clusters-

### Calculating Distance Between A1(2, 10) and C1(2, 10)-

P(A1, C1)

= |x2 – x1| + |y2 – y1|

= |2 – 2| + |10 – 10|

= 0

### Calculating Distance Between A1(2, 10) and C2(6, 6)-

P(A1, C2)

= |x2 – x1| + |y2 – y1|

= |6 – 2| + |6 – 10|

= 4 + 4

= 8

Calculating Distance Between A1(2, 10) and C3(1.5, 3.5)-

P(A1, C3)

= |x2 – x1| + |y2 – y1|

= |1.5 – 2| + |3.5 – 10|

= 0.5 + 6.5

= 7

In the similar manner, we calculate the distance of other points from each of the center of the three clusters.

Next,

- We draw a table showing all the results.
- Using the table, we decide which point belongs to which cluster.
- The given point belongs to that cluster whose center is nearest to it.

| Given Points | Distance from center (2, 10) of Cluster-01 | Distance from center (6, 6) of Cluster-02 | Distance from center (1.5, 3.5) of Cluster-03 | Point belongs to Cluster |
|---|---|---|---|---|
| A1(2, 10) | 0 | 8 | 7 | C1 |
| A2(2, 5) | 5 | 5 | 2 | C3 |
| A3(8, 4) | 12 | 4 | 7 | C2 |
| A4(5, 8) | 5 | 3 | 8 | C2 |

| | | | | |
|---|---|---|---|---|
| A5(7, 5) | 10 | 2 | 7 | C2 |
| A6(6, 4) | 10 | 2 | 5 | C2 |
| A7(1, 2) | 9 | 9 | 2 | C3 |
| A8(4, 9) | 3 | 5 | 8 | C1 |

From here, New clusters are-

*Cluster-01:*

First cluster contains points-

- A1(2, 10)
- A8(4, 9)

*Cluster-02:*

Second cluster contains points-

- A3(8, 4)
- A4(5, 8)
- A5(7, 5)
- A6(6, 4)

*Cluster-03:*

Third cluster contains points-

- A2(2, 5)
- A7(1, 2)

Now,

- We re-compute the new cluster clusters.
- The new cluster center is computed by taking mean of all the points contained in that cluster.

*For Cluster-01:*

Center of Cluster-01

= ((2 + 4)/2, (10 + 9)/2)

= (3, 9.5)

*For Cluster-02:*

Center of Cluster-02

= ((8 + 5 + 7 + 6)/4, (4 + 8 + 5 + 4)/4)

= (6.5, 5.25)

*For Cluster-03:*

Center of Cluster-03

= ((2 + 1)/2, (5 + 2)/2)

= (1.5, 3.5)

This is completion of Iteration-02.

After second iteration, the center of the three clusters are-

- C1(3, 9.5)
- C2(6.5, 5.25)
- C3(1.5, 3.5)

**6)Explain Bi-Secting K Means with example**

Bisecting K-Means Algorithm is a modification of the K-Means algorithm. It is a hybrid approach between partitional and hierarchical clustering. It can recognize clusters of any shape and size.

1: Initialize the list of clusters to contain the cluster containing all points.
2: **repeat**
3: Select a cluster from the list of clusters
4: **for** $i = 1$ to *number_of_iterations* **do**
5: Bisect the selected cluster using basic K-means
6: **end for**
7: Add the two clusters from the bisection with the lowest SSE to the list of clusters.
8: **until** Until the list of clusters contains $K$ clusters

### Bisecting K-means Example