Unit 3: Classification LH 7

1. Basics and Algorithms
2. Decision Tree Classifier
3. Rule Based Classifier
4. Nearest Neighbor Classifier
5. Bayesian Classifier
6. Artificial Neural Network Classifier
7. Issues : Overfitting, Validation, Model Comparison

Supervised Learning

- Supervised learning is a type of machine learning that uses labeled data to train machine learning models.
- In labeled data, the output is already known. In Supervised Learning, the machine learns under supervision.
- It contains a model that is able to predict with the help of a labeled dataset.

Supervised learning can be further divided into two types:

1. Classification
2. Regression

Unsupervised Learning

- Unsupervised learning is a type of machine learning that uses unlabeled data to train machines.
- Unlabeled data doesn’t have a fixed output variable.
- The model learns from the data, discovers the patterns and features in the data, and returns the output.
- Unsupervised learning finds patterns and understands the trends in the data to discover the output. So, the model tries to label the data based on the features of the input data.
- Unsupervised learning can be further divided into two types:
  - Clustering
  - Association

Three classes of learning problems

Supervised Learning

Data: (x, y)
x is an input data, y is a label (e.g. photo with label "cat")

Goal: Learn to map input to output i.e. $x \rightarrow y$

An example: to classify

This is a cat

Unsupervised Learning

Data: x
x is data, there's no labels!

Goal: Learn an underlying structure of the data.

An example: Comparison

The two things are alike

Reinforcement Learning

Data: No data, Only state-action pairs (s, a).

Goal: Maximize future reward over many time steps.

An example: reward = joy

Interaction with the cat gives joy

Classification

Classification is the process where a model or classifier is constructed to predict categorical label of unknown data.

- It is to classify whether data belongs to a known group or object class.
- Models will assign a class label to the data it processes, which is learned by the algorithm through training on labelled training data.
- The input and output of the data has been labelled, so the model can understand which features will classify an object or data point with different class labels.

Mainly two classification problem:

- Binary classification
- Multiple class classification

Binary Classification	Multiple class Classification
Binary classification is when a model can apply only two class labels.	Multiple class classification is when models reference more than the two class labels found in binary classification.
A popular use of a binary classification would be in detecting and filtering junk emails	Instead, there could be a huge array of possible class labels that could be applied to the object or data.
A model can be trained to label incoming emails as either junk or safe, based on learned patterns of what constitutes a spam email.	An example would be in facial recognition software, where a model may analyze an image against a huge range of possible class labels to identify the individual.
Binary classification is commonly performed by algorithms such as: • Logistic Regression • Decision Trees • Naïve Bayes	Multiple class classification is commonly performed by algorithms such as: • Random Forest • k-Nearest Neighbors • Naive Bayes

Data classification is a Three-step process:

Model construction

- In the first step, a classifier is built describing a predetermined set of data classes or concepts. This is the learning step (or training phase), where a classification algorithm builds the classifier by analyzing or "learning from" a training set made up of database tuples and their associated class labels.

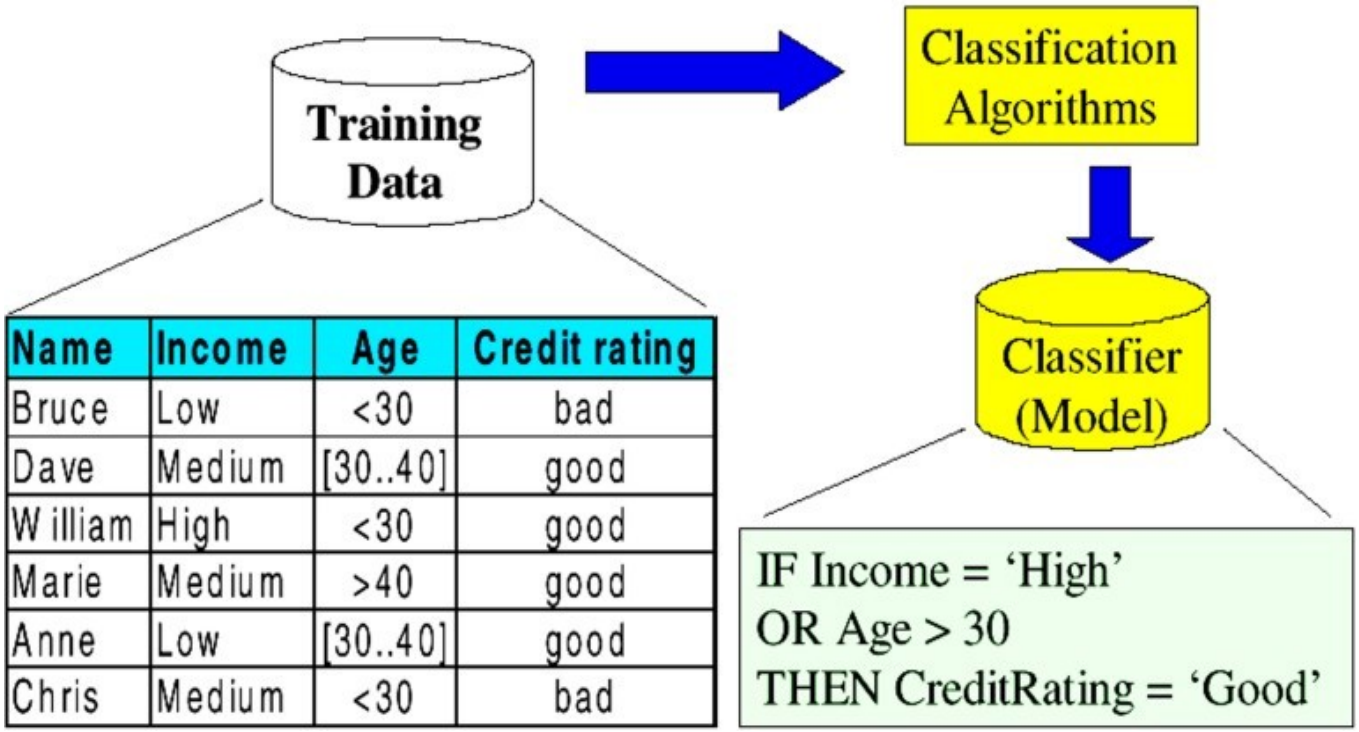
Model Evaluation:

- Based on the test dataset, the accuracy rate of the model is evaluated.

Model usage

- The model is used to classify unseen instances (i.e. to predict the class labels for new unclassified instances.

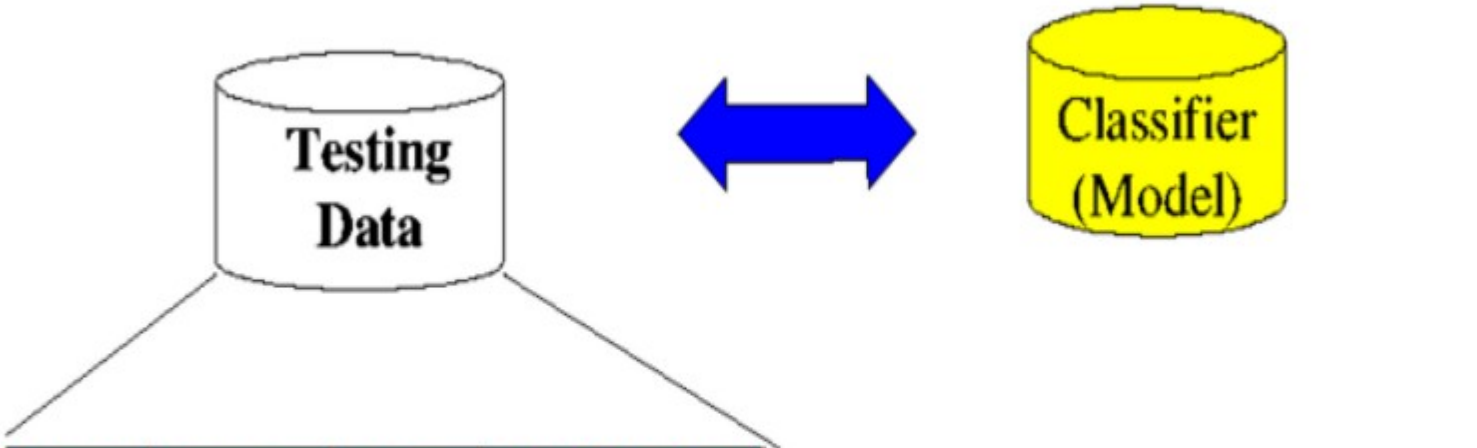
Model Construction



Name	Income	Age	Credit rating
Bruce	Low	<30	bad
Dave	Medium	[30..40]	good
William	High	<30	good
Marie	Medium	>40	good
Anne	Low	[30..40]	good
Chris	Medium	<30	bad

Fig: Model Construction and Usage

Model Evaluation

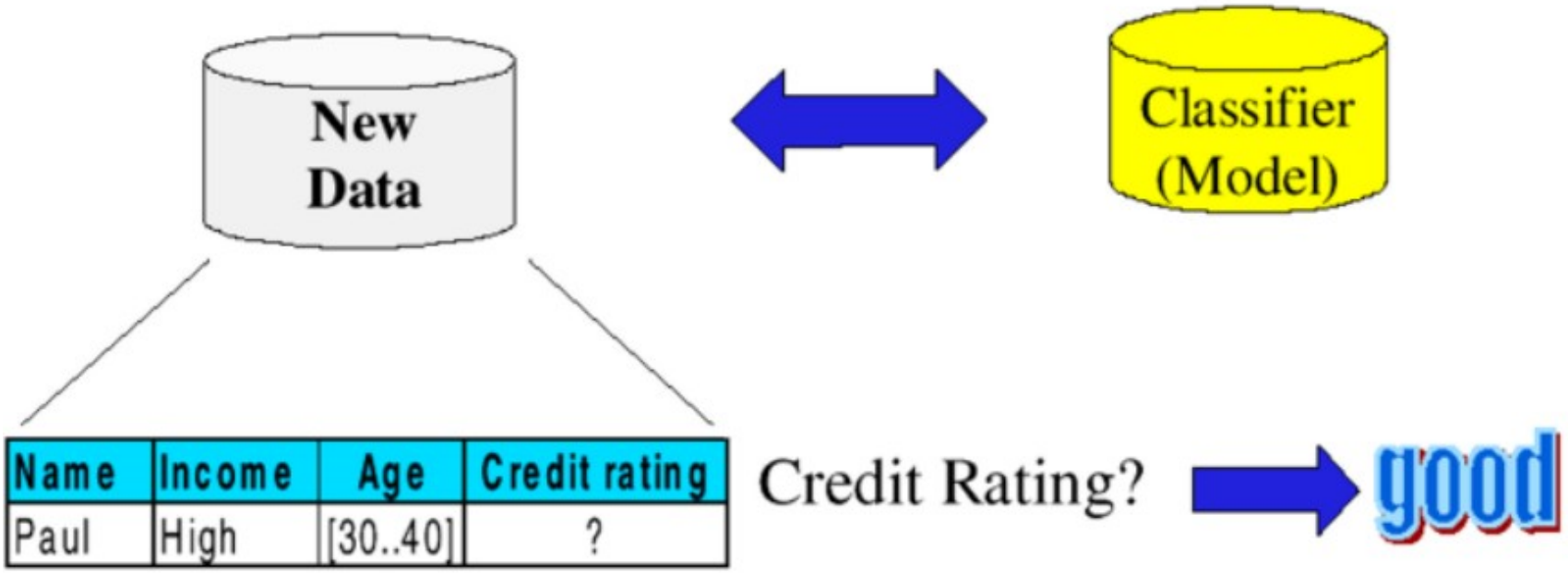


Name	Income	Age	Credit rating
Tom	Medium	<30	bad
Jane	High	<30	bad
Wei	High	>40	good
Hua	Medium	[30..40]	good

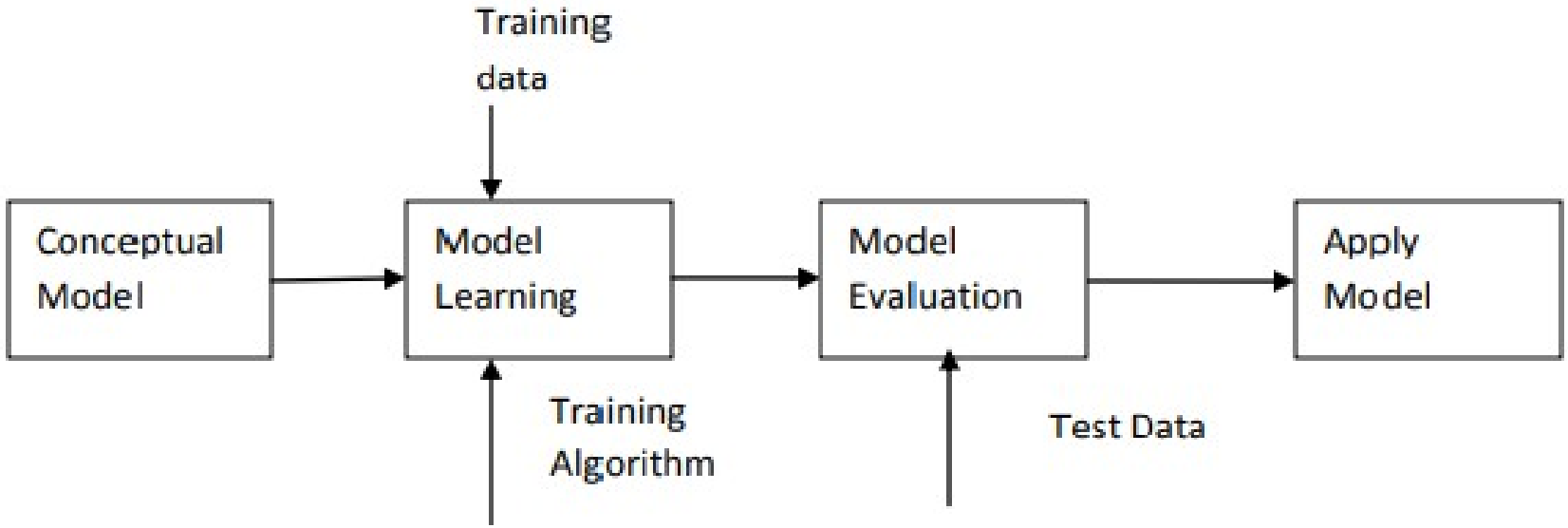
How accurate is the model?

IF Income = 'High'
OR Age > 30
THEN CreditRating = 'Good'

Model Use: Classification



Stage in Classification



Issues Regarding Classification

Preparing the Data for Classification

- Data cleaning: This refers to the preprocessing of data in order to remove or reduce noise (by applying smoothing techniques, for example) and the treatment of missing values (e.g., by replacing a missing value with the most commonly occurring value for that attribute, or with the most probable value based on statistics).

- Relevance analysis: Many of the attributes in the data may be redundant. Correlation analysis can be used to identify whether any two given attributes are statistically related. For example, a strong correlation between attributes A1 and A2 would suggest that one of the two could be removed from further analysis.

- Data transformation and reduction: The data may be transformed by normalization, particularly when neural networks or methods involving distance measurements are used in the learning step. Normalization involves scaling all values for a given attribute so that they fall within a small specified range, such as -1.0 to 1.0, or 0.0 to 1.0. Data can also be reduced by applying methods such as binning, histogram analysis, and clustering.

Evaluating Classification Methods

Classification methods can be compared and evaluated according to the following criteria:

- **Accuracy**: The accuracy of a classifier refers to the ability of a given classifier to correctly predict the class label of new or previously unseen data (i.e., tuples without class label information).

- **Speed**: This refers to the computational costs involved in generating and using the given classifier or predictor. Robustness: This is the ability of the classifier or predictor to make correct predictions given noisy data or data with missing values.

- **Scalability:** This refers to the ability to construct the classifier or predictor efficiently given large amounts of data.

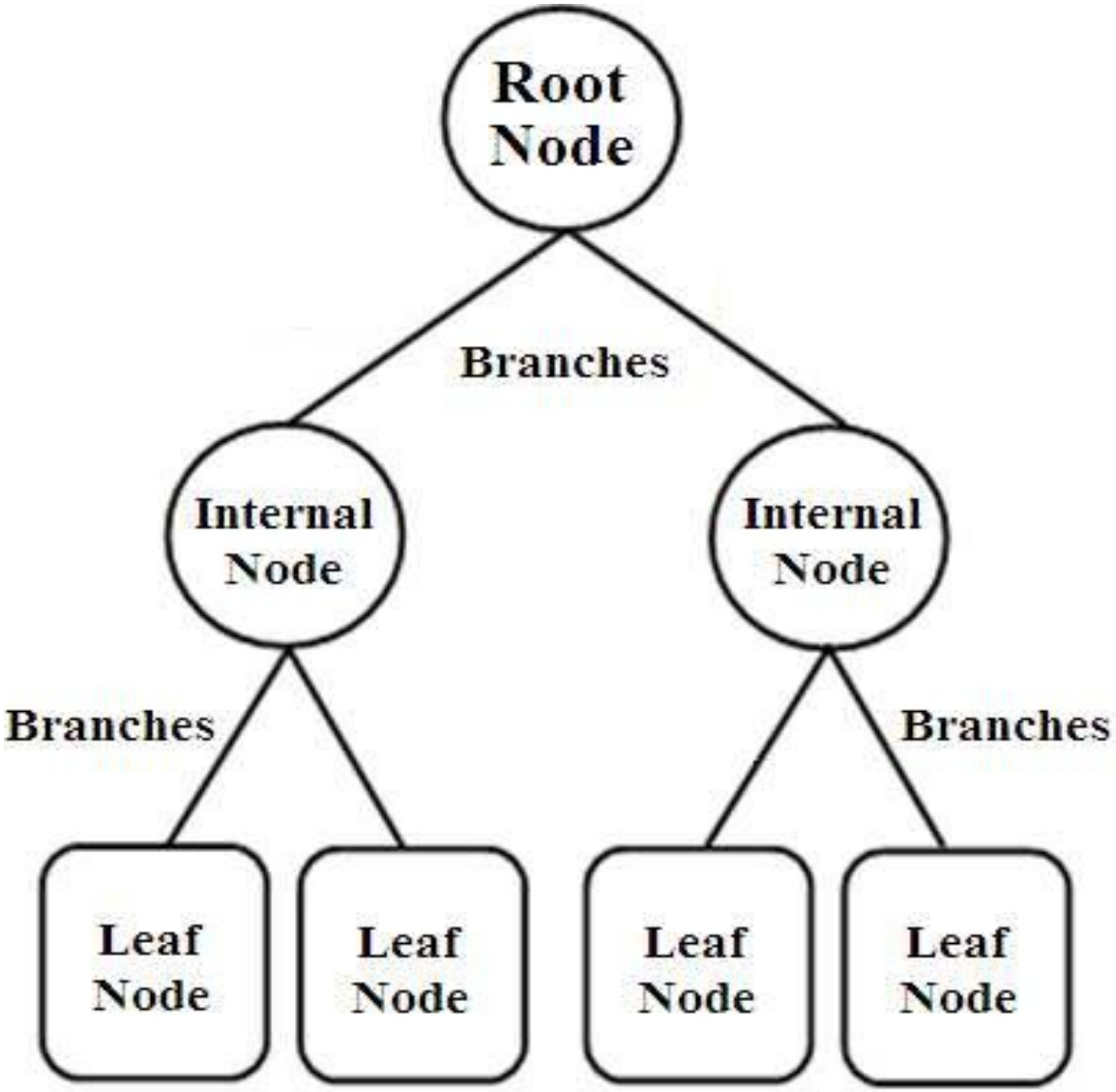
Types of classifier

i. Decision Tree classifier
ii. Rule Based Classifier
iii. Nearest Neighbor Classifier
iv. Bayesian Classifier
v. Artificial Neural Network (ANN) Classifier
vi. Others

Decision Tree Classifier

2. Decision Tree

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.
- It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
- Decision Tree learning is one of the most widely used and practical methods for inductive inference over supervised data.
- A decision tree represents a procedure for classifying categorical data based on their attributes.
- It is also efficient for processing large amount of data, so is often used in data mining application..
- The construction of decision tree does not require any domain knowledge or parameter setting, and therefore appropriate for exploratory knowledge discovery.
- Their representation of acquired knowledge in tree form is intuitive and easy to assimilate by humans


Root Node
Branches
Internal Node
Internal Node
Branches
Branches
Leaf Node
Leaf Node
Leaf Node
Leaf Node

(a)

Why use Decision Trees?

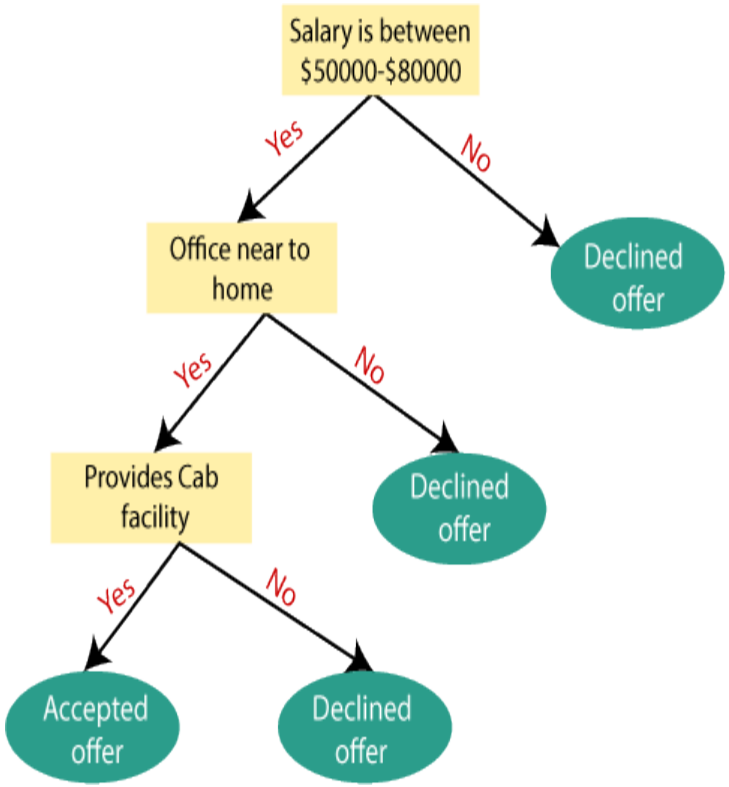
- Below are the two reasons for using the Decision tree:
  - Decision Trees **usually mimic human thinking** ability while making a decision, so it is easy to understand.
  - The logic behind the decision tree **can be easily understood** because it shows a tree-like structure.

Decision Tree Terminologies

- **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node
- **Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch/Sub Tree:** A tree formed by splitting the tree.
- **Pruning:** Pruning is the process of removing the unwanted branches from the tree.
- **Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

How does the Decision Tree algorithm Work?

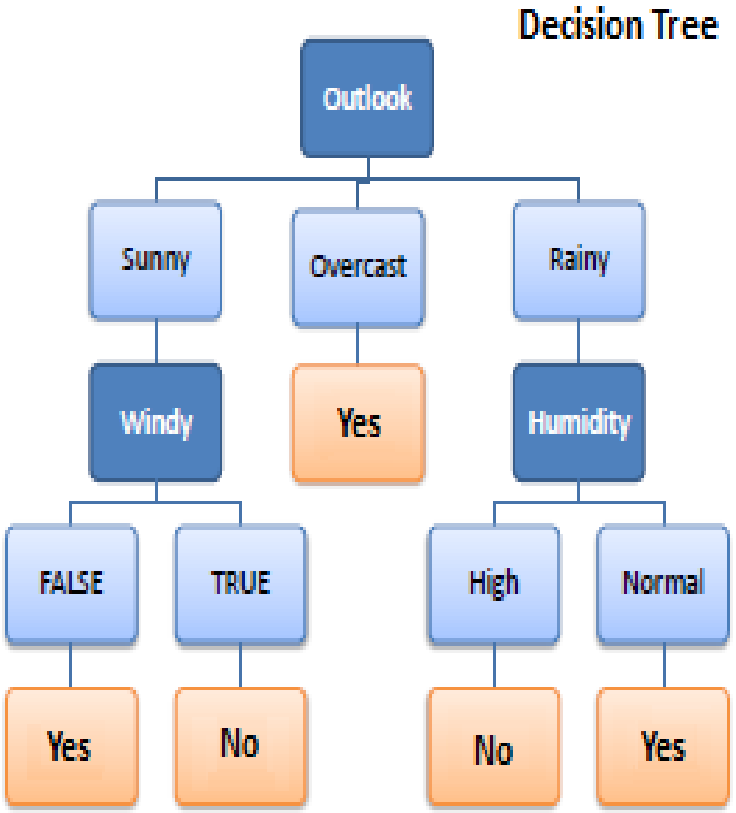
- In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.
- For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree.



2. Decision Tree

Predictors | Target

Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No



- The construction of decision tree classifiers does not require any domain knowledge or parameter setting.
- Decision trees can handle high dimensional data.
- Their representation of acquired knowledge in tree form is intuitive and generally easy to assimilate by humans.
- The learning and classification steps of decision tree induction are simple and fast. In general, decision tree classifiers have good accuracy.
- Decision tree induction algorithms have been used for classification in many application areas, such as medicine, manufacturing and production, financial analysis, astronomy, and molecular biology.

Decision Tree Algorithm

1. Hunt's Algorithm
2. ID3, J48, C4.5 (Based on Entropy Calculation)
3. SLIQ,SPRINT,CART (Based on Gini-Index)

Hunt’s Algorithm

- Hunt's algorithm grows a decision tree in **a recursive fashion by partitioning the training data into successively into subsets**.

- Let Dt: be the set of training data.
- t: Node
- Y: class label {y1, y2,…. Yn}

The general recursive procedure is defined as:

1. If Dt contains records that belong the same class yt , then t is a leaf node labeled as yt.
2. If Dt is an empty set, then t is a leaf node labeled by the default class, yd
3. If Dt contains records that belong to more than one class, use **an attribute test** to split the data into smaller subsets.

- It **recursively** applies the procedure to each **subset until** all the records in the subset belong to the **same class.**

- The Hunt's algorithm assumes that each combination of attribute sets has a unique class label during the procedure.

- If all the records associated with Dt have identical attribute values for the class label, then it is not possible to split these records any future.

Tid	Refund	Marital Status	Taxable Income	Cheat
		categorical	categorical	continuous	class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data

Model: Decision Tree

Tree Induction:

- Tree induction is based on Greedy Strategy i.e. that optimize certain criteria.

Types of Nodes:

- Root Node : Main Question
- Branch Node : Intermediate Process
- Leaf Node : Answer

Attribute Selection Measures:

Entropy:

- Amount of uncertainty in the info.
- As Information gain increase, Entropy Decrease.

Information Gain

- How much information (Accuracy) does the answer to the specific question provide.

Issues

1. How to split the record?

- Tree is constructed in a **top-down recursive divide-and-conquer** manner
- At start, all the training examples are at the root
- Attributes are categorical (if continuous-valued, they are discretized in advance)
- Nodes with **homogenous** class distribution are preferred

- **2. How to specify the attribute test condition?**
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)

  3. When to stop splitting?

  When all records are belongs to the same class or all records have similar attributes.

- **4. How to determine the best split?**

  Nodes with homogenous class distribution are preferred.

  - Measure the node impurity.
    - i. Gini-Index
    - ii. Entropy (Information gain)
    - iii.Misclassification Error

Attribute Selection Measures

- While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes.
- So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM.**
- By this measurement, we can easily select the best attribute for the nodes of the tree.
- There are three popular techniques for ASM, which are:
  1. **Information Gain**
  2. **Gain ratio**
  3. **Gini Index**

Entropy

Entropy H(S) is a measure of the amount of uncertainty in the dataset (S) (i.e. entropy characterizes the dataset (S)).

$$H(S) = -\sum_{x \in X} p(x) \log_2 p(x)$$

Where,

- S The current dataset for which entropy is being calculated (changes every iteration of the ID3 algorithm)
- X - Set of classes in S
- P(x) - The probability of each set S

- When H(S) = 0, the set S is perfectly classified (i.e. all elements in S are of the same class).
- In ID3, entropy is calculated for each remaining attribute. The attribute with the smallest entropy is used to split the set S on this iteration.
- The higher the entropy, the higher the potential to improve the classification here.

Note

Entropy ([All attribute in same class]) = 0 (i.e Entropy of (14, 0))

Entropy ([Attribute distribute equally]) = 1 (i.e Entropy of (7, 7))

Entropy ([Attribute distribute Unequally]) = i.e Entropy (9,5)

$-[9/14 \log_2(9/14)+5/14 \log_2(5/14)] = 0.94$

1. Information Gain:

- Information gain is the **measurement of changes in entropy** after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, **we split the node and build the decision tree.**
- A decision tree algorithm **always tries to maximize the value of information gain,** and a node/attribute having **the highest information gain is split first.**
- Information gain is used by ID3 algorithm.
- It can be calculated using the below formula:

  Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)

Information Gain

- It helps to reduction in entropy caused by portioning the examples according to an attribute. (Helps to reduce impurity).

$$IG(A,S) = H(S) - \sum_{t \in T} p(t) H(t)$$

Where,

- H(S) - Entropy of dataset S
- T - The subsets created from splitting dataset S by attribute A.
- P(t) - The probability of class t
- H(t) - Entropy of subset t

Decision Tree: Construction

Solved Example 1:

1. What is the entropy of this collection of training examples with respect to the target function Play Tennis?
2. What is the information gain of Outlook, Temp, Humidity and Wind relative to these training examples?
3. Draw decision tree for the given dataset using ID3 algorithm.

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

Reference video:
https://www.youtube.com/watch?v=coOTEc-0OGw

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

Attribute: Outlook

$Values(Outlook) = Sunny, Overcast, Rain$

$S = [9+, 5-]$ $\quad Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$

$S_{Sunny} \leftarrow [2+, 3-]$ $\quad Entropy(S_{Sunny}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.971$

$S_{Overcast} \leftarrow [4+, 0-]$ $\quad Entropy(S_{Overcast}) = -\frac{4}{4}log_2\frac{4}{4} - \frac{0}{4}log_2\frac{0}{4} = 0$

$S_{Rain} \leftarrow [3+, 2-]$ $\quad Entropy(S_{Rain}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.971$

$$Gain(S, Outlook) = Entropy(S) - \sum_{v \in \{Sunny, Overcast, Rain\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Outlook)$$

$$= Entropy(S) - \frac{5}{14}Entropy(S_{Sunny}) - \frac{4}{14}Entropy(S_{Overcast})$$

$$- \frac{5}{14}Entropy(S_{Rain})$$

$$Gain(S, Outlook) = 0.94 - \frac{5}{14}0.971 - \frac{4}{14}0 - \frac{5}{14}0.971 = 0.2464$$

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

Attribute: Temp

$Values(Temp) = Hot, Mild, Cool$

$S = [9+, 5-]$ $\quad Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$

$S_{Hot} \leftarrow [2+, 2-]$ $\quad Entropy(S_{Hot}) = -\frac{2}{4}log_2\frac{2}{4} - \frac{2}{4}log_2\frac{2}{4} = 1.0$

$S_{Mild} \leftarrow [4+, 2-]$ $\quad Entropy(S_{Mild}) = -\frac{4}{6}log_2\frac{4}{6} - \frac{2}{6}log_2\frac{2}{6} = 0.9183$

$S_{Cool} \leftarrow [3+, 1-]$ $\quad Entropy(S_{Cool}) = -\frac{3}{4}log_2\frac{3}{4} - \frac{1}{4}log_2\frac{1}{4} = 0.8113$

$$Gain(S, Temp) = Entropy(S) - \sum_{v \in \{Hot, Mild, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Temp) = Entropy(S) - \frac{4}{14}Entropy(S_{Hot}) - \frac{6}{14}Entropy(S_{Mild}) - \frac{4}{14}Entropy(S_{Cool})$$

$$Gain(S, Temp) = 0.94 - \frac{4}{14}1.0 - \frac{6}{14}0.9183 - \frac{4}{14}0.8113 = 0.0289$$

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

Attribute: Humidity

$$Values\,(Humidity) = High, Normal$$

$$S = [9+, 5-] \qquad Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$$

$$S_{High} \leftarrow [3+, 4-] \qquad Entropy(S_{High}) = -\frac{3}{7}log_2\frac{3}{7} - \frac{4}{7}log_2\frac{4}{7} = 0.9852$$

$$S_{Normal} \leftarrow [6+, 1-] \qquad Entropy(S_{Normal}) = -\frac{6}{7}log_2\frac{6}{7} - \frac{1}{7}log_2\frac{1}{7} = 0.5916$$

$$Gain\,(S, Humidity) = Entropy(S) - \sum_{v \in (High, Normal)} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Humidity)$$

$$= Entropy(S) - \frac{7}{14}Entropy(S_{High}) - \frac{7}{14}Entropy(S_{Normal})$$

$$Gain(S, Humidity) = 0.94 - \frac{7}{14}0.9852 - \frac{7}{14}0.5916 = 0.1516$$

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

Attribute: Wind

$Values\,(Wind) = Strong, Weak$

$S = [9+, 5-]$ $\qquad Entropy(S) = -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} = 0.94$

$S_{Strong} \leftarrow [3+, 3-]$ $\qquad Entropy(S_{Strong}) = 1.0$

$S_{Weak} \leftarrow [6+, 2-]$ $\qquad Entropy(S_{Weak}) = -\frac{6}{8}log_2\frac{6}{8} - \frac{2}{8}log_2\frac{2}{8} = 0.8113$

$$Gain\,(S, Wind) = Entropy(S) - \sum_{v \in \{Strong, Weak\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, Wind) = Entropy(S) - \frac{6}{14}Entropy(S_{Strong}) - \frac{8}{14}Entropy(S_{Weak})$$

$$Gain(S, Wind) = 0.94 - \frac{6}{14}\,1.0 - \frac{8}{14}\,0.8113 = 0.0478$$

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

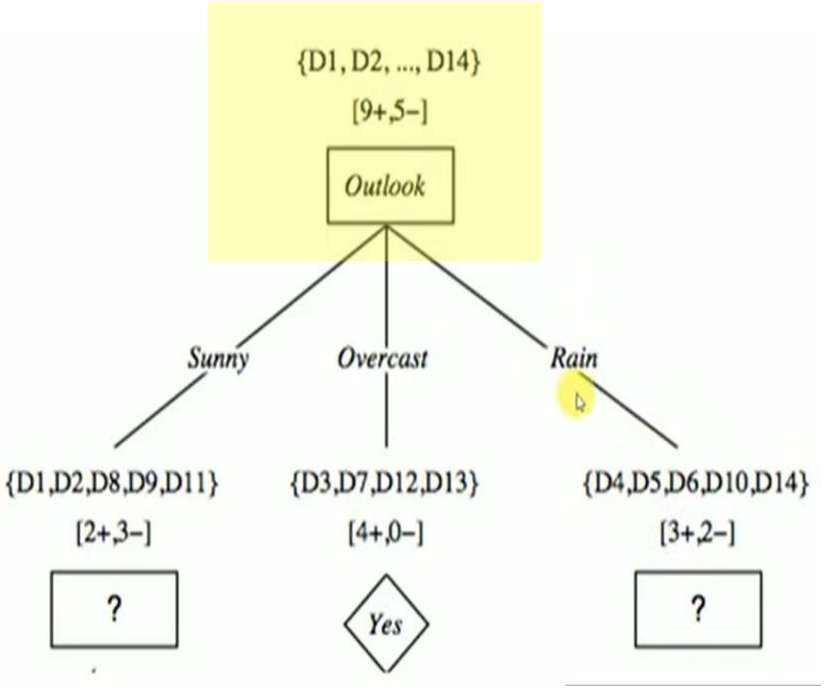
$Gain(S, Outlook) = 0.2464$

$Gain(S, Temp) = 0.0289$

$Gain(S, Humidity) = 0.1516$

$Gain(S, Wind) = 0.0478$

This has the highest Information gain



Day	Temp	Humidity	Wind	Play Tennis
D1	Hot	High	Weak	No
D2	Hot	High	Strong	No
D8	Mild	High	Weak	No
D9	Cool	Normal	Weak	Yes
D11	Mild	Normal	Strong	Yes

Attribute: Temp

$Values(Temp) = Hot, Mild, Cool$

$S_{Sunny} = [2+, 3-]$ $\quad Entropy(S_{Sunny}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.97$

$S_{Hot} \leftarrow [0+, 2-]$ $\quad Entropy(S_{Hot}) = 0.0$

$S_{Mild} \leftarrow [1+, 1-]$ $\quad Entropy(S_{Mild}) = 1.0$

$S_{Cool} \leftarrow [1+, 0-]$ $\quad Entropy(S_{Cool}) = 0.0$

$$Gain\,(S_{Sunny}, Temp) = Entropy(S) - \sum_{v \in \{Hot, Mild, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Sunny}, Temp)$$

$$= Entropy(S) - \frac{2}{5}Entropy(S_{Hot}) - \frac{2}{5}Entropy(S_{Mild})$$

$$- \frac{1}{5}Entropy(S_{Cool})$$

$$Gain(S_{sunny}, Temp) = 0.97 - \frac{2}{5}0.0 - \frac{2}{5}1 - \frac{1}{5}0.0 = 0.570$$

Day	Temp	Humidity	Wind	Play Tennis
Dl	Hot	High	Weak	No
D2	Hot	High	Strong	No
D8	Mild	High	Weak	No
D9	Cool	Normal	Weak	Yes
Dl1	Mild	Normal	Strong	Yes

Attribute: Humidity

$Values(Humidity) = High, Normal$

$S_{Sunny} = [2+, 3-]$ $\quad Entropy(S) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.97$

$S_{high} \leftarrow [0+, 3-]$ $\quad Entropy(S_{High}) = 0.0$

$S_{Normal} \leftarrow [2+, 0-]$ $\quad Entropy(S_{Normal}) = 0.0$

$$Gain\,(S_{Sunny}, Humidity) = Entropy(S) - \sum_{v \in (High, Normal)} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Sunny}, Humidity) = Entropy(S) - \frac{3}{5}Entropy(S_{High}) - \frac{2}{5}Entropy(S_{Normal})$$

$$Gain(S_{sunny}, Humidity) = 0.97 - \frac{3}{5}0.0 - \frac{2}{5}0.0 = 0.97$$

Day	Temp	Humidity	Wind	Play Tennis
Dl	Hot	High	Weak	No
D2	Hot	High	Strong	No
D8	Mild	High	Weak	No
D9	Cool	Normal	Weak	Yes
Dl1	Mild	Normal	Strong	Yes

Attribute: Wind

$Values(Wind) = Strong, Weak$

$S_{Sunny} = [2+, 3-]$ $\quad Entropy(S) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.97$

$S_{Strong} \leftarrow [1+, 1-]$ $\quad Entropy(S_{Strong}) = 1.0$

$S_{Weak} \leftarrow [1+, 2-]$ $\quad Entropy(S_{Weak}) = -\frac{1}{3}log_2\frac{1}{3} - \frac{2}{3}log_2\frac{2}{3} = 0.9183$

$$Gain\,(S_{Sunny}, Wind) = Entropy(S) - \sum_{v \in (Strong, Weak)} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Sunny}, Wind) = Entropy(S) - \frac{2}{5}Entropy(S_{Strong}) - \frac{3}{5}Entropy(S_{Weak})$$

$$Gain(S_{sunny}, Wind) = 0.97 - \frac{2}{5}1.0 - \frac{3}{5}0.918 = 0.0192$$

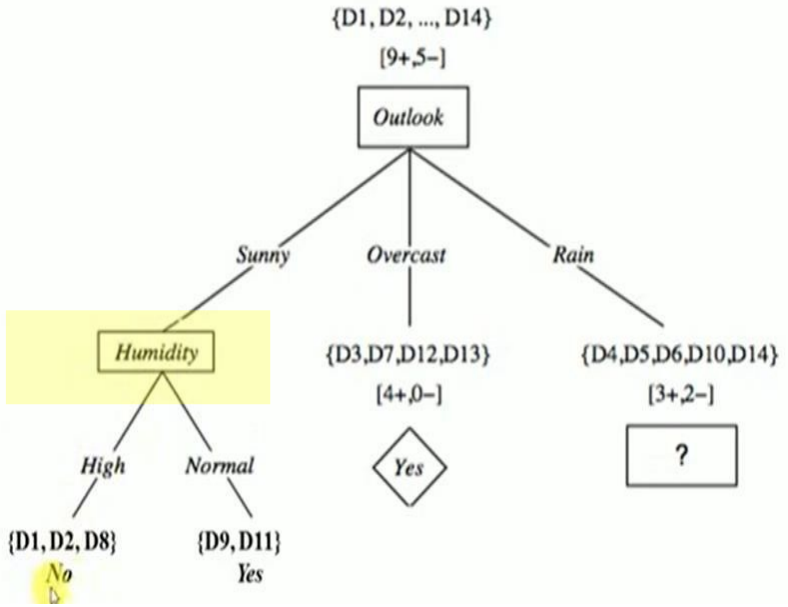
Day	Temp	Humidity	Wind	Play Tennis
D1	Hot	High	Weak	No
D2	Hot	High	Strong	No
D8	Mild	High	Weak	No
D9	Cool	Normal	Weak	Yes
D11	Mild	Normal	Strong	Yes

$$Gain(S_{sunny}, Temp) = 0.570$$

$$Gain(S_{sunny}, Humidity) = 0.97$$

$$Gain(S_{sunny}, Wind) = 0.0192$$

This has the highest Information gain



Day	Temp	Humidity	Wind	Play Tennis
D4	Mild	High	Weak	Yes
D5	Cool	Normal	Weak	Yes
D6	Cool	Normal	Strong	No
D10	Mild	Normal	Weak	Yes
D14	Mild	High	Strong	No

Attribute: Temp

$Values\,(Temp) = Hot, Mild, Cool$

$S_{Rain} = [3+, 2-]$ $\quad Entropy(S_{Sunny}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.97$

$S_{Hot} \leftarrow [0+, 0-]$ $\quad Entropy(S_{Hot}) = 0.0$

$S_{Mild} \leftarrow [2+, 1-]$ $\quad Entropy(S_{Mild}) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3} = 0.9183$

$S_{Cool} \leftarrow [1+, 1-]$ $\quad Entropy(S_{Cool}) = 1.0$

$$Gain\,(S_{Rain}, Temp) = Entropy(S) - \sum_{v \in (Hot, Mild, Cool)} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Rain}, Temp)$$

$$= Entropy(S) - \frac{0}{5}Entropy(S_{Hot}) - \frac{3}{5}Entropy(S_{Mild})$$

$$- \frac{2}{5}Entropy(S_{Cool})$$

$$Gain(S_{Rain}, Temp) = 0.97 - \frac{0}{5}0.0 - \frac{3}{5}0.918 - \frac{2}{5}1.0 = 0.0192$$

Day	Temp	Humidity	Wind	Play Tennis
D4	Mild	High	Weak	Yes
D5	Cool	Normal	Weak	Yes
D6	Cool	Normal	Strong	No
DI0	Mild	Normal	Weak	Yes
DI4	Mild	High	Strong	No

Attribute: Humidity

$Values(Humidity) = High, Normal$

$S_{Rain} = [3+, 2-]$ $\quad Entropy(S_{Sunny}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.97$

$S_{High} \leftarrow [1+, 1-]$ $\quad Entropy(S_{High}) = 1.0$

$S_{Normal} \leftarrow [2+, 1-]$ $\quad Entropy(S_{Normal}) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3} = 0.9183$

$$Gain\,(S_{Rain}, Humidity) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Rain}, Humidity) = Entropy(S) - \frac{2}{5}Entropy(S_{High}) - \frac{3}{5}Entropy(S_{Normal})$$

$$Gain(S_{Rain}, Humidity) = 0.97 - \frac{2}{5}\,1.0 - \frac{3}{5}0.918 = 0.0192$$

Day	Temp	Humidity	Wind	Play Tennis
D4	Mild	High	Weak	Yes
D5	Cool	Normal	Weak	Yes
D6	Cool	Normal	Strong	No
D10	Mild	Normal	Weak	Yes
D14	Mild	High	Strong	No

Attribute: Wind

$Values(wind) = Strong, Weak$

$S_{Rain} = [3+, 2-]$ $\quad Entropy(S_{Sunny}) = -\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5} = 0.97$

$S_{Strong} \leftarrow [0+, 2-]$ $\quad Entropy(S_{Strong}) = 0.0$

$S_{Weak} \leftarrow [3+, 0-]$ $\quad Entropy(S_{weak}) = 0.0$

$$Gain\,(S_{Rain}, Wind) = Entropy(S) - \sum_{v \in (Strong, Weak)} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S_{Rain}, Wind) = Entropy(S) - \frac{2}{5}Entropy(S_{Strong}) - \frac{3}{5}Entropy(S_{Weak})$$

$$Gain(S_{Rain}, Wind) = 0.97 - \frac{2}{5}\,0.0 - \frac{3}{5}0.0 = 0.97$$

Day	Temp	Humidity	Wind	Play Tennis
D4	Mild	High	Weak	Yes
D5	Cool	Normal	Weak	Yes
D6	Cool	Normal	Strong	No
DI0	Mild	Normal	Weak	Yes
DI4	Mild	High	Strong	No

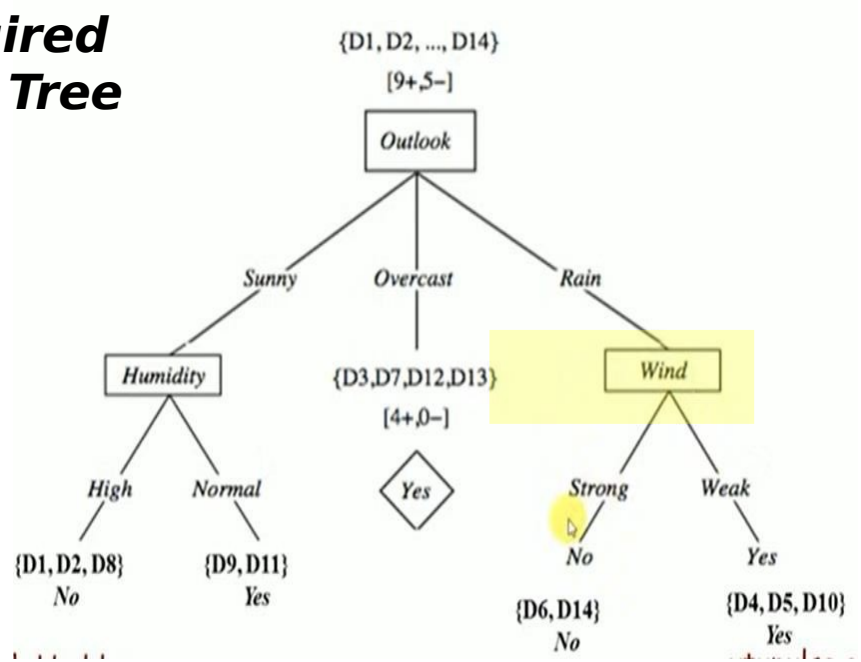
$$Gain(S_{Rain}, Temp) = 0.0192$$

$$Gain(S_{Rain}, Humidity) = 0.0192$$

$$Gain(S_{Rain}, Wind) = 0.97$$

This has the highest Information gain

Our required Decision Tree is:



Solved Example 2:

1. What is the entropy of this collection of training examples with respect to the target function classification?
2. What is the information gain of *a1* and *a2* relative to these training examples?
3. Draw decision tree for the given dataset.

Instance	Classification	a1	a2
1	+	T	T
2	+	T	T
3	-	T	F
4	+	F	F
5	-	F	T
6	-	F	T

Reference video: https://www.youtube.com/watch?v=JO2wiZif2OM&t=3s

Instance	Classification	a1	a2
1	+	T	T
2	+	T	T
3	-	T	F
4	+	F	F
5	-	F	T
6	-	F	T

Attribute: a1

$Values(a1) = T, F$

$S = [3+, 3-]$ $\quad Entropy(S) = 1.0$

$S_T = [2+, 1-]$ $\quad Entropy(S_T) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3} = 0.9183$

$S_F \leftarrow [1+, 2-]$ $\quad Entropy(S_F) = -\frac{1}{3}log_2\frac{1}{3} - \frac{2}{3}log_2\frac{2}{3} = 0.9183$

$$Gain(S, a1) = Entropy(S) - \sum_{v \in \{T,F\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a1) = Entropy(S) - \frac{3}{6}Entropy(S_T) - \frac{3}{6}Entropy(S_F)$$

$$Gain(S, a1) = 1.0 - \frac{3}{6} * 0.9183 - \frac{3}{6} * 0.9183 = 0.0817$$

Attribute: a2

$Values(a2) = T, F$

$S = [3+, 3-]$ $\quad Entropy(S) = 1.0$

$S_T = [2+, 2-]$ $\quad Entropy(S_T) = 1.0$

$S_F \leftarrow [1+, 1-]$ $\quad Entropy(S_F) = 1.0$

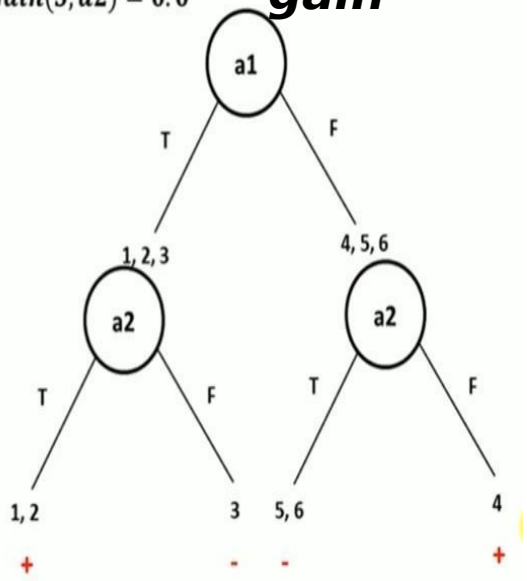
$$Gain(S, a2) = Entropy(S) - \sum_{v \in \{T,F\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a2) = Entropy(S) - \frac{4}{6}Entropy(S_T) - \frac{2}{6}Entropy(S_F)$$

$$Gain(S, a2) = 1.0 - \frac{4}{6} * 1.0 - \frac{2}{6} * 1.0 = 0.0$$

$Gain(S, a1) = 0.0817$ **Maximum gain**

$Gain(S, a2) = 0.0$



Solved Example 3:

Decision Tree Algorithm – ID3 Solved Example

Instance	a1	a2	a3	Classification
1	True	Hot	High	No
2	True	Hot	High	No
3	False	Hot	High	Yes
4	False	Cool	Normal	Yes
5	False	Cool	Normal	Yes
6	True	Cool	High	No
7	True	Hot	High	No
8	True	Hot	Normal	Yes
9	False	Cool	Normal	Yes
10	False	Cool	High	Yes

Instance	a1	a2	a3	Classification
1	True	Hot	High	No
2	True	Hot	High	No
3	False	Hot	High	Yes
4	False	Cool	Normal	Yes
5	False	Cool	Normal	Yes
6	True	Cool	High	No
7	True	Hot	High	No
8	True	Hot	Normal	Yes
9	False	Cool	Normal	Yes
10	False	Cool	High	Yes

Example - 3
Decision Tree Algorithm – ID3
Solved Example

Attribute: a1

$Values(a1) = True, False$

$S = [6+, 4-]$ $\quad Entropy(S) = -\frac{6}{10}log_2\frac{6}{10} - \frac{4}{10}log_2\frac{4}{10} = 0.9709$

$S_{True} = [1+, 4-]$ $\quad Entropy(S_{True}) = -\frac{1}{5}log_2\frac{1}{5} - \frac{4}{5}log_2\frac{4}{5} = 0.7219$

$S_{Flase} \leftarrow [5+, 0-]$ $\quad Entropy(S_{False}) = 0.0$

$$Gain(S, a1) = Entropy(S) - \sum_{v \in \{True, False\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a1) = Entropy(S) - \frac{5}{10}Entropy(S_{True}) - \frac{5}{10}Entropy(S_{False})$$

$$Gain(S, a1) = 0.9709 - \frac{5}{10} * 0.7219 - \frac{5}{10} * 1 = 0.6099$$

Attribute: a2

$Values(a2) = Hot, Cool$

$S = [6+, 4-]$ $\quad Entropy(S) = -\frac{6}{10}log_2\frac{6}{10} - \frac{4}{10}log_2\frac{4}{10} = 0.9709$

$S_{Hot} = [2+, 3-]$ $\quad Entropy(S_{Hot}) = -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} = 0.9709$

$S_{Cool} \leftarrow [4+, 1-]$ $\quad Entropy(S_{Cool}) = -\frac{4}{5}log_2\frac{4}{5} - \frac{1}{5}log_2\frac{1}{5} = 0.7219$

$$Gain(S, a2) = Entropy(S) - \sum_{v \in \{Hot, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a2) = Entropy(S) - \frac{5}{10}Entropy(S_{Hot}) - \frac{5}{10}Entropy(S_{Cool})$$

$$Gain(S, a2) = 0.9709 - \frac{5}{10} * 0.9709 - \frac{5}{10} * 0.7219 = 0.1245$$

Attribute: a3

$Values(a3) = High, Normal$

$S = [6+, 4-]$ $\quad Entropy(S) = -\frac{6}{10}log_2\frac{6}{10} - \frac{4}{10}log_2\frac{4}{10} = 0.9709$

$S_{High} = [2+, 4-]$ $\quad Entropy(S_{High}) = -\frac{2}{6}log_2\frac{2}{6} - \frac{4}{6}log_2\frac{4}{6} = 0.9183$

$S_{Normal} \leftarrow [4+, 0-]$ $\quad Entropy(S_{Normal}) = 0.0$

$$Gain(S, a3) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

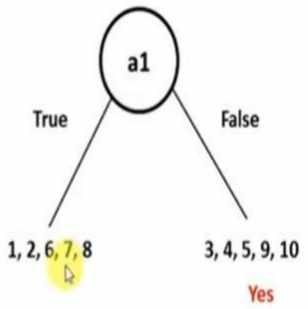
$$Gain(S, a3) = Entropy(S) - \frac{6}{10}Entropy(S_{High}) - \frac{4}{10}Entropy(S_{Normal})$$

$$Gain(S, a3) = 0.9709 - \frac{6}{10} * 0.9183 - \frac{4}{10} * 0.0 = 0.4199$$

$Gain(S, a1) = 0.6099$ – *Maximum Gain*

$Gain(S, a2) = 0.1245$

$Gain(S, a3) = 0.4199$



Instance	a2	a3	Classification
1	Hot	High	No
2	Hot	High	No
6	Cool	High	No
7	Hot	High	No
8	Hot	Normal	Yes

Example - 3
Decision Tree Algorithm – ID3
Solved Example

Attribute: a2

$Values(a2) = Hot, Cool$

$S_{a1} = [1+, 4-]$ $\quad Entropy(S_{a1}) = -\frac{1}{5}log_2\frac{1}{5} - \frac{4}{5}log_2\frac{4}{5} = 0.7219$

$S_{Hot} = [1+, 3-]$ $\quad Entropy(S_{Hot}) = -\frac{1}{4}log_2\frac{1}{4} - \frac{3}{4}log_2\frac{3}{4} = 0.8112$

$S_{Cool} \leftarrow [0+, 1-]$ $\quad Entropy(S_{Cool}) = 0.0$

$$Gain(S, a2) = Entropy(S) - \sum_{v \in \{Hot, Cool\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a2) = Entropy(S) - \frac{4}{5} Entropy(S_{Hot}) - \frac{1}{5} Entropy(S_{Cool})$$

$$Gain(S, a2) = 0.9709 - \frac{4}{5} * 0.8112 - \frac{1}{5} * 0.0 = 0.3219$$

Attribute: a3

$Values(a3) = High, Normal$

$S_{a1} = [1+, 4-]$ $\quad Entropy(S_{a1}) = -\frac{1}{5}log_2\frac{1}{5} - \frac{4}{5}log_2\frac{4}{5} = 0.7219$

$S_{High} = [0+, 4-]$ $\quad Entropy(S_{High}) = 0.0$

$S_{Normal} \leftarrow [1+, 0-]$ $\quad Entropy(S_{Normal}) = 0.0$

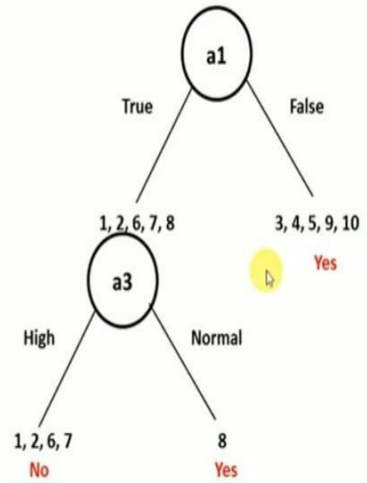
$$Gain(S, a3) = Entropy(S) - \sum_{v \in \{High, Normal\}} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$Gain(S, a3) = Entropy(S) - \frac{4}{5} Entropy(S_{High}) - \frac{1}{5} Entropy(S_{Normal})$$

$$Gain(S, a3) = 0.9709 - \frac{4}{5} * 0.0 - \frac{1}{5} * 0.0 = 0.7219$$

$Gain(S_{a1}, a2) = 0.3219$

$Gain(S_{a1}, a3) = 0.7219 - Maximum\ Gain$



2. Gain Ratio

- Used by C4.5 (the successor of ID3)
- Gain Ratio is an alternative to Information Gain that is used to select the attribute for splitting in a decision tree.
- It is used to overcome the problem of bias towards the attribute with many outcomes.
- Gain Ratio normalizes the Information Gain with respect to the total entropy of all splits based on values of an attribute
- It introduces a normalizing term called Intrinsic Information.
- Gain ratio is given by:

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

2. Gain Ratio

- For example,
  - Suppose we have two features, “Color” and “Size” and we want to build a decision tree to predict the type of fruit based on these two features.
  - The “Color” feature has three outcomes (red, green, yellow) and the “Size” feature has two outcomes (small, large).
  - Using the information gain method, the “Color” feature would be chosen as the best feature to split on because it has the highest information gain.
  - However, this could be a problem because the “Size” feature could be a better feature to split on because it is less ambiguous and has fewer outcomes.

- **Reference video for solved example:**
https://www.youtube.com/watch?v=cmXLhqv67ns

Gini Index:

- The Gini Index measures the impurity of data set (D) as:
- Gini(D) = 1 - $\sum_1^n {p_i}^2$

  Where,

  n = Number of classes, p_i = Probability of i^{th} class.
- It consider binary split for each attribute.
- When D is partition into D_1 and D_2 then

  $Gini(D) = D_1/D\ Gini(D_1) + D_2/D\ Gini(D_2)$
- The attribute that maximize the reduction in impurity is selected as splitting attribute.

Pi = count of specific class level/ total count of D

An attribute with the low Gini index should be preferred as compared to the high Gini index.

It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.

Decision Tree using Gini Index – Solved Example

Weekend	Weather	Parents	Money	Decision
W1	Sunny	Yes	Rich	Cinema
W2	Sunny	No	Rich	Tennis
W3	Windy	Yes	Rich	Cinema
W4	Rainy	Yes	Poor	Cinema
W5	Rainy	No	Rich	Stay In
W6	Rainy	Yes	Poor	Cinema
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W9	Windy	Yes	Rich	Cinema
W10	Sunny	No	Rich	Tennis

- Compute the Gini Index for the overall collection of training examples.
- There are four possible output variables Cinema, Tennis, Stay In and Shopping.
- The data has 6 instances of Cinema, 2 instances of Tennis, 1 instance of Stay In and 1 of shopping.

- $Gini(S) = 1 - [\left(\frac{6}{10}\right)^2 + \left(\frac{2}{10}\right)^2 + \left(\frac{1}{10}\right)^2 + \left(\frac{1}{10}\right)^2] = 0.58$

Node	Probability
Cinema	6/10
Tennis	2/10
Shopping	1/10
Stay In	1/10

Reference video:
https://www.youtube.com/watch?v=zNYdkpAcP-g

Weekend	Weather	Parents	Money	Decision
W1	Sunny	Yes	Rich	Cinema
W2	Sunny	No	Rich	Tennis
W3	Windy	Yes	Rich	Cinema
W4	Rainy	Yes	Poor	Cinema
W5	Rainy	No	Rich	Stay In
W6	Rainy	Yes	Poor	Cinema
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W9	Windy	Yes	Rich	Cinema
W10	Sunny	No	Rich	Tennis

- Computation of Gini Index for Money Attribute
- It has two possible values of Rich (7 examples) and Poor (3 examples).
- For Money = Poor, there are 3 examples with "Cinema".
- $Gini(S) = 1 - [\left(\frac{3}{3}\right)^2] = 0$ ✓
- For Money = Rich, there are 2 examples with "Tennis", 3 examples with "Cinema" and 1 example with "Stay in", "Shopping" each
- $Gini(S) = 1 - [\left(\frac{2}{7}\right)^2 + \left(\frac{3}{7}\right)^2 + \left(\frac{1}{7}\right)^2 + \left(\frac{1}{7}\right)^2] = 0.694$
- $Weighted\ Average(Money)$

$$= 0 * \left(\frac{3}{10}\right) + 0.694 * \left(\frac{7}{10}\right) = 0.486$$

Weekend	Weather	Parents	Money	Decision
W1	Sunny	Yes	Rich	Cinema
W2	Sunny	No	Rich	Tennis
W3	Windy	Yes	Rich	Cinema
W4	Rainy	Yes	Poor	Cinema
W5	Rainy	No	Rich	Stay In
W6	Rainy	Yes	Poor	Cinema
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W9	Windy	Yes	Rich	Cinema
W10	Sunny	No	Rich	Tennis

- Computation of **Gini Index for Parents** Attribute
- It has two possible values of Yes (5 examples) and No (5 examples).
- For **Parents = Yes,** there are 5 examples, all with "Cinema".
- $Gini(S) = 1 - [\left(\frac{5}{5}\right)^2] = 0$
- For **Parents = No,** there are 2 examples with "Tennis", 1 example with "Stay in", "Shopping" and "Cinema" each
- $Gini(S) = 1 - [\left(\frac{2}{5}\right)^2 + \left(\frac{1}{5}\right)^2 + \left(\frac{1}{5}\right)^2 + \left(\frac{1}{5}\right)^2] =$ 0.72
- $Weighted\ Average(Parents)$

$$= 0 * \left(\frac{5}{10}\right) + [0.72 * \left(\frac{5}{10}\right) = 0.36$$

Weekend	Weather	Parents	Money	Decision
W1	Sunny	Yes	Rich	Cinema
W2	Sunny	No	Rich	Tennis
W3	Windy	Yes	Rich	Cinema
W4	Rainy	Yes	Poor	Cinema
W5	Rainy	No	Rich	Stay In
W6	Rainy	Yes	Poor	Cinema
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W9	Windy	Yes	Rich	Cinema
W10	Sunny	No	Rich	Tennis

- Computation of Gini Index for Weather Attribute
- It has three possible values of Sunny (3 examples), Rainy (3 examples) and Windy (4 examples).
- For Weather = Sunny, there are 2 examples with "Cinema" and 1 with "Tennis".
- $Gini(Sunny) = 1 - [\left(\frac{2}{3}\right)^2 + \left(\frac{1}{3}\right)^2] = 0.444$
- For Weather = Rainy, there are 2 examples with "Cinema" and 1 example with "Stay in"
- $Gini(Rainy) = 1 - [\left(\frac{2}{3}\right)^2 + \left(\frac{1}{3}\right)^2] = 0.444$
- For Weather = Windy, there are 3 examples with "Cinema" and 1 example with "Shopping"
- $Gini(Windy) = 1 - [\left(\frac{3}{4}\right)^2 + \left(\frac{1}{4}\right)^2] = 0.375$

$$Weighted\ Average(Weather)$$

$$= 0.444 * \left(\frac{3}{10}\right) + 0.444 * \left(\frac{3}{10}\right) + 0.375 * \left(\frac{4}{10}\right)$$

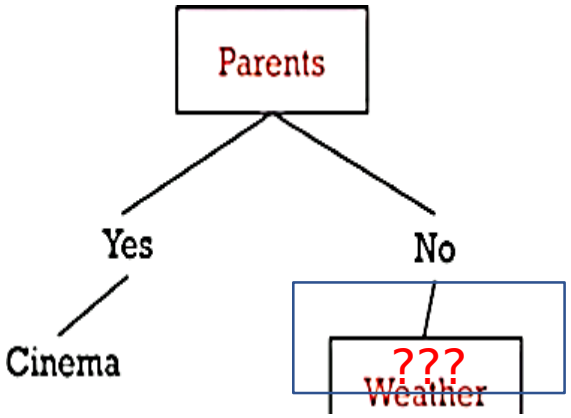
$$= 0.416$$

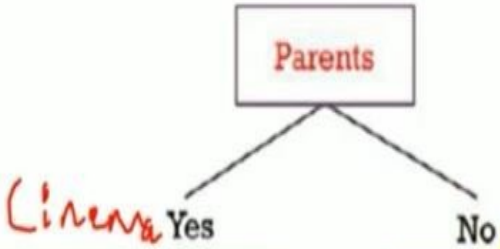
For Weather - Gini Index: 0.416

For Parents - Gini Index: 0.36

For Money - Gini Index: 0.486

Parents is selected as it has smallest Gini index.





Weekend	Weather	Parents	Money	Decision
W1	Sunny	Yes	Rich	Cinema
W3	Windy	Yes	Rich	Cinema
W4	Rainy	Yes	Poor	Cinema
W6	Rainy	Yes	Poor	Cinema
W9	Windy	Yes	Rich	Cinema

Weekend	Weather	Parents	Money	Decision
W2	Sunny	No	Rich	Tennis
W5	Rainy	No	Rich	Stay In
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W10	Sunny	No	Rich	Tennis

Weekend	Weather	Parents	Money	Decision
W2	Sunny	No	Rich	Tennis
W5	Rainy	No	Rich	Stay In
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W10	Sunny	No	Rich	Tennis

Weekend	Weather	Parents	Money	Decision
W2	Sunny	No	Rich	Tennis
W5	Rainy	No	Rich	Stay In
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W10	Sunny	No	Rich	Tennis

Computation of Gini Index for Parents = No | Weather Attribute

- **Sunny (2 examples)**
- For Parent = No | Weather = Sunny, there are 2 example with "Tennis.
- $Gini(S) = 1 - [(\frac{2}{2})^2] = 0$

Computation of Gini Index for Parents = No | Money Attribute

- Rich (4 examples)
- For Parents = No | Money = Rich, there is 1 example with "stay in" and "Shopping" each and 2 examples of "Tennis".
- $Gini(S) = 1 - [(\frac{1}{4})^2 + (\frac{1}{4})^2 + (\frac{2}{4})^2] = 0.625$

Computation of Gini Index for Parents = No | Weather Attribute

- **Rainy (1 example).**
- For Parents = No | Weather = Rainy, there is 1 example with "Stay In".
- $Gini(S) = 1 - [(\frac{1}{1})^2] = 0$

Computation of Gini Index for Parents = No | Money Attribute

- Poor (1 example)
- For Parents = No | Money = Poor, there is 1 example with "Cinema".
- $Gini(S) = 1 - [(\frac{1}{1})^2] = 0$
- $Weighted\ Average\ (Parents = No | Money) = 0.625 * (4/5) + 0 * (1/5) = 0.5$

Computation of Gini Index for Parents = No | Weather Attribute

- **Windy (2 example)**
- For Parents = No | Weather = Windy, there is 1 example with "Cinema" and 1 examp with "Shopping".
- $Gini(S) = 1 - [(\frac{1}{2})^2 + (\frac{1}{2})^2] = 0.5$

$$Weighted\ Average(Parents = No | Weather) = 0 * (\frac{2}{5}) + 0 * (\frac{1}{5}) + 0.5 * (\frac{2}{5}) = 0.2$$

For Parents = No | Weather - Gini Index: 0.2

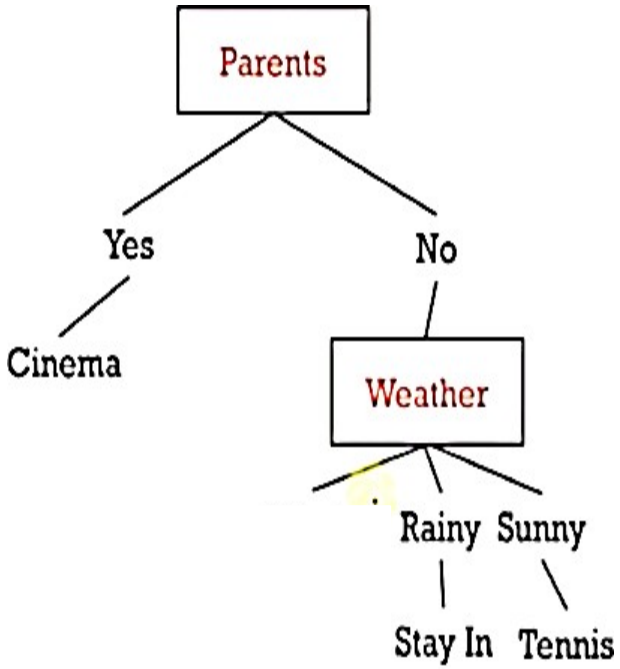
For Parents = No | Money - Gini Index: 0.5

Weather is selected as it has smallest Gini index.

For Parents = No | Weather - Gini Index: 0.2

For Parents = No | Money - Gini Index: 0.5

Weather is selected as it has smallest Gini index.



Weekend	Weather	Parents	Money	Decision
W2	Sunny	No	Rich	Tennis
W5	Rainy	No	Rich	Stay In
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W10	Sunny	No	Rich	Tennis

Now, for Parent=No & Weather=Sunny, we have all instances as Tennis.

Weekend	Weather	Parents	Money	Decision
W2	Sunny	No	Rich	Tennis
W10	Sunny	No	Rich	Tennis

Now, for Parents=No & Weather=Rainy, we have all instances as Stay In.

Weekend	Weather	Parents	Money	Decision
W5	Rainy	No	Rich	Stay In

Weekend	Weather	Parents	Money	Decision
W2	Sunny	No	Rich	Tennis
W5	Rainy	No	Rich	Stay In
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W10	Sunny	No	Rich	Tennis

Now, for Parent=No & Weather=Sunny, we have all instances as Tennis.

Weekend	Weather	Parents	Money	Decision
W2	Sunny	No	Rich	Tennis
W10	Sunny	No	Rich	Tennis

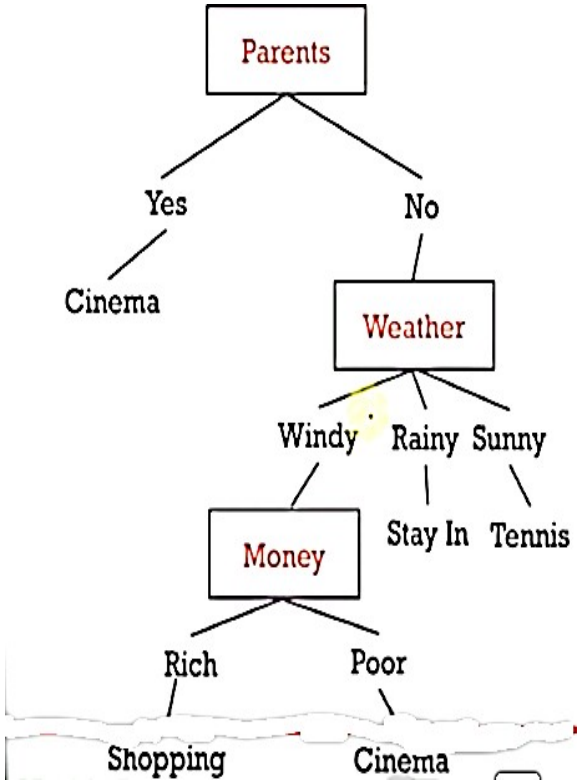
Now, for Parents=No & Weather=Rainy, we have all instances as Stay In.

Weekend	Weather	Parents	Money	Decision
W5	Rainy	No	Rich	Stay In

Now, for Parent=No & Weather=Windy, we need to split.

Weekend	Weather	Parents	Money	Decision
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping

Now, our required Decision tree is:



Pruning: Getting an Optimal Decision tree

- *Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.*
- A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset.
- Therefore, a technique that decreases the size of the learning tree without reducing accuracy is known as Pruning.
- There are mainly two types of tree **pruning** technology used:
  - **Cost Complexity Pruning**
  - **Reduced Error Pruning.**

Advantages and Disadvantages of Decision Tree

Advantages of the Decision Tree
Decision Tree

1. Inexpensive to construct
2. Extremely fast at classifying unknown records
3. Easy to interpret for small-sized trees
4. Accuracy is comparable to other classification techniques for many simple data sets

Disadvantages of the

1. The decision tree contains lots of layers, which makes it complex.
2. It may have an overfitting issue, which can be resolved using the **Random Forest algorithm.**
3. For more class labels, the computational complexity of the decision tree may increase.

Tree Pruning –

- Tree Pruning is performed in order to remove anomalies in training data due to noise or outliers. - The pruned trees are smaller and less complex.
- Pruning is a technique in machine learning that reduces the size of decision trees by removing sections of the tree that provide little power to classify instances.
- The dual goal of pruning is reduced complexity of the final classifier as well as better predictive accuracy by the reduction of over fitting and removal of sections of a classifier that may be based on noisy or erroneous data.
- One of the questions that arise in a decision tree algorithm is the optimal size of the final tree.
- A tree that is too large risks over fitting the training data and poorly generalizing to new samples.
- A small tree might not capture important structural information about the sample space.
- It is hard to tell when a tree algorithm should stop because it is impossible to tell if the addition of a single extra node will dramatically decrease error.
- A common strategy is to grow the tree until each node contains a small number of instances then use pruning to remove nodes that do not provide additional information.
- Pruning should reduce the size of a learning tree without reducing predictive accuracy as measured by a test set or using cross-validation.
- There are many techniques for tree pruning that differ in the measurement that is used to optimize performance.

Tree pruning approaches

- Prepruning - The tree is pruned by halting its construction early.
- Postpruning - This approach removes subtree form fully grown tree.

Pre-pruning

- Based on statistical significance test.
- Stop growing the tree when there is no statistically significant association between any attribute and the class at a particular node
- Most popular test: chi-squared test
- ID3 used chi-squared test in addition to information gain.
- Only statistically significant attributes were allowed to be selected by information gain procedure.
- Pre-pruning may stop the growth process prematurely: early stopping
- Pre-pruning faster than post-pruning

Post-pruning

- First, build full tree then, prune it.
- Fully-grown tree shows all attribute interactions
- Problem: some subtrees might be due to chance effects
- Two pruning operations:
- Subtree replacement
- Subtree raising

Possible strategies:

- error estimation
- Significance testing
- MDL principle

Subtree replacement selects a subtree and replaces it with a single leaf.

Subtree raising selects a subtree and replaces it with the child one ie, a "sub-subtree" replaces its parent)

Advantages of Decision Tree Classifier

- Inexpensive to construct
- Extremely fast at classifying unknown records
- Easy to interpret for small-sized trees
- Accuracy is comparable to other classification techniques for many simple data sets

Rule Based Classifier:

- It classifies records by using a collection of **“If ……. Then…..”** rules.
- A rule base classifier uses a set of “If ……. Then…..” rules for classification. eg: If age = youth AND student = yes THEN buys_computer = yes.
- The ‘If’ part or left hand side of a rule is known as the rule antecedent or precondition where as the ‘Then” part or right hand side is the rule consequent.
- If the condition in a rule antecedent holds true for a given tuple, the rule antecedent is satisfied and that the rule covers the tuple.

https://www.youtube.com/watch?v=MFig-zD8tNY

- Coverage of a rule is the fraction of records that satisfy the antecedent of a rule.
- i.e Coverage = Ncovers / D
  - Where, Ncovers = number of record that can be classified by the rule.
  - D = total data set.

- Accuracy of a rule is fraction of records that satisfy both the antecedent and consequent of a rule.
  - Accuracy = Ncorrect / Ncovers
  - Where,
  - Ncorrect = Number of records that are correctly classified by the rule
  - Ncovers = Number of record that can be classified by the rule

How does Rule-Based Classifier work?

- If a rule is **satisfied** by a tuple, the **rule is said to be triggered**. **Triggering doesn't always mean firing because** there may be more than one rules that can be satisfied.
- Three different cases occur for classification.

Case-I: If only one rule is satisfied

- When any instances is triggered by only one **rule then triggered is consider as firing rule**

Case-II: If more than one rules are satisfied

- If more than one rules are triggered, we need a **conflict resolution strategy to find which rule is fired.**
- Size ordering: assign the highest priority to the triggering rules that has the "toughest" requirement (i.e., with the most attribute tests)
- Class-based ordering: decreasing order of prevalence or misclassification cost per class
- Rule-based ordering (decision list): rules are organized into one long priority list, according to some measure of rule quality or by expert

Case-III: If no rule is satisfied

- If any instance not triggered by any rule, use **default class for classification**. **Mostly most frequent class is assigned as default class.**

3. Rule Based

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
human	warm	yes	no	no	mammals
python	cold	no	no	no	reptiles
salmon	cold	no	no	yes	fishes
whale	warm	yes	no	yes	mammals
frog	cold	no	no	sometimes	amphibians
komodo	cold	no	no	no	reptiles
bat	warm	yes	yes	no	mammals
pigeon	warm	no	yes	no	birds
cat	warm	yes	no	no	mammals
leopard shark	cold	yes	no	yes	fishes
turtle	cold	no	no	sometimes	reptiles
penguin	warm	no	no	sometimes	birds
porcupine	warm	yes	no	no	mammals
eel	cold	no	no	yes	fishes
salamander	cold	no	no	sometimes	amphibians
gila monster	cold	no	no	no	reptiles
platypus	warm	no	no	no	mammals
owl	warm	no	yes	no	birds
dolphin	warm	yes	no	yes	mammals
eagle	warm	no	yes	no	birds

R1: (Give Birth = no) & (Can Fly = yes) → Birds

R2: (Give Birth = no) & (Live in Water = yes) → Fishes

R3: (Give Birth = yes) & (Blood Type = warm) → Mammals

R4: (Give Birth = no) & (Can Fly = no) → Reptiles

R5: (Live in Water = sometimes) → Amphibians

A rule r **covers** an instance x if the attributes of the instance satisfy the condition of the rule

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
hawk	warm	no	yes	no	?
grizzly bear	warm	yes	no	no	?

The rule R1 above covers a **hawk** => **Bird**

The rule R3 covers the **grizzly bear** => **Mammal**

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
lemur	warm	yes	no	no	?
turtle	cold	no	no	sometimes	?
dogfish shark	cold	yes	no	yes	?

A **lemur** triggers rule R3, so it is classified => **Mammal**

A **turtle** triggers both R4 and R5

A **dogfish shark** trigger matches none of the rules

S.No.	Name	Blood Type	Give Birth	Can fly	Live in water	Class
1	Lemur	Warm	Yes	No	No	?
2	Turtle	Cold	No	No	Sometimes	?
3	Shark	Cold	Yes	No	Yes	?

Rule base

R1: (Give Birth = No) ^ (Can fly = Yes) => Birds
R2: (Give Birth = No) ^ (Live in Water = Yes) => Fishes
R3: (Give Birth = Yes) ^ (Blood Type = Warm) => Mammals
R4: (Give Birth = No) ^ (Can fly = No) => Reptiles
R5: (Live in Water = Sometimes) => Amphibians

- In above example, R1 and R2 don't have any coverage. R3, R4 & R5 have coverage.
- Instance 1 is triggered by R3, instance 2 is triggered by R4 & R5 and instance 3 is not triggered by any instances.
- Since instance 1 is triggered by only one rule (R3) so it is fired as a class mammal, instance 2 is triggered by more than two rules (R4 & R5) and hence conflict occurs.
- To resolve the conflict the class can be identified using priority (rule priority or class priority). Instance 3 is not triggered by any rules, to resolve this conflict default class can be used.

Characteristics of Rule Based Data Mining Classifiers

Rule Based Data Mining classifiers possess two significant characteristics:

1. **Rules may not be mutually exclusive.**

   Different rules are generated for data, so it is possible that many rules can cover the same record. That is why rules are called non-mutually exclusive.

2. **Rules may not be exhaustive.**

   It is possible that some of the data entries may not be covered by any of the rules; thus, rules are called not to be exhaustive.

Effect of Rule Simplification

- Rules are no longer mutually exclusive
  - A record may trigger more than one rule
  - Solution?
    - Ordered rule set
    - Unordered rule set – use voting schemes
- Rules are no longer exhaustive
  - A record may not trigger any rules
  - Solution?
    - Use a default class

Rule Ordering Schemes

- Rule-based ordering
  - Individual rules are ranked based on their quality
- Class-based ordering
  - Rules that belong to the same class appear together

Rule-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

Class-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

(Refund=No, Marital Status={Married}) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

Building Classification Rules –

- Two approaches are used to build classification rules.

1) Direct Method - Extract rules directly from data. It is an inductive and sequential approach.

Sequential Covering

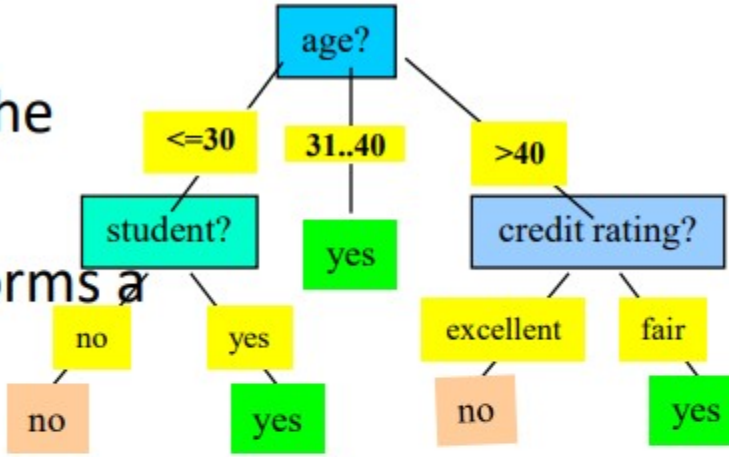
- 1. Start from an empty rule
- 2. Grow a rule using the Learn-One-Rule function
- 3. Remove training records covered by the rule
- 4. Repeat Step (2) and (3) until stopping criterion is met.

Aspects of Sequential Covering

- Rule Growing
- Instance Elimination
- Rule Evaluation
- Stopping Criterion
- Rule Pruning

2) Rule extraction from Decision Tree

- Rules are *easier to understand* than large trees
- One rule is created *for each path* from the root to a leaf
- Each attribute-value pair along a path forms a conjunction: the leaf holds the class prediction



- Rules are mutually exclusive and exhaustive
- Example: Rule extraction from our *buys_computer* decision-tree

IF *age* = young AND *student* = *no* THEN *buys_computer* = *no*
IF *age* = young AND *student* = *yes* THEN *buys_computer* = *yes*
IF *age* = mid-age THEN *buys_computer* = *yes*
IF *age* = old AND *credit_rating* = *excellent* THEN *buys_computer* = *no*
IF *age* = old AND *credit_rating* = *fair* THEN *buys_computer* = *yes*

Measures of Coverage and Accuracy

Tid	Refund	Marital Status	Taxable Income	Class
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Coverage of a rule:

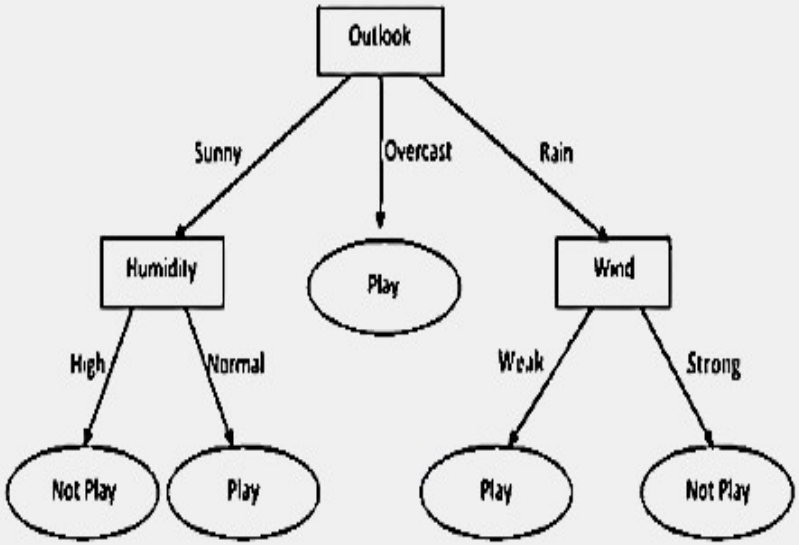
- Fraction of all records that satisfy the antecedent of a rule
- Count(instances with antecedent) / Count(training set)
- Example on left: ***(Status = 'Single')*** **->** ***no***, Coverage = 4/10 = 40%

Accuracy of a rule:

- Fraction of records that satisfy the antecedent that also satisfy the consequent of a rule
- Count (instances with antecedent AND consequent) / Count(instances with antecedent)
- Example on left: ***(Status = 'Single')*** **->** ***no***, accuracy = 2/4 = 50%

Rule extraction from Decision Tree

- To extract rules form a Decision tree, one rule is created for each path from the root to a lead node.
- Each splitting criterion is logically ANDed to form the rule antecedent (If part).
- Leaf node holds the class prediction for rule consequent (Then part).



For the decision tree above, there are five possible rules which can be extracted (because there are five leaf nodes). They are as follows:

R1: IF *Outlook* = *sunny*	AND *Humidity* = *High*	THEN *Play_Tennis* = *no*
R2: IF *Outlook* = *sunny*	AND *Humidity* = *Normal*	THEN *Play_Tennis* = *yes*
R3: IF *Outlook* = *Overcast*		THEN *Play_Tennis* = *yes*
R4: IF *Outlook* = *Rain*	AND *Wind* = *Weak*	THEN *Play_Tennis* = *yes*
R5: IF *Outlook* = *Rain*	AND *Wind* = *Strong*	THEN *Play_Tennis* = *no*

4. Nearest Neighbor Classifier

4. Nearest Neighbor Classifier

- KNN is one of the simplest and strong supervised learning algorithms used for classification and for regression in data mining.
- K- NN algorithm is based on the principle that, **“the similar things or objects exist closer to each other.”**
- KNN is most commonly used to classify the data points that are separated into several classes, in order to make prediction for new sample data points.
- KNN is a **non-parametric** learning algorithm.
- KNN is a **lazy learning algorithm.**
- KNN classifies the data points based on the different kind of similarity measures (e.g. Euclidean distance, Manhattan distance, Hamming distance, etc).
- In KNN algorithm ‘K’ refers to the number of neighbors to consider for classification.
- The value of ‘K’ in KNN algorithm must be selected carefully otherwise it may cause defects in our model.

Nearest neighbor classifier requires:

- Set of stored records
- Distance matric to compute distance between records. For distance calculation any standard approach can be used such as **Euclidean distance.**
- The **value of 'K',** the number of nearest neighbor to retrieve.

To classify the unknown records

- Compute distance to other training records.
- Identify the k-nearest neighbor.
- Use class label nearest neighbors to determine the class label of unknown record. In case of conflict, use majority vote for classification.

Issues of classification using k-nearest neighbor classification

1. **Choosing the value of K**

- One of challenge in classification is to choose the appropriate value of K. If K is **too small**, it is **sensitive to noise** points. If K is too **large**, neighbor may include points **from other classes**.
- With the change of value of K, the classification result may vary

2. **Scaling Issue**

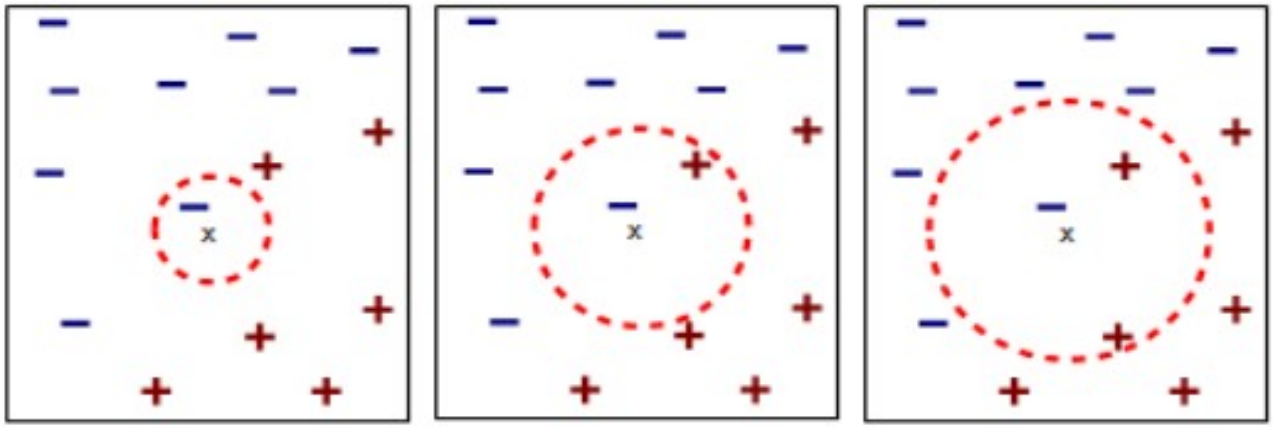
- Attribute may have to be scaled to prevent distance measure from being dominated by one of attributes. Eg. Height, Temperature etc.

3. **Distance computing for non-numeric data.**

- Use Distance as 0 for the same data and maximum possible distance for different data.

4. **Missing values**

- Use maximum possible distance



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

Disadvantages:

•

4. Nearest Neighbor Classifier

- 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.

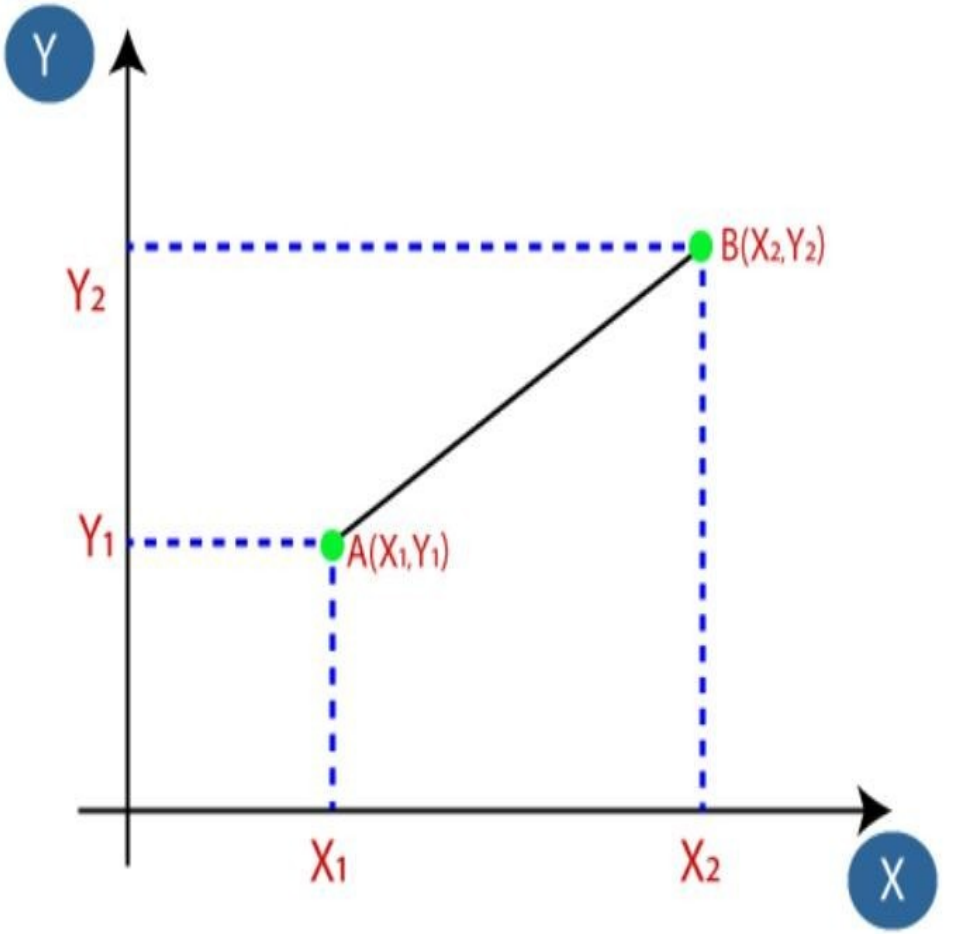
- Disadvantages of KNN Algorithm:
  - Always needs to determine the value of K which may be complex some time.
  - The computation cost is high because of calculating the distance between then data points for all the training samples.

Poor accuracy when data have noise and irrelevant attributes.

Slow when classifying test tuples.

Classifying unknown records are relatively expensive.

Euclidian Distance



Euclidean Distance between A_1 and $B_2 = \sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

Hamming Distance

- Given two integers, the task is to find the hamming distance between two integers.
- Hamming Distance between two integers is the number of bits that are different at the same position in both numbers.

Examples:

Input: n1 = 9, n2 = 14

Output: 3

9 = 1001, 14 = 1110

No. of Different bits = 3

- Hamming distance= 3

- Let x1 and x2 are the attribute values of two instances.
- Then, in hamming distance, if the categorical values are same or matching that is x1 is same as x2 then distance is 0, otherwise 1.
- For example,
- If value of **x1 is blue** and **x2 is also blue** then the distance between x1 and x2 is **0**.
- If value of **x1 is blue** and **x2 is red** then the distance between x1 and x2 is **1**.

KNN algorithm

- **Step-1:** Select the parameter K
- **Step-2:** Calculate the distance of the new data with all the training data.
- **Step 3:** Take the K nearest neighbors as per the calculated Euclidean distance.
- **Step-4:** Among these k neighbors, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that *category (i.e.. Target class)* for which the number of the neighbor is maximum.

Solved Example 1:

- Apply K Nearest Neighbor classifier to predict the diabetic patient with the given features (BMI, Age). The target label is Sugar. The test example is: BMI= 43.6 and Age= 40, Sugar= ?. Assume K=3

BMI	Age	Sugar
33.6	50	1
26.6	30	0
23.4	40	0
43.1	67	0
35.3	23	1
35.9	67	1
36.7	45	1
25.7	46	0
23.3	29	0
31	56	1

BMI	Age	Sugar
33.6	50	1
26.6	30	0
23.4	40	0
43.1	67	0
35.3	23	1
35.9	67	1
36.7	45	1
25.7	46	0
23.3	29	0
31	56	1

- First Calculate the distance between the test instance and training instances.

Test Example

BMI=43.6, Age=40, Sugar=?

- $Distance = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

Step-1: Calculate the distance of the new data with all the training data.

BMI	Age	Sugar	Distance	
33.6	50	1	$\sqrt{(43.6-33.6)^2+(40-50)^2}$	14.14
26.6	30	0	$\sqrt{(43.6-26.6)^2+(40-30)^2}$	19.72
23.4	40	0	$\sqrt{(43.6-23.4)^2+(40-40)^2}$	20.20
43.1	67	0	$\sqrt{(43.6-43.1)^2+(40-67)^2}$	27.00
35.3	23	1	$\sqrt{(43.6-35.3)^2+(40-23)^2}$	18.92
35.9	67	1	$\sqrt{(43.6-35.9)^2+(40-67)^2}$	28.08
36.7	45	1	$\sqrt{(43.6-36.7)^2+(40-45)^2}$	8.52
25.7	46	0	$\sqrt{(43.6-25.7)^2+(40-46)^2}$	18.88
23.3	29	0	$\sqrt{(43.6-23.3)^2+(40-29)^2}$	23.09
31	56	1	$\sqrt{(43.6-31)^2+(40-56)^2}$	20.37

Test Example

BMI=43.6, Age=40, Sugar=?

Step 2: Take the K nearest neighbors as per the calculated Euclidean distance.

BMI	Age	Sugar	Distance	Rank
33.6	50	1	14.14	2
26.6	30	0	19.72	
23.4	40	0	20.20	
43.1	67	0	27.00	
35.3	23	1	18.92	
35.9	67	1	28.08	
36.7	45	1	8.52	1
25.7	46	0	18.88	3
23.3	29	0	23.09	
31	56	1	20.37	

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that *category (i.e.. Target class)* for which the number of the neighbor is maximum

BMI	Age	Sugar	Distance	Rank
33.6	50	1	14.14	2
26.6	30	0	19.72	
23.4	40	0	20.20	
43.1	67	0	27.00	
35.3	23	1	18.92	
35.9	67	1	28.08	
36.7	45	1	8.52	1
25.7	46	0	18.88	3
23.3	29	0	23.09	
31	56	1	20.37	

Test Example

BMI=43.6, Age=40, Sugar=?

Sugar = 1

Solved Example 2

- The “Restaurant A” sells burger with optional flavors: Pepper, Ginger and Chilly. Everyday this week you have tried a burger (A to E) and kept a record of which you liked.

	Pepper	Ginger	Chilly	Liked
A	True	True	True	False
B	True	False	False	True
C	False	True	True	False
D	False	True	False	True
E	True	False	False	True

- Using Hamming distance, show how the KNN classifier (k=3) would classify (liked or Not liked) with majority voting for burger with below flavor:
  - Pepper: false,
  - Ginger: true,
  - Chilly: true

Reference Video:
https://www.youtube.com/watch?v=T0YkfWssHjk

Step 1: Finding the Hamming distance from query example (Q) to Training examples (A-E),

	Pepper	Ginger	Chilly	Liked	Distance
A	True	True	True	False	1 + 0 + 0 = 1
B	True	False	False	True	1 + 1 + 1 = 3
C	False	True	True	False	0 + 0 + 0 = 0
D	False	True	False	True	0 + 0 + 1 = 1
E	True	False	False	True	1 + 1 + 1 = 3

New Example - Q: 1 0 0

pepper: false, ginger: true, chilly : true

Step 2: Take the K nearest neighbors as per the calculated Hamming distance.

	Pepper	Ginger	Chilly	Liked	Distance	3NN
A	True	True	True	False	1 + 0 + 0 = 1	2
B	True	False	False	True	1 + 1 + 1 = 3	
C	False	True	True	False	0 + 0 + 0 = 0	1
D	False	True	False	True	0 + 0 + 1 = 1	2
E	True	False	False	True	1 + 1 + 1 = 3	

Step-4: Among these k neighbors, count the number of the data points in each category.
Step-5: Assign the new data points to that *category (i.e.. Target clas*

for which the number of the neighbor is maximum

	Pepper	Ginger	Chilly	Liked	Distance	3NN
A	True	True	True	False	1 + 0 + 0 = 1	2
B	True	False	False	True	1 + 1 + 1 = 3	
C	False	True	True	False	0 + 0 + 0 = 0	1
D	False	True	False	True	0 + 0 + 1 = 1	2
E	True	False	False	True	1 + 1 + 1 = 3	

Hence, the classification by majority voting is : **Unliked (i.e False)**

Solved Example 3:

Assume the following training set with two classes, Food and Beverage. Apply KNN with K=3 to classify the new document “turkey soda”

Food : "turkey stuffing"

Food : "buffalo wings"

Beverage : "cream soda"

Beverage : "orange soda"

Solution:

Here, the vocabulary is: **Buffalo, Cream, Orange, Soda, Stuffing, Turkey, Wings**

	Buffalo	Cream	Orange	Soda	Stuffing	Turkey	Wings	Category	Euclidian Distance	Rank with 3NN
D1: “Turkey Stuffing”	0	0	0	0	1	1	0	Food	$(0-1)^2-(1-0)^2 = 2$ $=1.414$	1
D2: “Buffalo Wings”	1	0	0	0	0	0	1	Food	$(1-0)^2-(0-1)^2-(0-1)^2(1-0)^2$ $= 4 =2$	
D3: “Cream Soda”	0	1	0	1	0	0	0	Beverage	$(1-0)^2-(0-1)^2 = 2$ $=1.414$	2
D4: “Orange Soda”	0	0	1	1	0	0	0	Beverage	$(1-0)^2-(0-1)^2 = 2$ $=1.414$	3

Q : “Turkey Soda	0	0	0	1	0	1

Hence, the classification for “Turkey Soda” by majority voting is : **Beverage**

For more examples:

- Reference Videos:
- https://www.youtube.com/watch?v=Vk9lGGODaJA
- https://www.youtube.com/watch?v=HZT0lxD5h6k
- https://www.youtube.com/watch?v=kCNpLbCUo7g

Practical Session:

Perform in https://colab.research.google.com

```
[2]  import matplotlib.pyplot as plt
     from sklearn.neighbors import KNeighborsClassifier
```

```
[3]  x = [4, 5, 10, 4, 3, 11, 14 , 8, 10, 12]
     y = [21, 19, 24, 17, 16, 25, 24, 22, 21, 21]
     classes = [0, 0, 1, 0, 0, 1, 1, 0, 1, 1]
```

```
[4]  data = list(zip(x, y))
     print(data)
```

```
[(4, 21), (5, 19), (10, 24), (4, 17), (3, 16), (11, 25), (14, 24), (8, 22), (10, 21), (12, 21)]
```

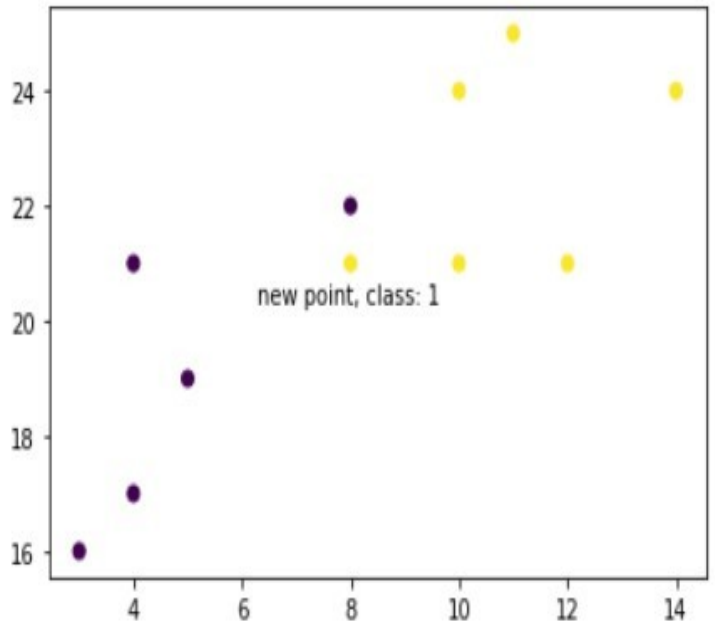
```
[11] knn = KNeighborsClassifier(n_neighbors=5)
     knn.fit(data, classes)
```

```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

```
[9]  new_x = 8
     new_y = 21
     new_point = [(new_x, new_y)]
     prediction = knn.predict(new_point)
     print(prediction)
```

```
     [1]
```

```
[10] plt.scatter(x + [new_x], y + [new_y], c=classes + [prediction[0]])
     plt.text(x=new_x-1.7, y=new_y-0.7, s=f"new point, class: {prediction[0]}")
     plt.show()
```



Assignment:

1. Write the algorithm for K nearest neighbor algorithm. [BIM 2017, Group B]
2. Assume the following training set with two classes, Food and Beverage. Apply KNN with K=3 to classify the new document “turkey soda”. [BIM 2022, Group B]

Food : "turkey stuffing"

Food : "buffalo wings"

Beverage : "cream soda"

Beverage : "orange soda"

5. Bayesian Classifier

5. Bayesian Classifier

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on **Bayes theorem** and used for solving classification problems.
- **It is a probabilistic classifier, which means it predicts on the basis of the probability of an object**.

Bayesian Classifier:

- Baye's Law P(A | B) = P(B | A) P(A)/ P(B)
- Has **high accuracy and speed for large** databases.
- Has **minimum error** rate in comparison to all other classifier

Here

- **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.

Algorithm:

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.

Solved Example 1:

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

Problem: Given below dataset, If the weather is sunny, then the Player should play or not?

Solution: To solve this, first consider the below dataset:

	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

Frequency table for the Weather Conditions:

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	4

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.

Naive Bayesian classifier With Multiple event:

- The formula for Bayes' theorem is given as: $$\mathbf{P(Y|X) = \frac{P(X|Y)\ P(Y)}{P(X)}}$$
- If Variable **X** represents the parameters/features, and X is given as:

$$X = (x_1, x_2, x_3, ..., x_n)$$

- By substituting for **X** and expanding using the chain rule we get,

$$P(y|x_1, ..., x_n) = \frac{P(x_1|y)P(x_2|y)...P(x_n|y)P(y)}{P(x_1)P(x_2)...P(x_n)}$$

- The **denominator can be removed** and proportionality can be injected.

$$P(y|x_1, ..., x_n) \propto P(y) \prod_{i=1}^{n} P(x_i|y)$$

Solved Example 2:

- Estimate conditional probabilities of each attributes {color, legs, height, smelly} for the species classes: {M, H} using the data given in the table.
- Using these probabilities estimate the probability values for the new instance – (Color=Green, legs=2, Height=Tall, and Smelly=No).

No	Color	Legs	Height	Smelly	Species
1	White	3	Short	Yes	M
2	Green	2	Tall	No	M
3	Green	3	Short	Yes	M
4	White	3	Short	Yes	M
5	Green	2	Short	No	H
6	White	2	Tall	No	H
7	White	2	Tall	No	H
8	White	2	Short	Yes	H

Reference Video:
https://www.youtube.com/watch?v=z8K-598fq
So

No	Color	Legs	Height	Smelly	Species
1	White	3	Short	Yes	M
2	Green	2	Tall	No	M
3	Green	3	Short	Yes	M
4	White	3	Short	Yes	M
5	Green	2	Short	No	H
6	White	2	Tall	No	H
7	White	2	Tall	No	H
8	White	2	Short	Yes	H

Calculation of Prior probabilities:

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

(Color=Green, legs=2, Height=Tall, and Smelly=No).

$$P(M) = \frac{4}{8} = 0.5 \quad P(H) = \frac{4}{8} = 0.5$$

Calculation of Conditional probabilities:

Color	M	H
White	2/4	3/4
Green	2/4	1/4

Legs	M	H
2	1/4	4/4
3	3/4	0/4

Height	M	H
Tall	3/4	2/4
Short	1/4	2/4

Smelly	M	H
Yes	3/4	1/4
No	1/4	3/4

$$p(M|New\ Instance) = p(M) * p(Color = Green|M) * p(Legs = 2|M) * p(Height = tall|M) * p(Smelly = no\ |M)$$

$$p(M|New\ Instance) = 0.5 * \frac{2}{4} * \frac{1}{4} * \frac{3}{4} * \frac{1}{4} = 0.0117$$

$$p(H|New\ Instance) = p(H) * p(Color = Green|H) * p(Legs = 2|H) * p(Height = tall|H) * p(Smelly = no\ |H)$$

$$p(H|New\ Instance) = 0.5 * \frac{1}{4} * \frac{4}{4} * \frac{2}{4} * \frac{3}{4} = 0.047$$

$$p(H|New\ Instance) > p(M|New\ Instance)$$

Hence the new instance belongs to Speices H

Assignment:

1. **From the given data set, using Naïve Bayes Classifier, find if a Car (with Red color, SUV type and Domestic origin) will fall in Stolen class or not.**

Example No.	Color	Type	Origin	Stolen?
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	Yes
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	Yes
8	Yellow	SUV	Domestic	No
9	Red	SUV	Imported	No
10	Red	Sports	Imported	Yes

Refer:

1. https://www.kdnuggets.com/2020/06/naive-bayes-algorithm-everything.html
2. https://www.youtube.com/watch?v=fOK9DiKUGYs&t=285s

Assignment:

2. Consider given training data set: Check whether given person does cheat or no using Bayesian classifier. Refund (x, 'yes') ^ Marital status (x, Divorced) ^ Income (x,
<80K). [BIM 2018, Group C]

ID	Refund	Martial Status	Income	Cheat
1	Yes	Single	>80K	No
2	No	Married	>80K	No
3	No	Single	<80K	No
4	Yes	Married	>80K	No
5	No	Divorced	>80K	Yes
6	No	Married	<80K	No
7	Yes	Divorced	>80K	No
8	No	Single	>80K	Yes
9	No	Married	<80K	No
10	No	Single	>80K	Yes

Assignment:

3. Consider given training data set: Check whether given student buys computer or not using Bayesian classifier.

Test data: X:(Age=Youth, Income=Medium, Student=Yes, Credit_Rating=Fair)

ID	Age	Income	Student	Credit_Rating	Buy_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	Middle_aged	Medium	No	Excellent	Yes
13	Middle_aged	High	Yes	Fair	Yes
14	Senior	Medium	No	Excellent	No

Other Solved Examples:

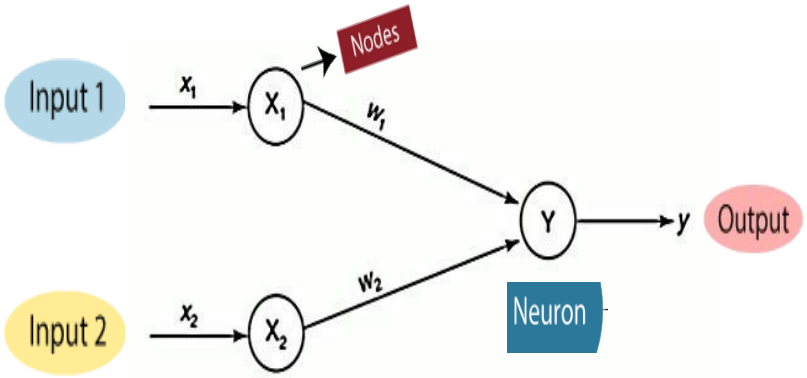
- https://www.youtube.com/watch?v=z8K-598fqSo&t=201s
  - https://www.youtube.com/watch?v=fOK9DiKUGYs&t=285s
  - https://docplayer.net/41403217-Text-categorization-1.html
  - https://www.site.uottawa.ca/~diana/csi4107/example_classi_f_clustering_sol.pdf

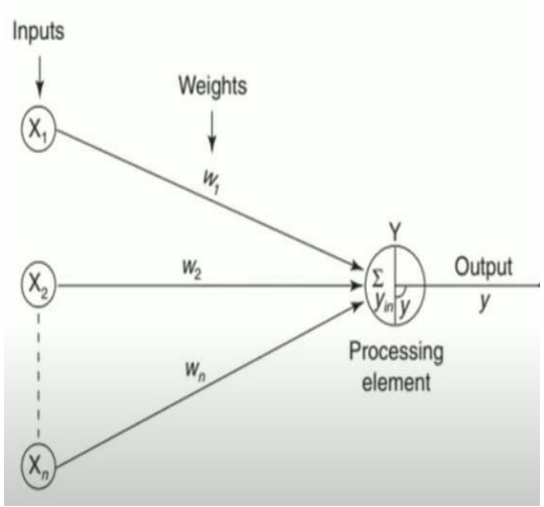
6. Artificial Neural Network Classifier

6. Artificial Neural Network Classifier

- An Artificial Neural Network (ANN) may be defined as an Information-processing model that is inspired by the way biological nervous systems (such as the brain) process information.
- An ANN is composed of a large number of highly interconnected processing units (neurons) working in union to solve specific problems.
- Each neuron is connected with the other by a connection link.
- Each connection link is associated with weights which contain information about the input signal.
- This information is used by the neuron network to solve a particular problem.
- Like human being, ANNs learn by example.
- An ANN is configured for a specific application , such as Spam classification, Face recognition, Pattern recognition through a learning process.

A perceptron of Neural Network





- Input nodes X_1 and X_2 are connected to the output neuron Y, over a weighted interconnection links (W_1 and W_2).
- For the above simple neural net architecture, the net input has to be calculated in the following way:
  - $Y_{in} = X_1W_1 + X_2W_2 + b$
  - $Y = f(Y_{in})$
  - Where,
    - X1, X2 are inputs
    - W1, W2 are weights
    - b is bias
    - f is activation function

Layers in ANN

1. **Input Layer:**

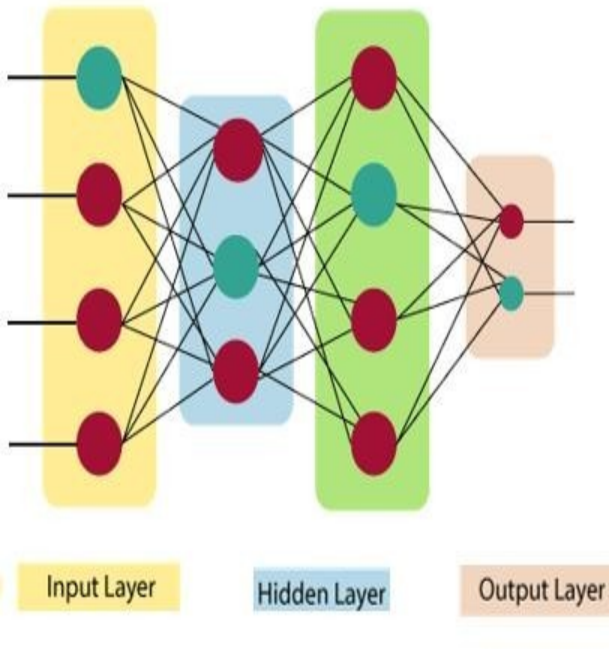
   As the name suggests, it accepts inputs in several different formats provided by the programmer.

2. **Hidden Layer:**

   The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns.

3. **Output Layer:**

   The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.



Solved Example:

1. Find the weights required to perform the following classification using perceptron Network. The target classes are 1 and -1. Assume learning rate (α)=1, initial weights as 0 and bias=0. The activation function is as:

Input					Target
x_1	x_2	x_3	x_4	b	(t)
1	1	1	1	1	1
-1	1	-1	-1	1	1
1	1	1	-1	1	-1
1	-1	-1	1	1	-1

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } y_{in} = 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$$

Also construct the Neural Network with

weights and bias.

Also classify the entity with features vector

[1 1 -1 1], i.e. which class do they belong to.

Solution:

$$y_{in} = b + x_1 w_1 + x_2 w_2 + x_3 w_3 + x_4 w_4$$

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } y_{in} = 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$$

$$\Delta w_1 = \alpha t x_1;$$
$$\Delta w_2 = \alpha t x_2;$$
$$\Delta w_3 = \alpha t x_3;$$
$$\Delta w_4 = \alpha t x_4;$$
$$\Delta b = \alpha t$$

Inputs				Target	Net input	output	Weight changes					Weights				
$(x_1$	x_2	x_3	x_4	(t)	(y_{in})	(y)	$(\Delta w_1$	Δw_2	Δw_3	Δw_4	$\Delta b)$	w_1 (0	w_2 0	w_3 0	w_4 0	b 0)
EPOCH-1																
(1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1
(-1	1	-1	-1	1	-1	-1	-1	1	-1	-1	1	0	2	0	0	2
(1	1	1	-1	-1	4	1	-1	-1	-1	1	-1	-1	1	-1	1	1
(1	-1	-1	1	-1	1	1	-1	1	1	-1	-1	-2	2	0	0	0

Working notes for Row 1 in Epoch 1:

Net input (yin) = $b + x_1w_1 + x_2w_2 + x_3w_3 + x_4w_4$

= 0+ 1.0+1.0+1.0+1.0

= 0

Output (y) = 0 using the activation function

$\Delta w_1 = \alpha t x_1 = 1.1.1 = 1$

$\Delta w_2 = \alpha t x_2 = 1.1.1 = 1$

$\Delta w_3 = \alpha t x_3 = 1$

$\Delta w_4 = \alpha t x_4 = 1$

New weights:

W_1 = old $w_1 + \Delta w_1$ = 0 + 1 = 1

W_2 = old $w_2 + \Delta w_2$ = 0 + 1 = 1

W_3 = old $w_3 + \Delta w_3$ = 0 + 1 = 1

W_4 = old $w_4 + \Delta w_4$ = 0 + 1 = 1

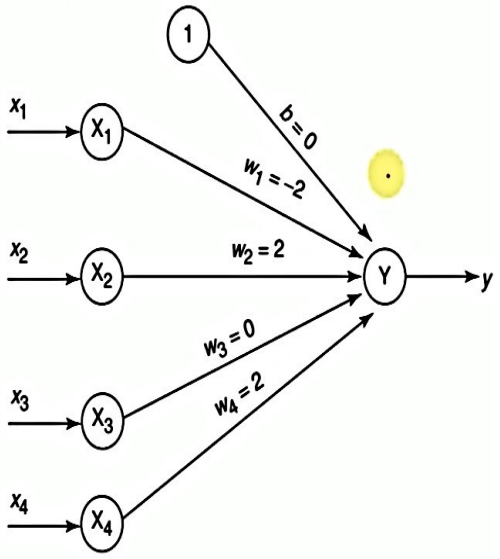
B = old b + Δb = 0+1=1

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } y_{in} = 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$$

$$y_{in} = b + x_1 w_1 + x_2 w_2 + x_3 w_3 + x_4 w_4$$

$$\Delta w_1 = \alpha t x_1; \quad \Delta w_2 = \alpha t x_2; \quad \Delta w_3 = \alpha t x_3; \quad \Delta w_4 = \alpha t x_4; \quad \Delta b = \alpha t$$

Epoch	Inputs				Traget (t)	Net Input (Yin)	Output (Y)	Weight changes					Weights					Remarks
	x1	x2	x3	x4				Δw1	Δw2	Δw3	Δw4	Δb	w1	w2	w3	w4	b	
													0	0	0	0	0	Initialization
Epoch 1	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	
	-1	1	-1	-1	1	-1	-1	-1	1	-1	-1	1	0	2	0	0	2	
	1	1	1	-1	-1	4	1	-1	-1	-1	1	-1	-1	1	-1	1	1	
	1	-1	-1	1	-1	1	1	-1	1	1	-1	-1	-2	2	0	0	0	
Epoch 2	1	1	1	1	1	0	0	1	1	1	1	1	-1	3	1	1	1	
	-1	1	-1	-1	1	3	1	0	0	0	0	0	-1	3	1	1	1	No weight updates because y=t
	1	1	1	-1	-1	3	1	-1	-1	-1	1	-1	-2	2	0	2	0	
	1	-1	-1	1	-1	-2	-1	0	0	0	0	0	-2	2	0	2	0	No weight updates because y=t
Epoch 3	1	1	1	1	1	2	1	0	0	0	0	0	-2	2	0	2	0	No weight updates because y=t
	-1	1	-1	-1	1	2	1	0	0	0	0	0	-2	2	0	2	0	No weight updates because y=t
	1	1	1	-1	-1	-2	-1	0	0	0	0	0	-2	2	0	2	0	No weight updates because y=t
	1	-1	-1	1	-1	-2	-1	0	0	0	0	0	-2	2	0	2	0	No weight updates because y=t

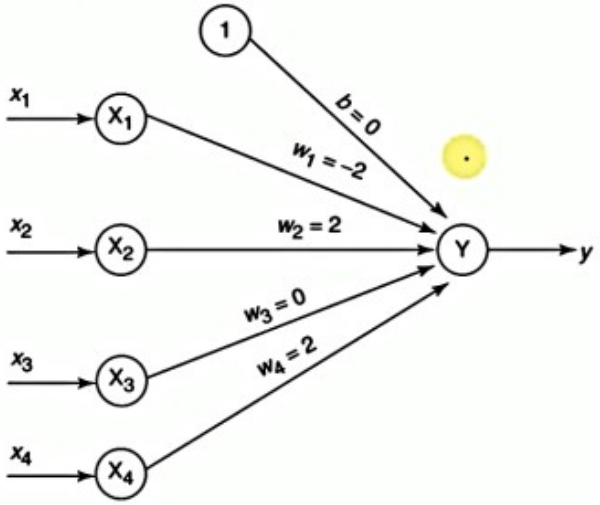


Reference Video:
https://www.youtube.com/watch?v=KKSCmPUyczU

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } y_{in} = 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$$

Classification of given vector [1 1 -1 1]

$$Y_{in} = x_1w_1 + x_2w_2 + x_3w_3 + x_4w_4 + b$$
$$= 1*-2 + 1*2 + -1*0 + 1*2 + 0$$
$$= 2$$



Here $Y_{in} > 0$, Hence using activation function, output y=1

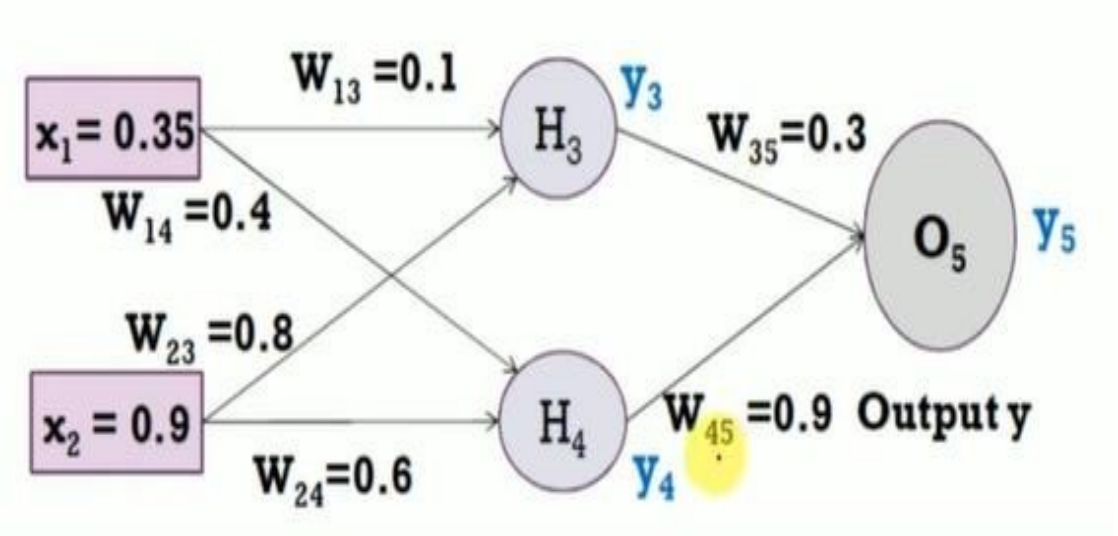
So, the given vector belongs to target (t) 1

Backpropagation in NN

- Backpropagation learns by iteratively processing a dataset of training tuples, comparing the network's prediction for each tuple with the actual known ***target*** value.
- For each training tuple, the weights are modified so as to minimize the mean squared error between the network's prediction and the actual target value.
- These modifications are made in the "backwards" direction, that is , from the output layer, through each hidden layer down to the first hidden layer.
- Hence, named backpropagation.
- Although it is not guaranteed, in general, the weights will eventually converge and the learning process stops.

Assignment:

- Assume that the neurons have a sigmoid activation function, perform a forward pass and a backward pass on the network. Assume that the actual output pf y is 0.5 and learning rate is 1. Perform another forward pass.



Reference video for solution:
https://www.youtube.com/watch?v=tUoUdOdTkRw

Before training the network topology must be designed by:

- *Specifying number of input nodes/units:* Depends upon number of independent variable in data set.
- *Specifying Number of hidden layers:* Generally only one layer is considered in most of theproblem. Two layers can be designed for complex problem. Number of nodes in the hidden layer can be adjusted iteratively.
- *Number of output nodes/units:* Depends upon number of class labels of the data set.
- *Learning rate:* Can be adjusted iteratively.
- *Learning algorithm:* Any appropriate learning algorithm can be selected during training phase.
- *Bias value:* Can be adjusted iteratively.

Advantages

- High tolerance of noisy data
- Classify patterns on which they have not been trained
- Can be used in various applications such as handwriting recognition, image classification, text narration etc.
- Parallelization can be implemented

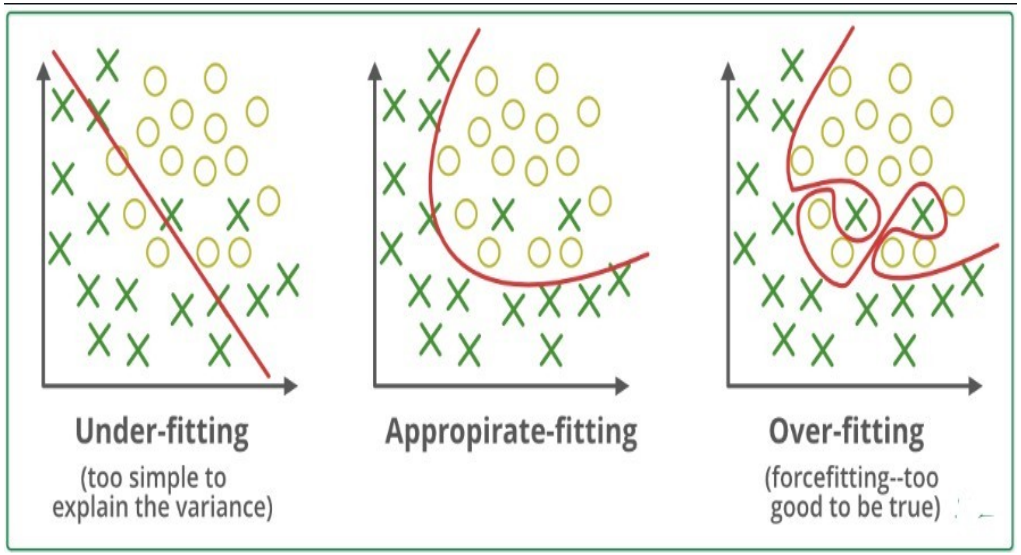
Disadvantages

- Require long training time
- Requires number of parameters whose best value is unknown
- Difficulty to interpret the meaning of weights and hidden network

7. Issues: Overfitting, Validation, Model Comparison

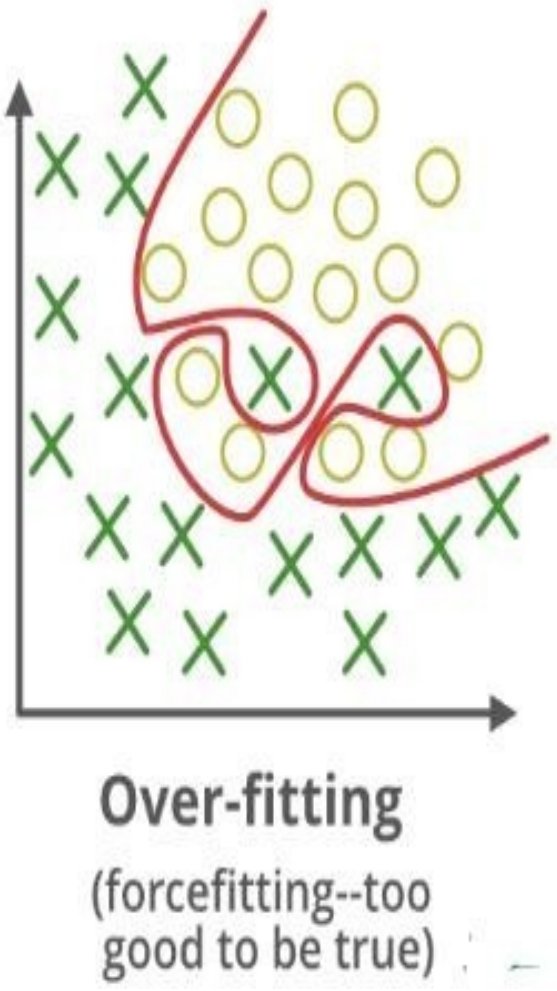
7. Issues: Overfitting and Underfitting

- Overfitting and Underfitting are the two main problems that occur in machine learning and degrade the performance of the machine learning models.
- The main goal of each machine learning model is **to generalize well**. Here **generalization** defines the ability of an ML model to provide a suitable output by adapting the given set of unknown input.
- It means after providing training on the dataset, it can produce reliable and accurate output.
- Hence, the underfitting and overfitting are the two terms that need to be checked for the performance of the model and whether the model is generalizing well or not.



Overfitting

- A model is considered overfitting when it does extremely well on training data but fails to perform on the same level on the validation data.
  - Analogy: like the child who memorized every math problem in the problem book and would struggle when facing problems from anywhere else
- Overfitting occurs when our machine learning model tries to cover all the data points or more than the required data points present in the given dataset.
- Because of this, the model starts caching noise and inaccurate values present in the dataset, and all these factors reduce the efficiency and accuracy of the model.
- The overfitted model has **low bias** and **high variance.**
- The chances of occurrence of overfitting increase as much we provide training to our model. It means the more we train our model, the more chances of occurring the overfitted model.
- Overfitting is the main problem that occurs in supervised learning.

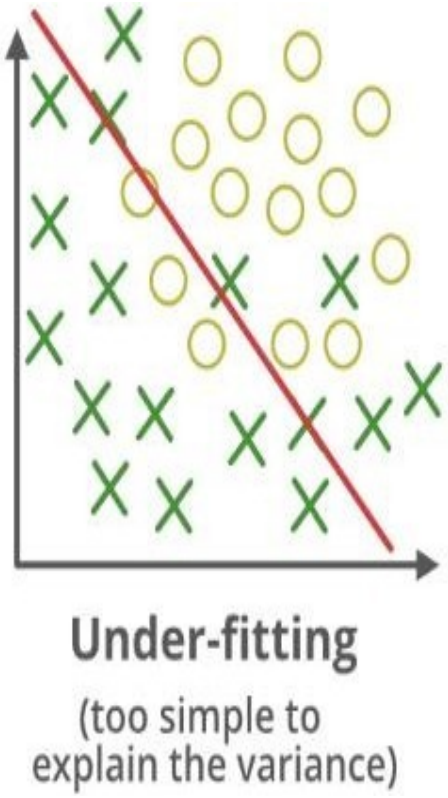


- **Reasons for Overfitting**
  - High variance and low bias.
  - The model is too complex.
  - The size of the training data.

- **How to reduce overfitting?**
  - Cross-Validation
  - Training with more data
  - Removing features
  - Early stopping during the training phase (have an eye over the loss over the training period as soon as loss begins to increase stop training).
  - Regularization (Ridge Regularization and Lasso Regularization)
  - Ensembling
  - Reduce model complexity.
  - Use dropout for neural networks to tackle overfitting.

Underfitting

- Underfitting occurs when our machine learning model is not able to capture the underlying trend of the data.
  - Analogy: like the child who learned only addition and was not able to solve problems related to other basic arithmetic operations both from his math problem book and during the math exam.
- To avoid the overfitting in the model, the fed of training data can be stopped at an early stage, due to which the model may not learn enough from the training data.
- As a result, it may fail to find the best fit of the dominant trend in the data.
- In the case of underfitting, the model is not able to learn enough from the training data, and hence it reduces the accuracy and produces unreliable predictions.
- An underfitted model has high bias and low variance.



- **Reasons for underfitting**
  - High bias and low variance
  - The size of the training dataset used is not enough.
  - The model is too simple.
  - Training data is not cleaned and also contains noise in it.

- **How to reduce underfitting?**
  - Increase the number of features,performing feature engineering
  - By increasing the training time of the model, i.e. Increase the number of epochs or increase the duration of training to get better results.
  - Increase model complexity
  - Remove noise from the data.

Goodness of Fit

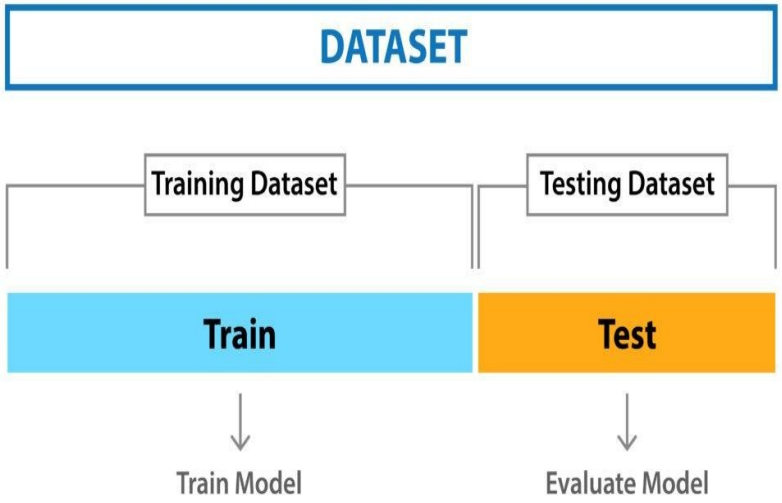
- The "Goodness of fit" term is taken from the statistics, and the goal of the machine learning models to achieve the goodness of fit. In statistics modeling, *it defines how closely the result or predicted values match the true values of the dataset.*
- The model with a good fit is between the underfitted and overfitted model,and ideally, it makes predictions with 0 errors, but in practice, it is difficult to achieve it.
- As when we train our model for a time, the errors in the training data go down,and the same happens with test data. But if we train the model for a long duration, then the performance of the model may decrease due to the overfitting, as the model also learn the noise present in the dataset. The errors in the test dataset start increasing, *so the point, just before the raising of errors, is the good point, and we can stop here for achieving a good model.*
- There are two other methods by which we can get a good point for our model, which are the **resampling method** to estimate model accuracy and **validation dataset**.

Validation

- Validation is the process of evaluating the model using the Training dataset.
- It is done by a resampling technique called cross validation.
- Different validation techniques are:
    1. The hold-out method
    2. K-fold cross validation
    3. Leave-one-Out Cross-Validation
    4. Stratified K-Fold Cross-Validation

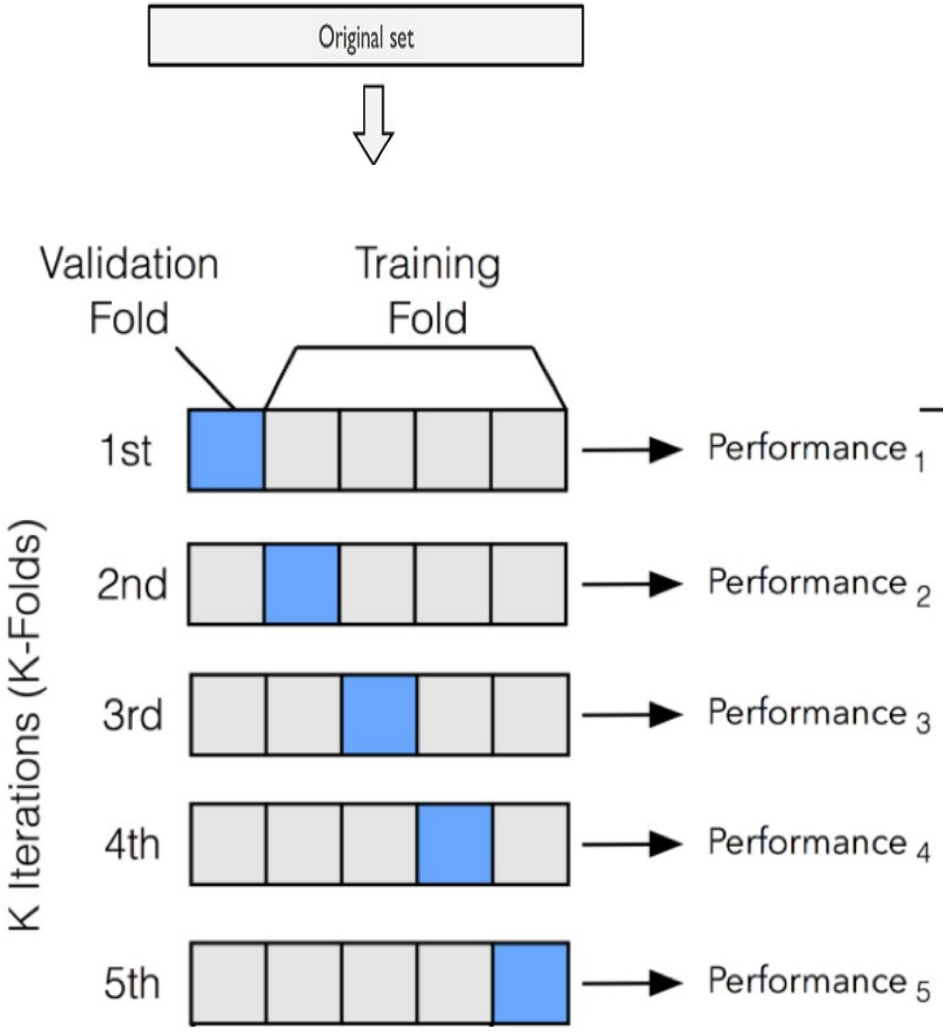
1. Hold out Validation approach

- To avoid the re-substitution error, the data is split into two different datasets labeled as a training and a testing dataset.
- This can be a 60/40 or 70/30 or 80/20 split.
- Then, we train the model on the training data and then see the performance on the unseen data.

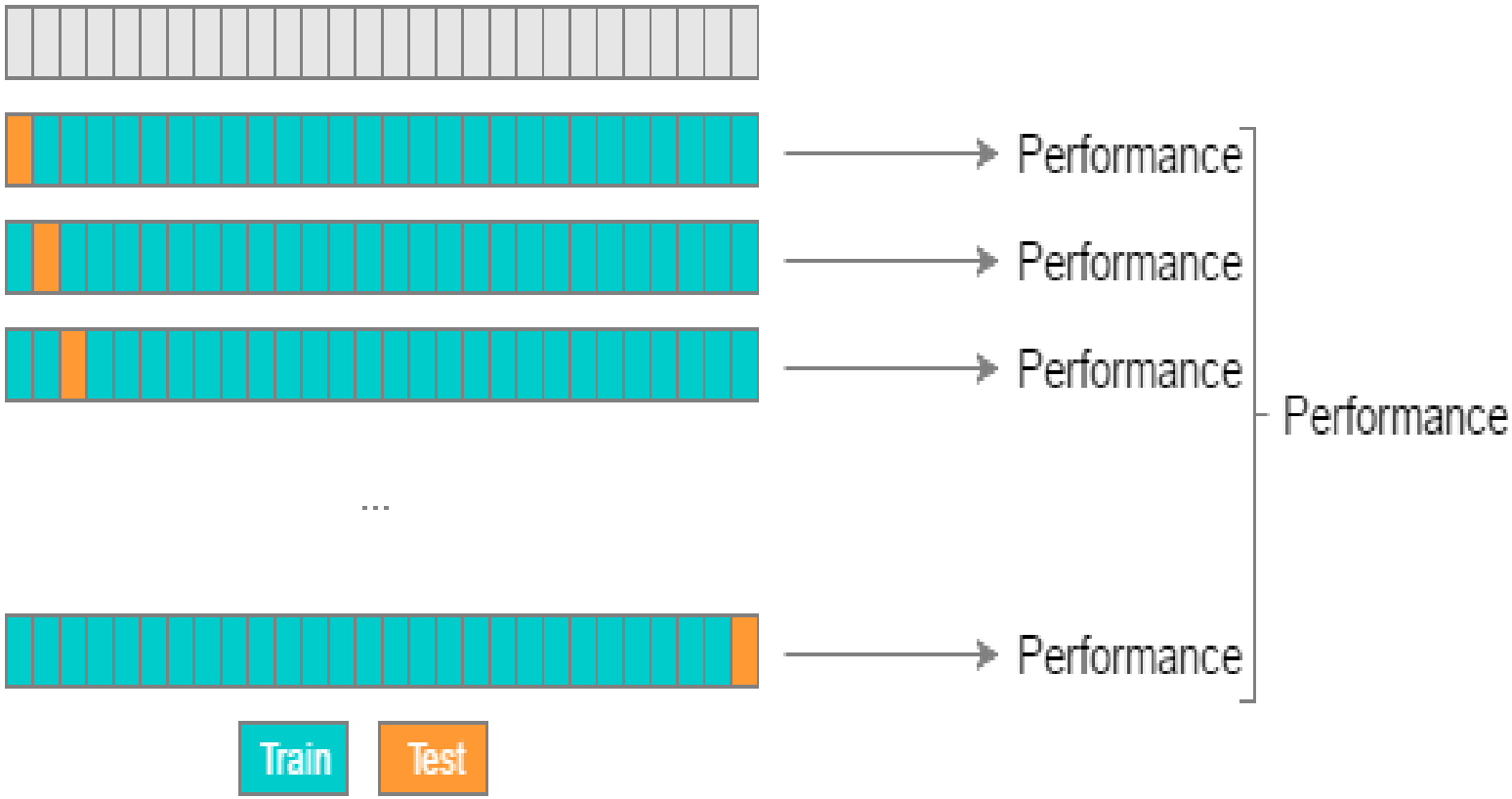


2. K-fold Cross validation

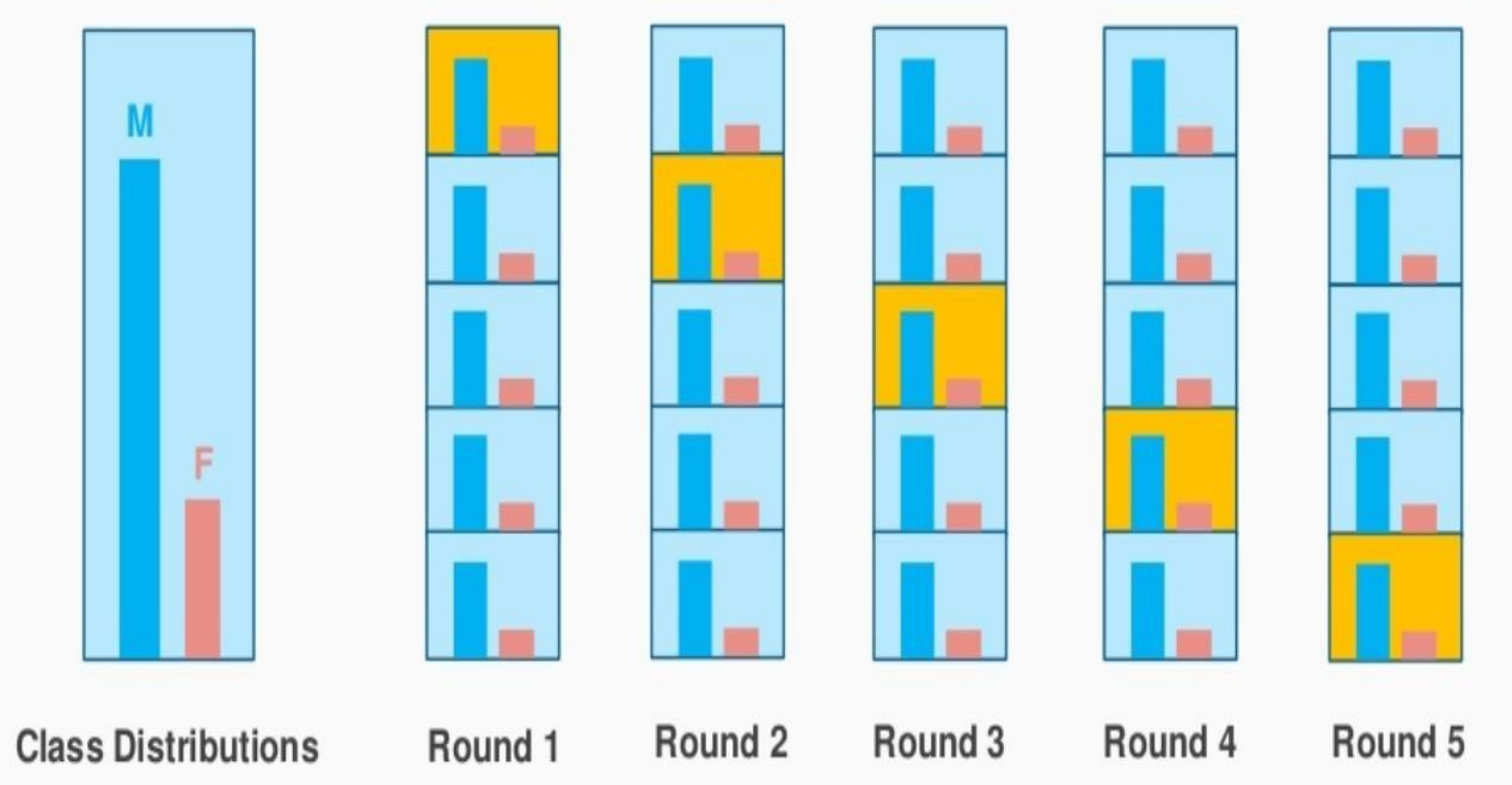
- In this approach, we divide the data set into k number of subsets and the holdout method is repeated k number of times.
- It tries to address the problem of the holdout method.



3. Leave one out Cross-validation (LOOCV)



4. Startified K-fold Cross Validation



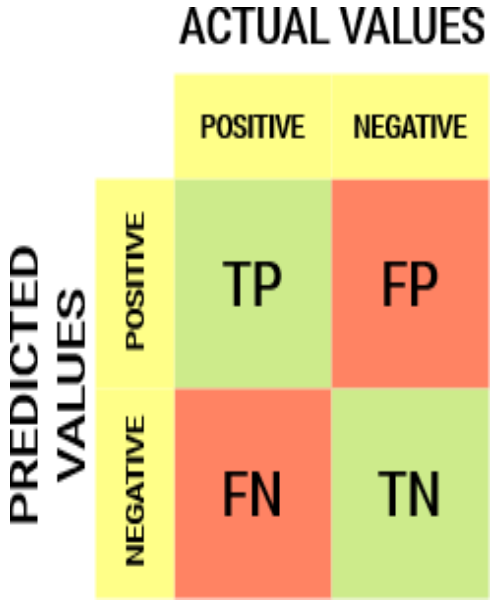
Measures for Model Comparison

1. Confusion Matrix
2. ROC
3. AUC

1. Confusion Matrix

- A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the total number of target classes.

For a binary classification problem, we would have a 2 x 2 matrix, as shown aside, with 4 values:

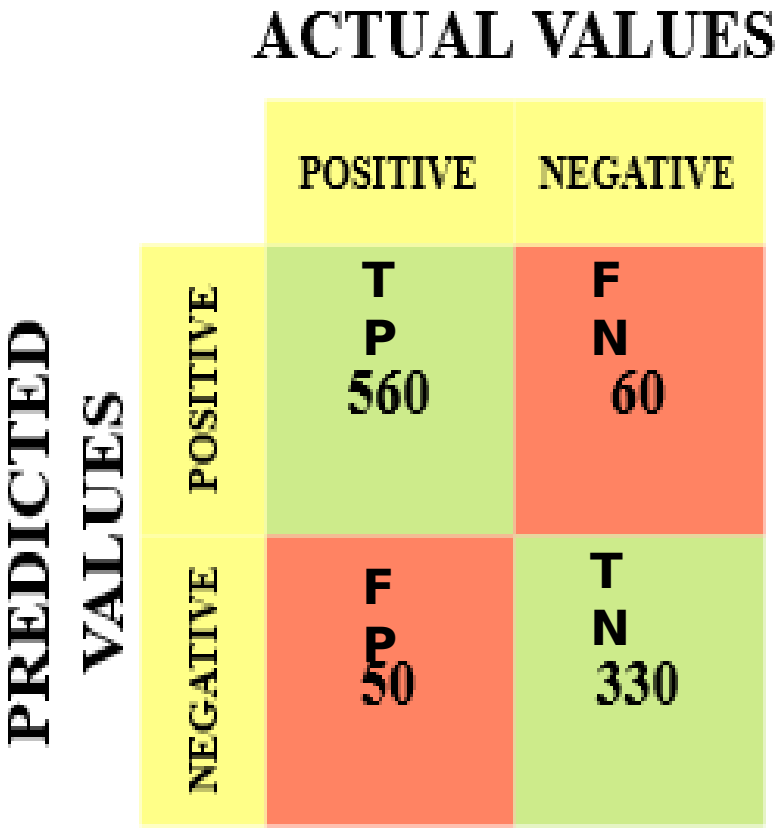


- Also sometimes called a classification matrix, is used to assess the prediction accuracy of a model.
- It measures whether a model is confused or not, that is, whether the model is making mistakes in its predictions or not.

- **True Positive (TP) :** the no. of positive tuples that were correctly predicted by the classifier.
- **True Negative (TN):** the no. of negative tuples that were correctly predicted by the classifier.
- **False Positive (FP):** The no. of Negative tuples that were incorrectly predicted as Positive (or YES)
- **False Negative (FN):** The no. of Positive tuples that were incorrectly predicted as Negative (or NO)

Confusion Matrix

- True Positive (TP) = 560, meaning the model correctly classified 560 positive class data points.
- True Negative (TN) = 330, meaning the model correctly classified 330 negative class data points.
- False Positive (FP) = 60, meaning the model incorrectly classified 60 negative class data points as belonging to the positive class.
- False Negative (FN) = 50, meaning the model incorrectly classified 50 positive class data points as belonging to the negative class.



Metrics

- **Accuracy**: Accuracy measures how often the model is correct.
- **Error rate:** It defines how often the model gives the wrong predictions.
- **Precision:** tells us how many of the correctly predicted cases actually turned out to be positive.
- **Recall:** It is defined as the out of total positive classes, how our model predicted correctly. The recall must be as high as possible.
- **F1-score:** is a harmonic mean of Precision and Recall,

Accuracy = $\frac{TP}{P+N}$

Error Rate = $\frac{FP+FN}{P+N}$

Precision = $\frac{TP}{TP+FP}$

Recall = $\frac{TP}{TP+FN}$

F1-Score = $\frac{2*Recall*Precision}{Recall+Precision}$

Measures

- **Accuracy** is not always the best measure of the quality of the classification model.
- It is especially true for the real-world problems where the distribution of classes is unbalanced.
- For example, if the problem is classification of healthy persons from those with the disease.
- In many cases, the medical database for training and testing will contain mostly healthy persons (99%), and only small percentage of people with disease (about 1%).
- In that case, no matter how good the accuracy of a model is estimated to be,

- In practice, several measures are developed, and some of the best known are as :
  - **Accuracy** = $\frac{TP+TN}{P+N}$
  - **Error Rate** = $\frac{FP+FN}{P+N}$
  - **Precision** = $\frac{TP}{TP+FP}$
  - **Recall** = $\frac{TP}{TP+FN}$
  - **Specificity** = $\frac{TN}{FP+TN}$
  - **F1-Score** = $\frac{2*Recall*Precision}{Recall+Precision}$

Solved example

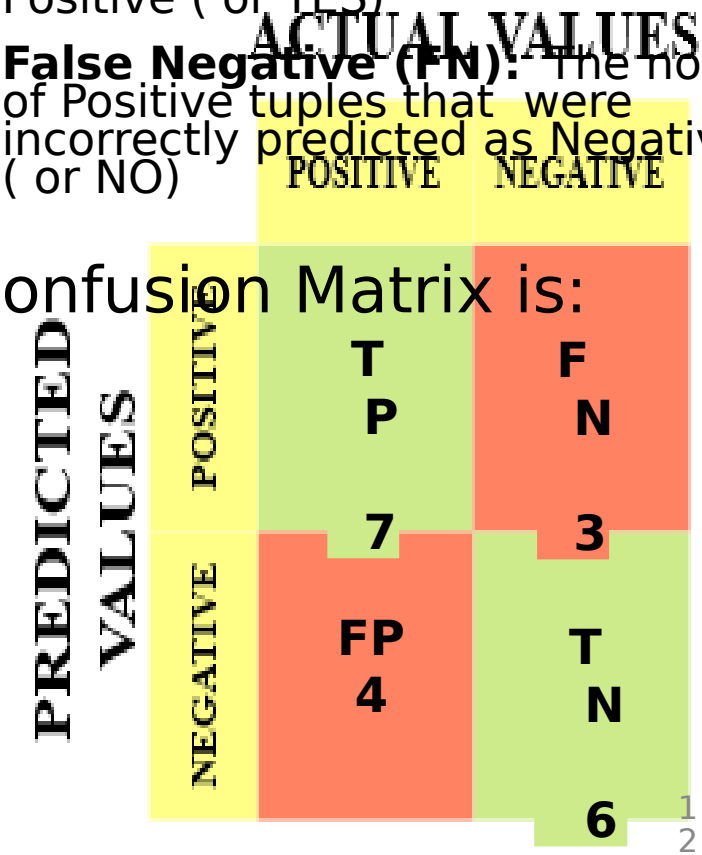
- Calculate the measures after constructing a confusion matrix for the below output of a model.

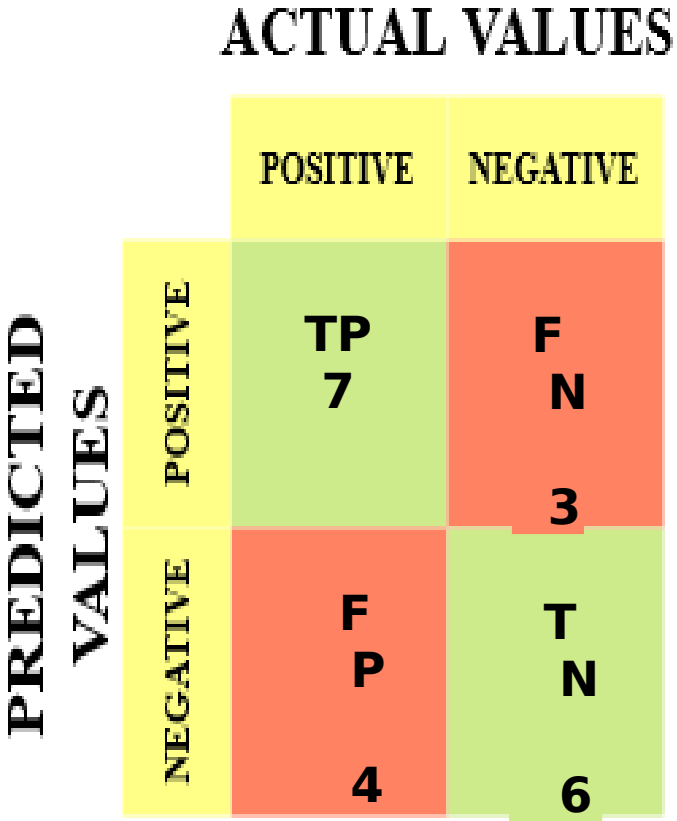
review	true_label	predicted_label
sample review 1	0	1
sample review 2	0	0
sample review 3	1	0
sample review 4	1	1
sample review 5	1	1
sample review 6	0	1
sample review 7	1	1
sample review 8	0	1
sample review 9	0	0
sample review 10	0	0
sample review 11	1	0
sample review 12	1	1
sample review 13	0	0
sample review 14	1	1
sample review 15	0	0
sample review 16	0	1
sample review 17	1	0
sample review 18	1	1
sample review 19	1	1
sample review 20	0	0

review	true_label	predicted_label	TP/TN/FP/FN
sample review 1	0	1	FP
sample review 2	0	0	TN
sample review 3	1	0	FN
sample review 4	1	1	TP
sample review 5	1	1	TP
sample review 6	0	1	FP
sample review 7	1	1	TP
sample review 8	0	1	FP
sample review 9	0	0	TN
sample review 10	0	0	TN
sample review 11	1	0	FN
sample review 12	1	1	TP
sample review 13	0	0	TN
sample review 14	1	1	TP
sample review 15	0	0	TN
sample review 16	0	1	FP
sample review 17	1	0	FN
sample review 18	1	1	TP
sample review 19	1	1	TP
sample review 20	0	0	TN

- **True Positive (TP) :** the no. of positive tuples that were correctly predicted by the classifier.
- **True Negative (TN):** the no. of negative tuples that were correctly predicted by the classifier.
- **False Positive (FP):** The no. of Negative tuples that were incorrectly predicted as Positive (or YES)
- **False Negative (FN):** The no. of Positive tuples that were incorrectly predicted as Negative (or NO)

- Confusion Matrix is:





Accuracy $= \frac{TP+TN}{P+N} = \frac{7+6}{11+9} = 0.65 = 65\%$

Error Rate $= \frac{FP+FN}{P+N} = \frac{4+3}{11+9} = 0.35$

Precision $= \frac{TP}{TP+FP} = \frac{7}{7+4} = 0.6333$

Recall $= \frac{TP}{7} = \frac{}{} = 0.7$

F1-Score $= \frac{2*Recall*Precision}{Recall+Precision} = \frac{2*0.7*0.63}{0.7+0.63} = 0.6649$

Receiver Operating Characteristics (ROC)

- An **ROC curve** (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:
  - True Positive Rate (Sensitivity)

$$TPR = \frac{TP}{TP + FN}$$

  - False Positive Rate (Specificity)

$$FPR = \frac{FP}{FP + TN}$$

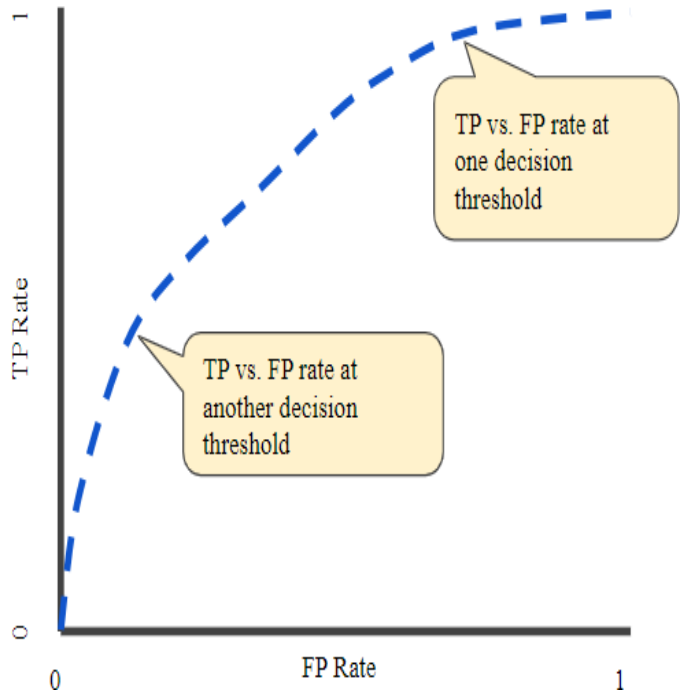


Figure : TP vs. FP rate at different classification threshold

Area Under Curve (AUC)

- **AUC** stands for "Area under the ROC Curve."
- AUC measures the entire two-dimensional area underneath the entire ROC curve.
- AUC provides an aggregate measure of performance across all possible classification thresholds.

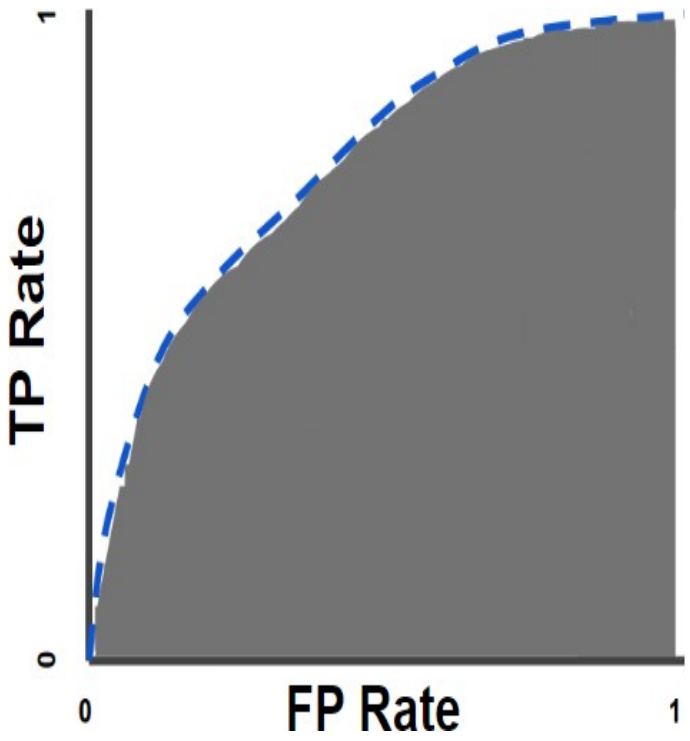


Figure: AUC (Area under the ROC Curve).

Exercise:

1. What Is Machine learning? Write its importance.
2. Differentiate between supervised and unsupervised learning.
3. What is classification? Differentiate between binary and multiple class classification.
4. Mention the steps in classification process.
5. What are the issues regarding classification? Mention them.
6. Differentiate classification and predication.

Exercise:

11. Write the formula to calculate conditional entropy.
12. What is Gain ration? Write its formula.
13. What is Gini index? Write its formula.
14. What do you mean by pruning of Decision tree? Why pruning is required?
15. Mention the advantages and disadvantages of Decision Tree.
16. Create a decision tree using Information gain for below dataset.
17. Explain Rule based classifier with an example.
18. Define Coverage and Accuracy measures of a rule.

Exercise:

21. What is Hamming distance? Compute the Hamming distance between two data point (blue, red) and (blue, black).
22. Mention the KNN algorithm.
23. What are the advantages and disadvantages of KNN classifier?
24. Define Bayes theorem and Naïve Bayes Classifier.
25. What do you mean by prior probability and conditional probability?
26. Define ANN classifier.
27. Define a perceptron.
28. What is activation function? What is the use of activation function in the output layer?

Exercise:

31. What do you mean by back propagation in machine learning?
32. What do you mean by overfitting and underfitting in model evaluation?
33. Write the possible causes of overfitting.
34. What are the possible ways to reduce overfitting? Mention them.
35. Write the possible causes of underfitting.
36. What are the possible ways to reduce underfitting? Mention them.
37. What is a generalized model?
38. What is validation? How do you validate the classification model?
39. What is K-fold cross validation? Illustrate with a figure.
40. Define confusion matrix with a relevant example.
41. Define the terms: accuracy, error rate, precision,

Exercise:

44. Given the following training set with class “Play tennis” for decision tree [illegible] information gain

Day	Outlook	Temperature	Humidity	Wind	Play Tennis (Class
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	overcast	Hot	High	weak	Yes
D4	Rain	mild	High	Weak	Yes
D5	rain	Cool	Normal	Weak	Yes

45. Construct a decision tree using ID3 algorithm

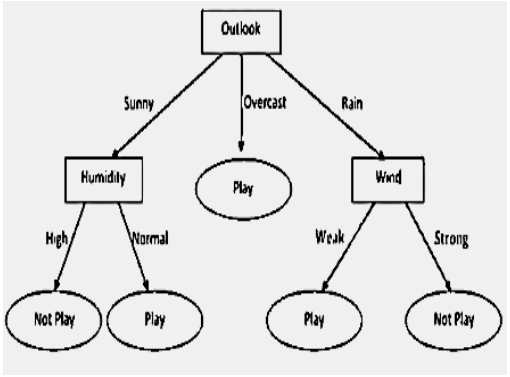
Instance	a1	a2	a3	Classification
1	True	Hot	High	No
2	True	Hot	High	No
3	False	Hot	High	Yes
4	False	Cool	Normal	Yes
5	False	Cool	Normal	Yes
6	True	Cool	High	No
7	True	Hot	High	No
8	True	Hot	Normal	Yes
9	False	Cool	Normal	Yes
10	False	Cool	High	Yes

Exercise

46. Construct a decision tree using CART algorithm for the below

Weekend	Weather	Parents	Money	Decision
W1	Sunny	Yes	Rich	Cinema
W2	Sunny	No	Rich	Tennis
W3	Windy	Yes	Rich	Cinema
W4	Rainy	Yes	Poor	Cinema
W5	Rainy	No	Rich	Stay In
W6	Rainy	Yes	Poor	Cinema
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W9	Windy	Yes	Rich	Cinema
W10	Sunny	No	Rich	Tennis

47. Extract rules form below Decisi



Exercise:

48. Apply K Nearest Neighbor classifier to predict the diabetic patient with the given features (BMI, Age). The target label is Sugar. The test example is: BMI= 43.6 and Age= 40, Sugar= ?. Assume K=3

BMI	Age	Sugar
33.6	50	1
26.6	30	0
23.4	40	0
43.1	67	0
35.3	23	1
35.9	67	1
36.7	45	1
25.7	46	0
23.3	29	0
31	56	1

49. Assume the followir
training set with two
classes, Food and Be
Apply KNN with K=3 to
classify the new document

Food : "turkey stuffing"
Food : "buffalo wings"
Beverage : "cream soda"
Beverage : "orange soda"

Exercise:

50.

- Estimate conditional probabilities of each attributes {color, legs, height, smelly} for the species classes: {M, H} using the data given in the table.
- Using these probabilities estimate the probability values for the new instance – (Color=Green, legs=2, Height=Tall, and Smelly=No).

No	Color	Legs	Height	Smelly	Species
1	White	3	Short	Yes	M
2	Green	2	Tall	No	M
3	Green	3	Short	Yes	M
4	White	3	Short	Yes	M
5	Green	2	Short	No	H
6	White	2	Tall	No	H
7	White	2	Tall	No	H
8	White	2	Short	Yes	H

Exercise:

51. Find the weights required to perform the following classification using perceptron Network. The target classes are 1 and -1. Assume learning rate (α)=1, initial weights as 0 and bias=0. The activa

$$y = f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} > 0 \\ 0 & \text{if } y_{in} = 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$$ is as:

- Also construct the Neural Network and bias.

- Also classify the entity with features vector [1 1 -1 1] , i.e. which class do they belong to.

Input					Target
x_1	x_2	x_3	x_4	b	(t)
1	1	1	1	1	1
-1	1	-1	-1	1	1
1	1	1	-1	1	-1
1	-1	-1	1	1	-1

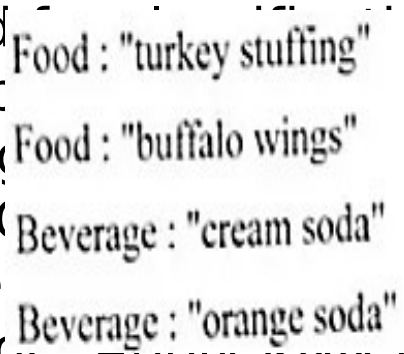
Exercise:

52. Calculate all the required metrics after constructing a confusion matrix for the below output of a model.

review	true_label	predicted_label
sample review 1	0	1
sample review 2	0	0
sample review 3	1	0
sample review 4	1	1
sample review 5	1	1
sample review 6	0	1
sample review 7	1	1
sample review 8	0	1
sample review 9	0	0
sample review 10	0	0
sample review 11	1	0
sample review 12	1	1
sample review 13	0	0
sample review 14	1	1
sample review 15	0	0
sample review 16	0	1
sample review 17	1	0
sample review 18	1	1
sample review 19	1	1
sample review 20	0	0

Exam questions:

1. What is attribute selection measure in decision tree? [BIM 2022, Group A]
2. What is entropy? [BIM 2017, Group A]
3. How do you compare the accuracy of two classifier? [BIM 2018, Group A]
4. How do you compare two classifiers? [BIM 2021, Group A]
5. How do you validate the classification model? [BIM 2022, Group A]
6. Explain classification and prediction. How Decision Tree can be used for classification of elements (data)? [BIM 2018, Group
7. Write the algorithm for nearest neighbor algorithm. [BIM 2017, G
8. Assume the following training set with two classes, Food and Beverage. Apply KNN with K=3 to classify the new document "turkey soda". [BIM 2022, Group B]



Exam questions:

9. What might be the cause of overfitting in classifier? [BIM 2022, Group C]
10. Given the following training set with class "Play tennis" for decision tree classifier, calculate information gain for attribute

Day	Outlook	Temperature	Humidity	Wind	Play Tennis (Class
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	overcast	Hot	High	weak	Yes
D4	Rain	mild	High	Weak	Yes
D5	rain	Cool	Normal	Weak	Yes

11. Consider given training data set: Check whether given person does cheat or no using Bayesian classifier. Refund (x, 'yes') ^ Marital status (x, Divorced) ^ Income (x, <80K). [BIM 2018, Group C]

ID	Refund	Martial Status	Income	Cheat
1	Yes	Single	>80K	No
2	No	Married	>80K	No
3	No	Single	<80K	No
4	Yes	Married	>80K	No
5	No	Divorced	>80K	Yes
6	No	Married	<80K	No
7	Yes	Divorced	>80K	No
8	No	Single	>80K	Yes
9	No	Married	<80K	No
10	No	Single	>80K	Yes

Exam questions:

12. Consider the 14 training datasets with 9 positive and 5 negative classes. Suppose one of the attributes is Wind, which have values Weak and Strong. There are 8 occurrences of Weak winds and 6 occurrences of Strong winds. For the weak winds, 6 are positive and 2 are negative. For the strong winds, 3 are positive and 3 are negative. Calculate the information gain of wind. [BIM 2021, Group B]

End of chapter

Overfitting

- Overfitting generally occurs when a model is excessively complex, such as having **too many parameters relative** to the number of observations.
- A model which has been overfit will generally have poor predictive performance.
- Overfitting depends not only on the number of parameters and data but also the conformability of the model structure.
- In order to avoid overfitting, it is necessary to use additional techniques (e.g. crossvalidation, pruning (Pre or Post), model comparison.

Reason

- Noise in training data.
- Incomplete training data.
- Flaw in assumed theory

Under fitting:

- It refers to a model that can neither model the training data nor generalize to new data.
- An underfit machine learning model is not a suitable model and will be obvious as it will have poor performance on the training data.
- Under fitting is often not discussed as it is easy to detect given a good performance metric.
- The remedy is to move on and try alternate machine learning algorithms.

Validation

- Validation techniques are motivated by two fundamental problems in pattern recognition: **model selection and performance estimation**

Validation Approaches:

- One approach is to use the entire training data **to select our classifier and estimate the error rate**, but the final model will normally overfit the training data.
- A much better approach is to split the training data into disjoint subsets cross validation (The Holdout Method)

Cross Validation (The holdout method)

- Data set divided into two groups. Training set: used to train the classifier and Test set: used to estimate the error rate of the trained classifier
- Total number of examples = Training Set +Test Set

K-Fold Cross-Validation

- K-Fold Cross validation is similar to Random Sub sampling.
- Create a K-fold (5 say) partition of the dataset, For each of K experiments, use K-1 (4) folds for training and the remaining one (1) for testing.
- The advantage of K-Fold Cross validation is that all the examples in the dataset are eventually used for both training and testing.
- The true error is estimated as the average error rate.

Model Comparison:

Models can be evaluated based on the output using different method :

1. Confusion Matrix
2. ROC Analysis
3. Others such as: Gain and Lift Charts, K-S Charts

Confusion Matrix (Contingency Table):

- A confusion matrix contains information about actual and predicted classifications done by classifier.
- Performance of such system is commonly evaluated using data in the matrix.
- It is also known as a contingency table or an error matrix, is a specific table layout that allows visualization of the performance of an algorithm.
- Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class.

	Predicted Positive	Predicted Negative
Positive Examples	True Positive (TP)	False Negative (FN)
Negative Examples	False Positive (FP)	True Negative (TN)

- Accuracy: (TP + TN) / Total data count
- Positive Rate (TPR): TP / (TP +TN) True
- Negative Rate (TNR): TN / (TP +TN) False
- Positive Rate (FPR): FP / (FP +FN) False
- Negative Rate (FNR): FN / (FP +FN)

- True positive (TP) refer to the positive tuples that were correctly labeled by the classifier.
- True Negative (TN) are the negative tuples that were correctly labeled by the classifier.
- A false positive (FP) occurs when the outcome is incorrectly predicted as yes (or positive) when it is actually no (negative)
- A false negative (FN) occurs when the outcome is incorrectly predicted as negative when it is actually positive

ROC curve:

- ROC Analysis - Receiver Operating Characteristic (ROC), or ROC curve, is a graphical plot that illustrates the performance of a **binary classifier** system as its discrimination threshold is varied.
- The curve is created by plotting the true positive rate against the false positive rate at various threshold settings.
- The ROC curve is thus the sensitivity as a function of fall-out.
- In general, if the probability distributions for both detection and false alarm are known, the ROC curve can be generated by plotting the cumulative distribution function (area under the probability distribution from to) of the detection probability in the y-axis versus the cumulative distribution function of the false-alarm probability in x-axis.
- ROC analysis provides tools to select possibly optimal models and to discard suboptimal ones independently from (and prior to specifying) the cost context or the class distribution.
- ROC analysis is related in a direct and natural way to cost/benefit analysis of diagnostic decision making.

- On the ROC curve, we move right and plot a point. This process is repeated for each of the test tuples, each time moving up on the curve for a true positive or toward the right for a false positive. To assess the accuracy of a model, we can measure the area under the curve. The closer the area is to 0.5, the less accurate the corresponding model is. A model with perfect accuracy will have an area of 1.0.