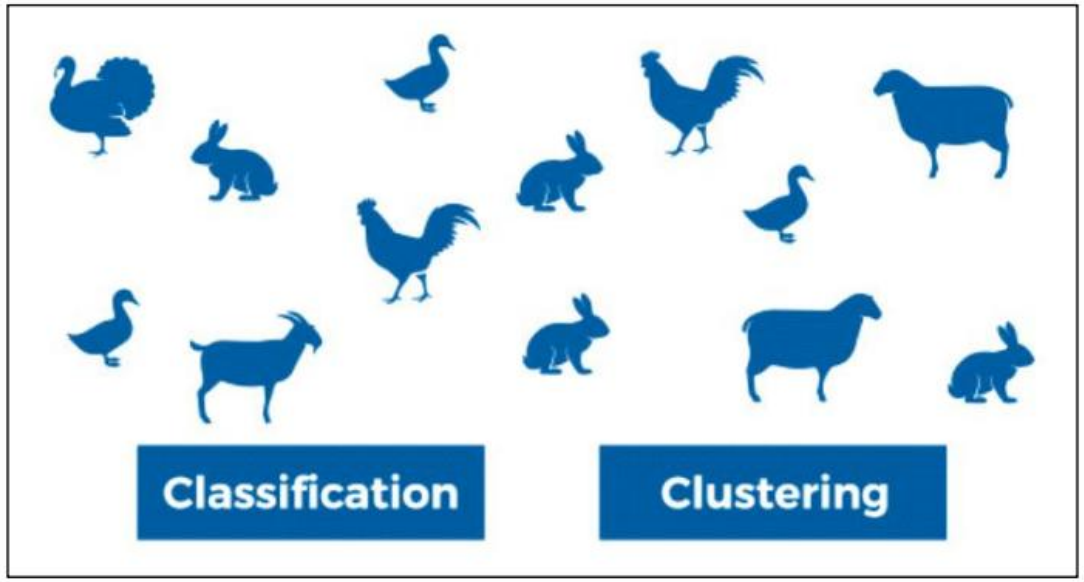
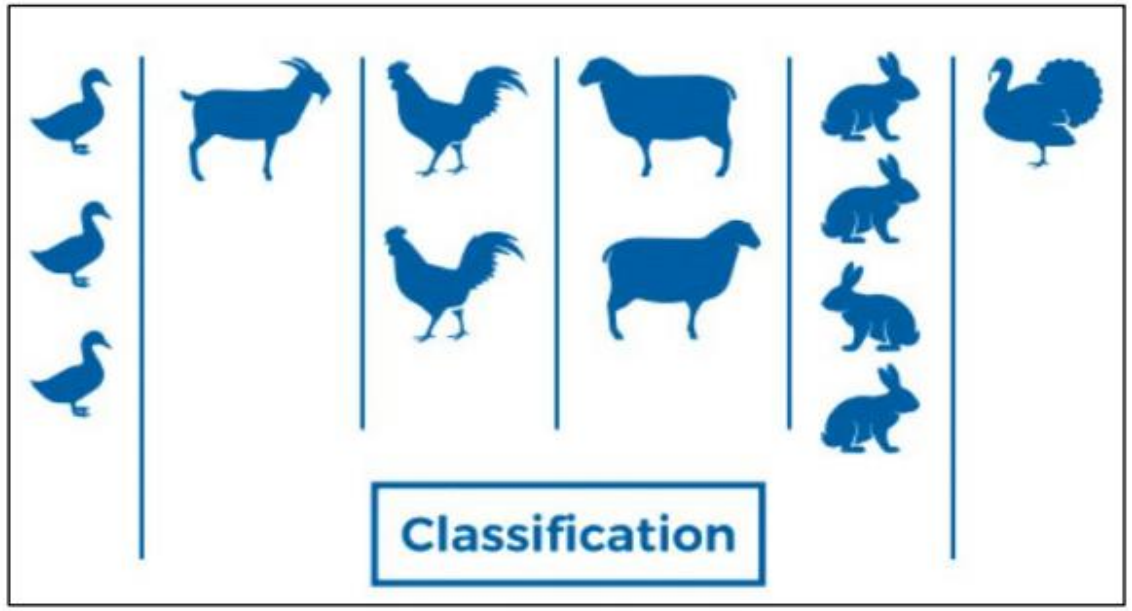
Data Mining: Classification

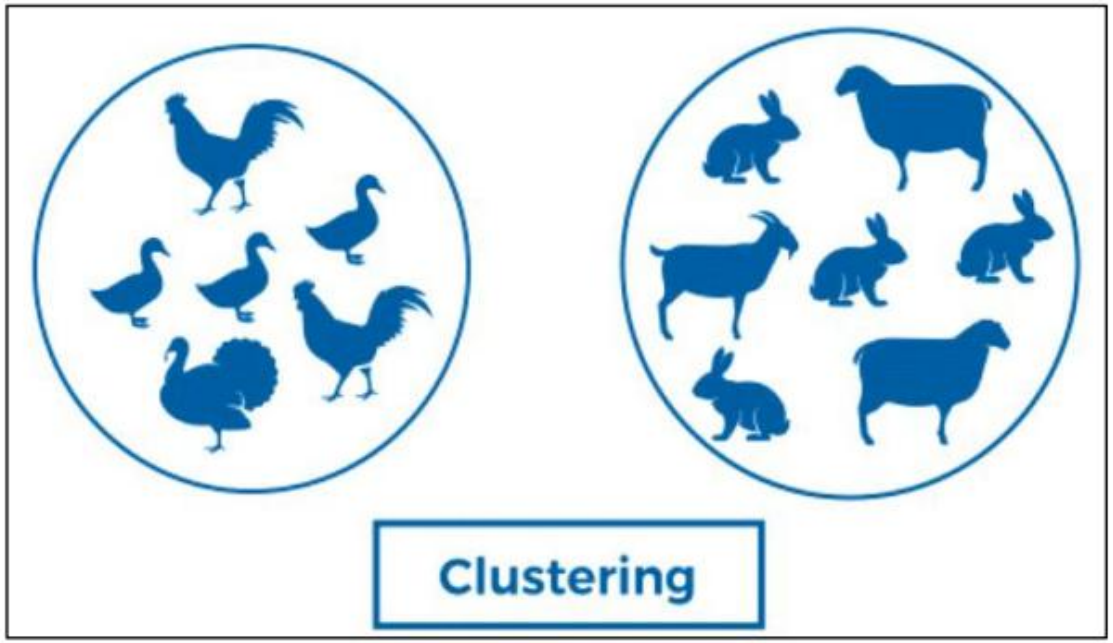
Classification and Clustering

- Classification and clustering are two methods of pattern identification with some similarity and disimilarity.
- **Classification** uses predefined classes in which objects are assigned.
- **Clustering** identifies similarities between objects, which it groups according to those characteristics in common and which differentiate them from other groups of objects. These groups are known as "clusters"

Classification and Clustering: Example







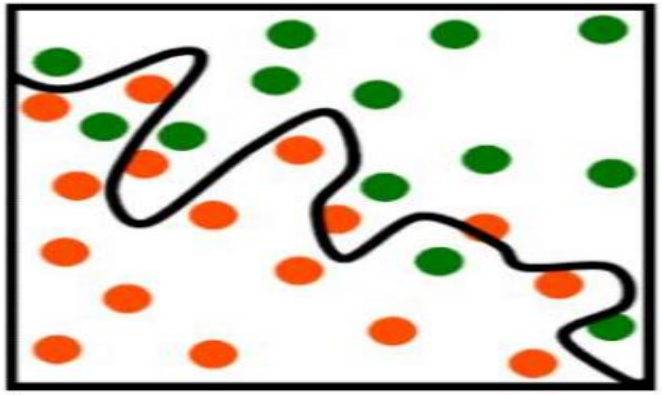
Classification

- Classification is the Data mining technique used to predict group membership for data instances.
- There are two ways to assign a new value to a given class
  - Crispy Classification
  - Probabilistic Classification

➢Crispy Classification: Given an input, the classifier excatly returns its class label.

➢Probabilistic Classification: Given an input, the classifier returns its probabilities to belong to each class. This is useful when some some mistakes can be more costly than others

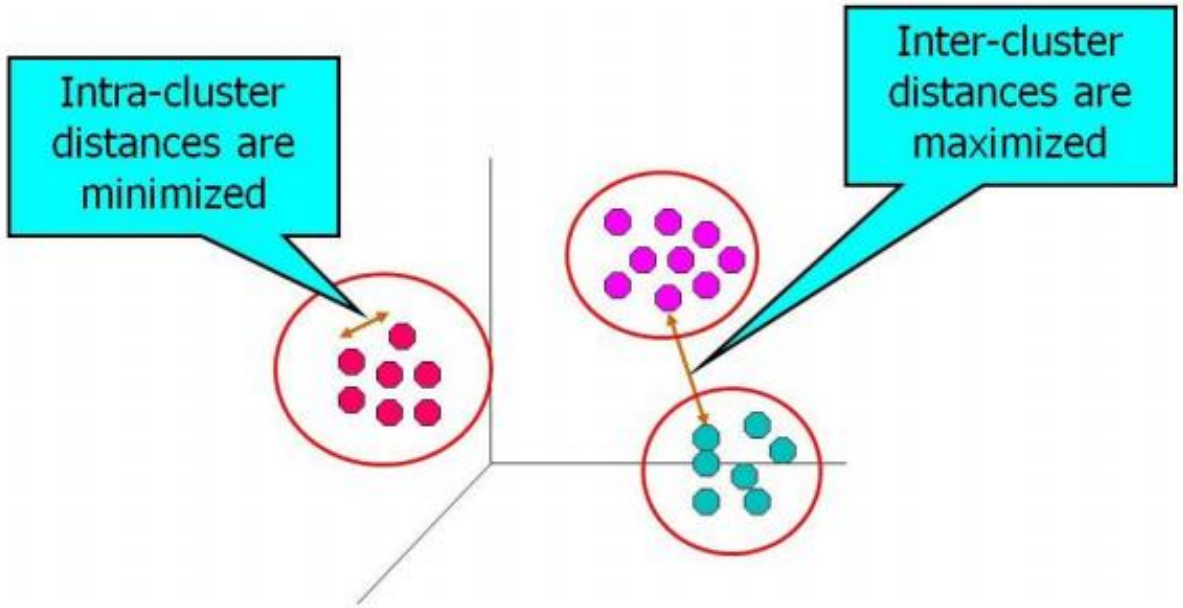
- Give only data > 90%
- Assign the object to the class with the highest probability
- Assign the object to the class only if its probability >40%



For example, in a banking application, the customer who applies for a loan may be classified as a safe and risky according to his/her age and salary.

Clustering

- Clustering is a technique of organising a group of data into classes and clusters where the objects reside inside a cluster will have high similarity and the objects of two clusters would be dissimilar to each other.
- In clustering, the similarity as well as distance between two objects is measured by the similarity function. Generally, Intra-cluster distance are less than inter-cluster distance.

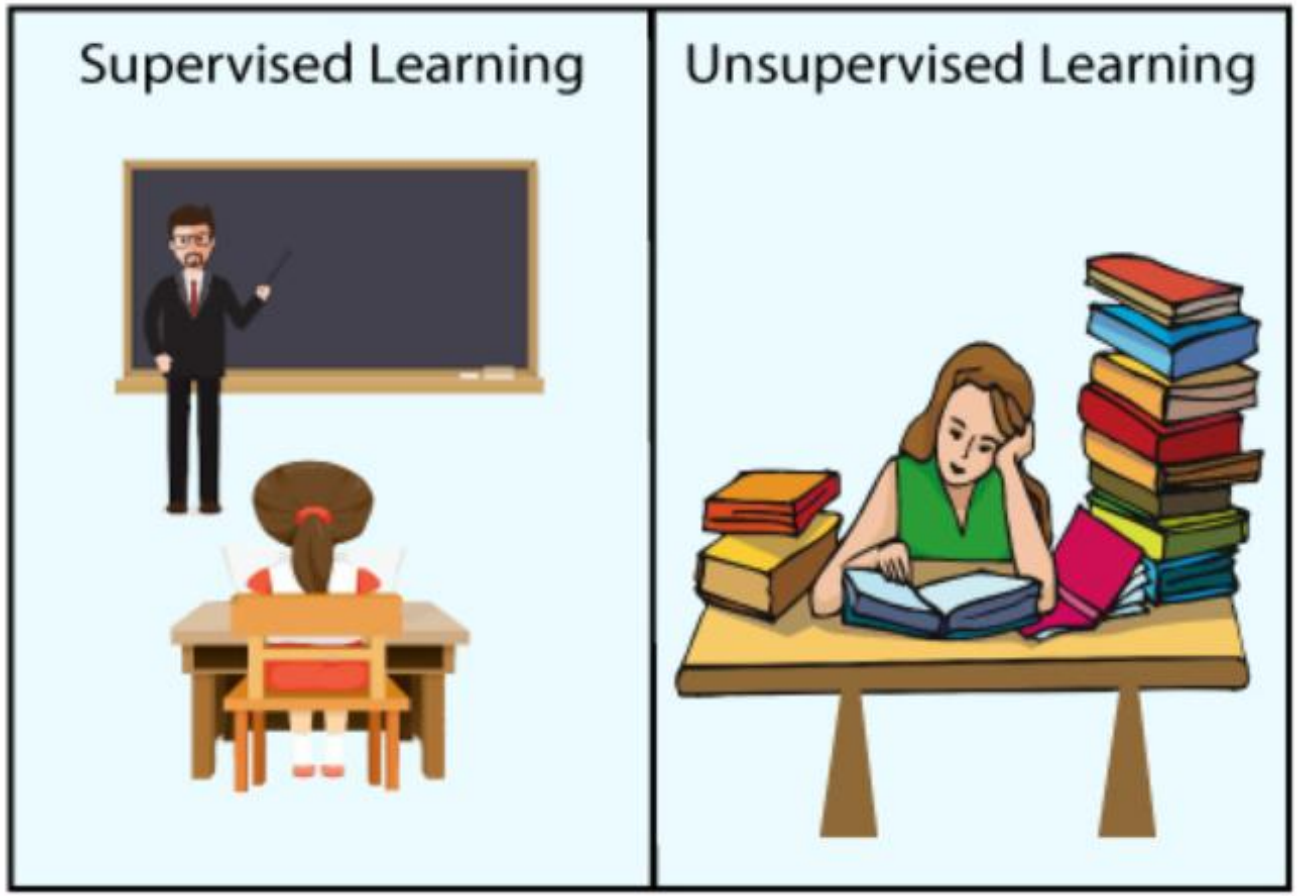


Classification Vs Clustering

Parameter	Classification	Clustering
Type	Used for supervised learning	Used for unsupervised learning
Basic	process of classifying the input instances based on their corresponding class labels	grouping the instances based on their similarity without the help of class labels
Need	it has labels, so there is need of training and testing dataset for verifying the created model	there is no need of training and testing dataset
Complexity	more complex	less complex
Example Algorithms	Logistic regression, Naive Bayes classifier, Support vector machines, KNN, Decision Tree etc.	k-means clustering algorithm, Fuzzy c-means clustering algorithm, Gaussian (EM) clustering algorithm, etc.
Application	Detection of unsolicited email Recognition of the face Approval of a Bank Loan	Investigation of the social networks Segmentation of an image Recommendation Engines

Supervised vs. Unsupervised Learning

Supervised and Unsupervised learning are the two techniques of ML. Both are used in different scenarios and with different dataset.



Supervised vs. Unsupervised Learning

Supervised learning (classification) –

- In Supervised learning models are trained using labeled data.
- In supervised learning, models need to **find the mapping function** to map the input variable (X) with the output variable (Y).

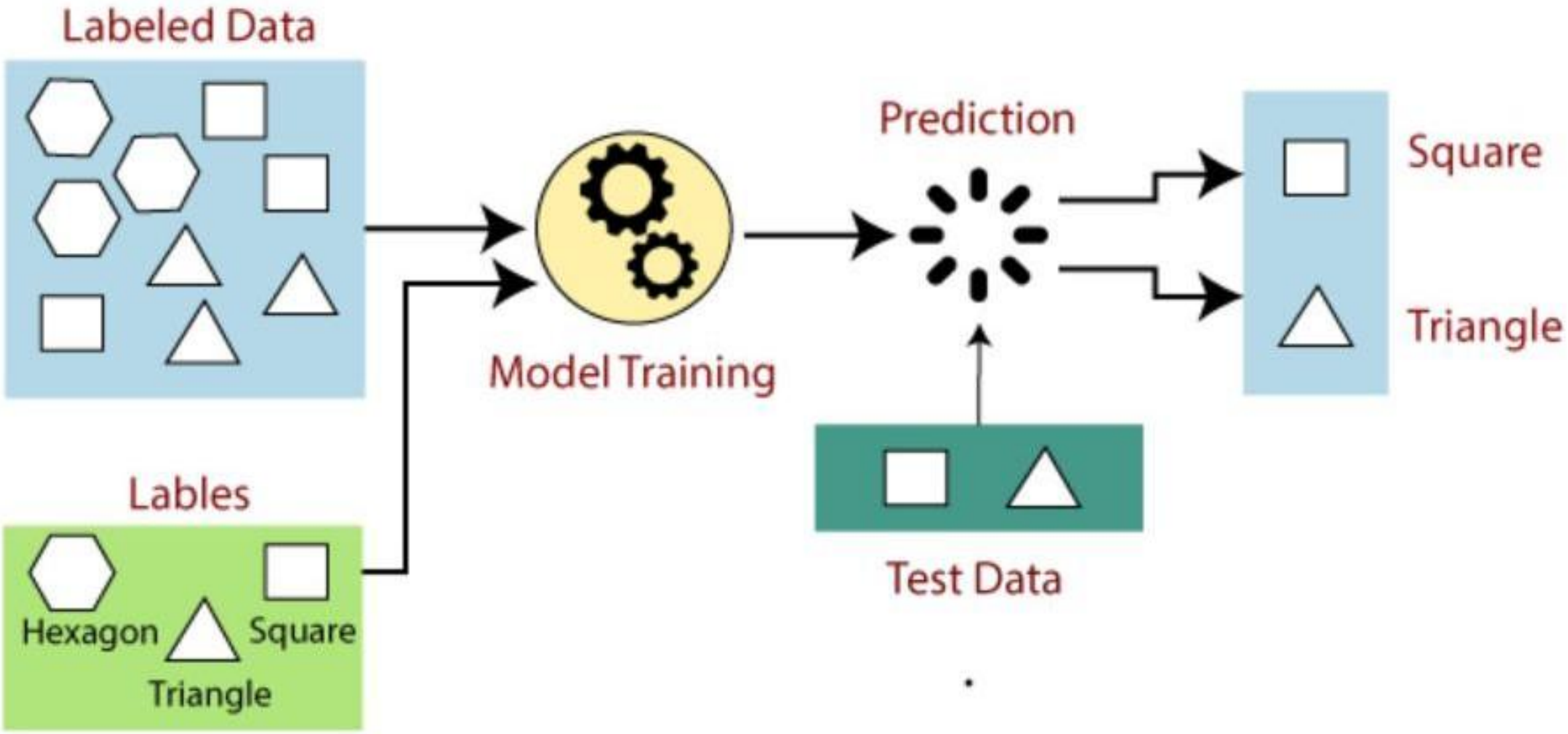
  $$Y = f(X)$$

- Supervised learning **needs supervision to train the model**, which is similar to as a student learns things in the presence of a teacher.
- Supervised learning can be used for two types of problems: Classification and Regression.

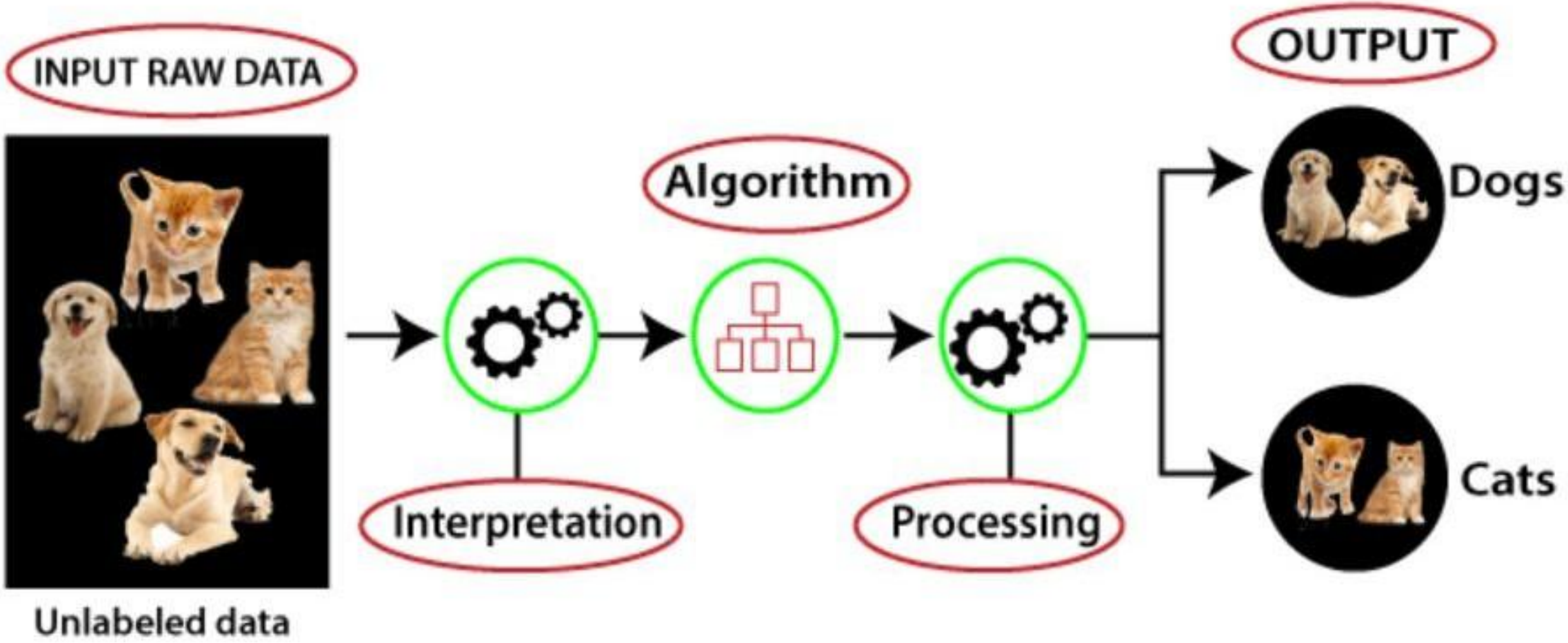
Unsupervised learning (clustering) –

- In Unsupervised learning patterns are inferred from the unlabeled input data.
- The goal of unsupervised learning is to find the structure and patterns from the input data.
- Unsupervised learning does not need any supervision. Instead, it finds patterns from the data by its own
- Unupervised learning can be used for two types of problems: Clustering and Association.

How Supervised Learning Works?



How Un-supervised Learning Works?



Trainning Dataset Vs. Test Dataset

- The **training data** is the biggest subset (>=60%) of the original dataset, which is used to train or fit the model. Firstly, the training data is fed to the algorithms, which lets them learn how to make predictions for the given task. The type of training data that we provide to the model is highly responsible for the model's accuracy and prediction ability. It means that the better the quality of the training data, the better will be the performance of the model.

- The **test dataset** is another well-organized subset (20%-40%) of original data, which is independent of the training dataset. However, it has some similar types of features and class probability distribution for each type of scenario and uses it as a benchmark for model evaluation once the model training is completed.

Classification and Prediction

What is classification and What is prediction?

Issues regarding classification and prediction

What is classification and What is prediction?

Classification:

Classification is to identify the category or the class label of a new observation.

First a set of training data (input data with their class label) is given to the algorithm, the algorithm derives a model or classifier (that can be decision tree, mathematical formula, or a neural network). When a new unlabeled data is given to the classifier model, it should find the class to which it belongs.

Prediction:

Prediction is to compute/predict the numeric output value of a new observation.

Same as classification, a set of training data (input data with their numeric output value) is given to the algorithm, the algorithm derives a model or predictor (that can be regression, K-nearest neighbor). When a new unlabeled data is given to the predictor model, it should find the contineous numeric output.

We can think of prediction as predicting the correct treatment for a particular disease for an individual person whereas the grouping of patients based on their medical records can be considered classification.

Classification Learning: Definition

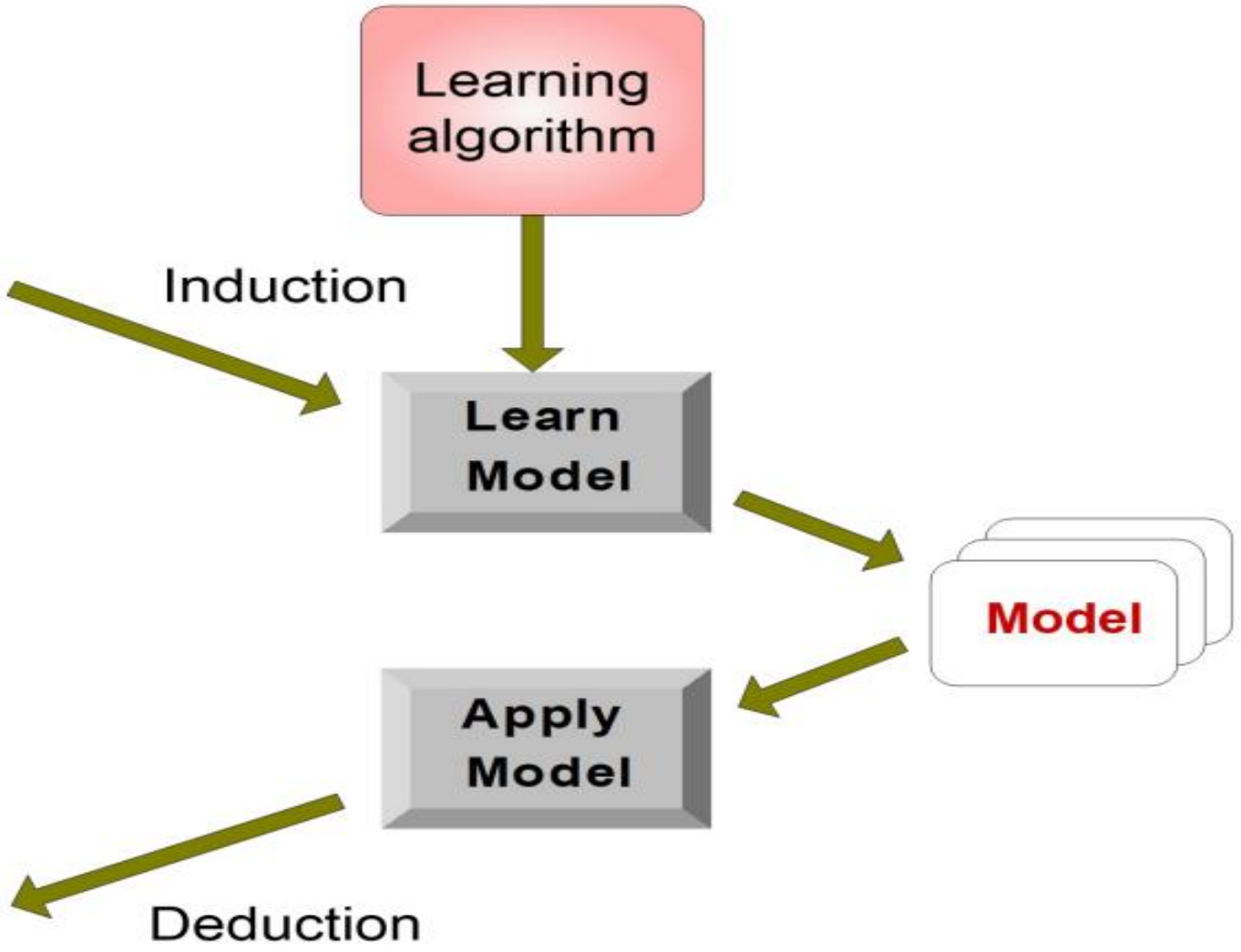
- Given a collection of records (training set)
  - –Each record contains a set of attributes, one of the attributes is the class
- Find a model for the class attribute as a function of the values of the other attributes
- Use ttraining set to construct the model and use test set to estimate the accuracy of the model
- Goal: previously unseen records should be assigned a class as accurately as possible

Illustrating Classification Learning

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?



Classification - A Two-Step Process

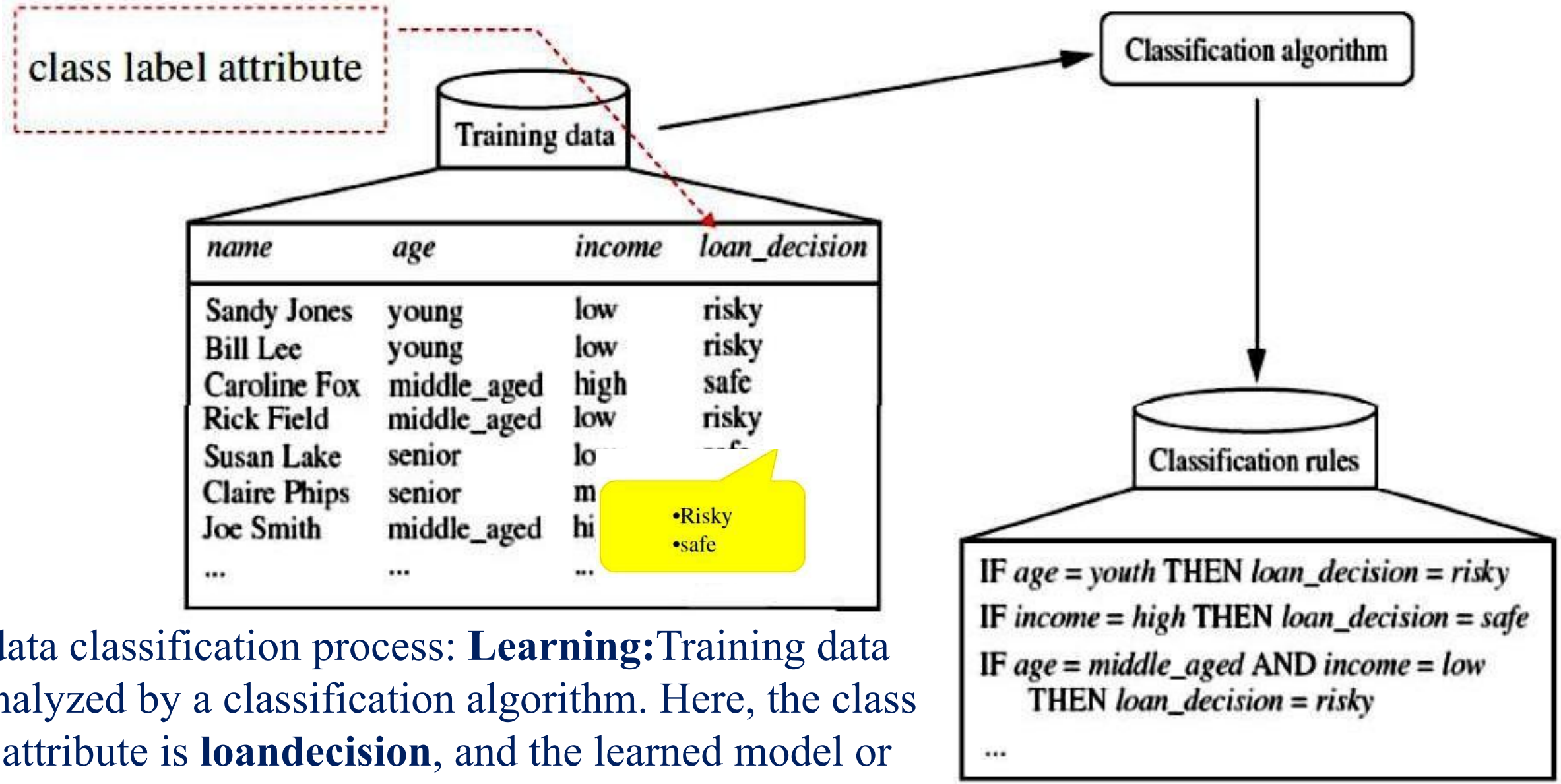
Model construction: describing a set of predetermined classes

- Building the Classifier or Model
- Each tuple/sample is assumed to belong to a predefined class, as defined by the class label attribute
- The set of tuples used for model construction: training set
- The model is represented as classification rules / decision trees / mathematical formulae

Model usage: for classifying future or unknown objects

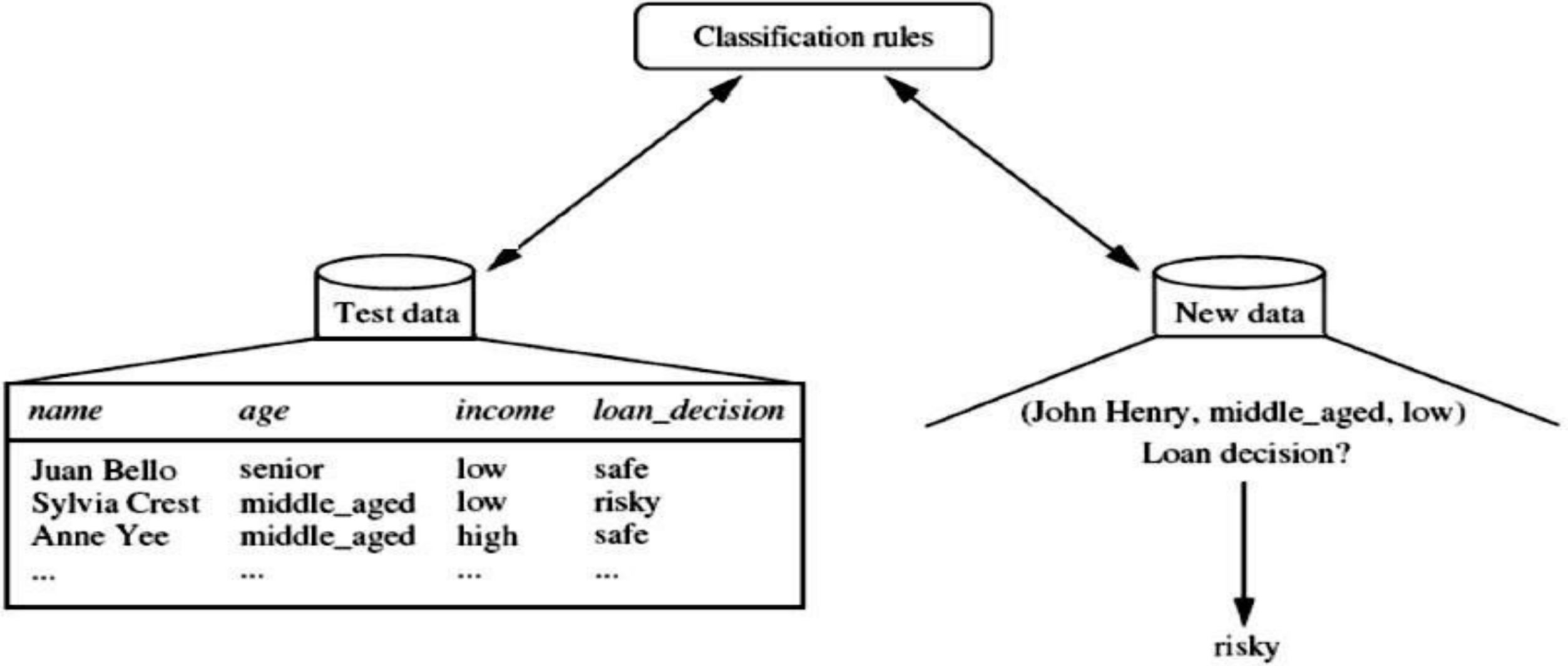
- Using Classifier for Classification estimate accuracy of the model
- The known label of test sample is compared with the classified result from the model
- Accuracy rate is the percentage of test set that are correctly classified by the model
- Test set is independent of training set, otherwise over-fitting will occur

Classification - A Two-Step Process: Model Construction



The data classification process: **Learning:**Training data are analyzed by a classification algorithm. Here, the class label attribute is **loandecision**, and the learned model or classifier is represented in the form of classification rules.

Classification - A Two-Step Process: Model Usage



Classifiaction: Test data are used to estimate the accuracy of the classification rules. If the accuracy is considered acceptable, the rules can be applied to the classification of new data tuples

Issues regarding classification and prediction: (1) Data Preparation

- **Data cleaning**
  - Preprocess data in order to reduce noise and handle missing values
- **Relevance analysis (feature selection)**
  - Remove the irrelevant or redundant attributes
- **Data transformation**
  - Generalize and/or normalize data

Issues regarding classification and prediction: (2) Evaluating Classification Methods

- Predictive accuracy
- Speed and scalability
  - ✓time to construct the model
  - ✓time to use the model
- Robustness
  - ✓handling noise and missing values
- Scalability
  - ✓efficiency in disk-resident databases
- Interpretability
  - ✓understanding and insight provded by the model
- Goodness of rules
  - ✓decision tree size
  - ✓compactness of classification rules

Classification Technique

- Decision Trees
- Bayesian Classifiers
- Neural Networks
- K-Nearest Neighbour
- Support Vector Machines
- Linear Regression
- Logistic Regression
- Fuzzzy Algorithm
- Genetic Algorithm

Classification by Decision Tree Induction

- Decision tree
  - A flow-chart-like tree structure
  - **Internal node** denotes a test on an attribute
  - **Branch** represents an outcome of the test
  - **Leaf** nodes represent class labels or class distribution
  - The **topmost node** in the tree is the root node.
- Decision tree generation consists of two phases

–**Tree construction**

- ➢ At start, all the training examples are at the root
- ➢ Partition examples recursively based on selected attributes

–**Tree pruning**

- ➢Identify and remove branches that reflect noise or outliers

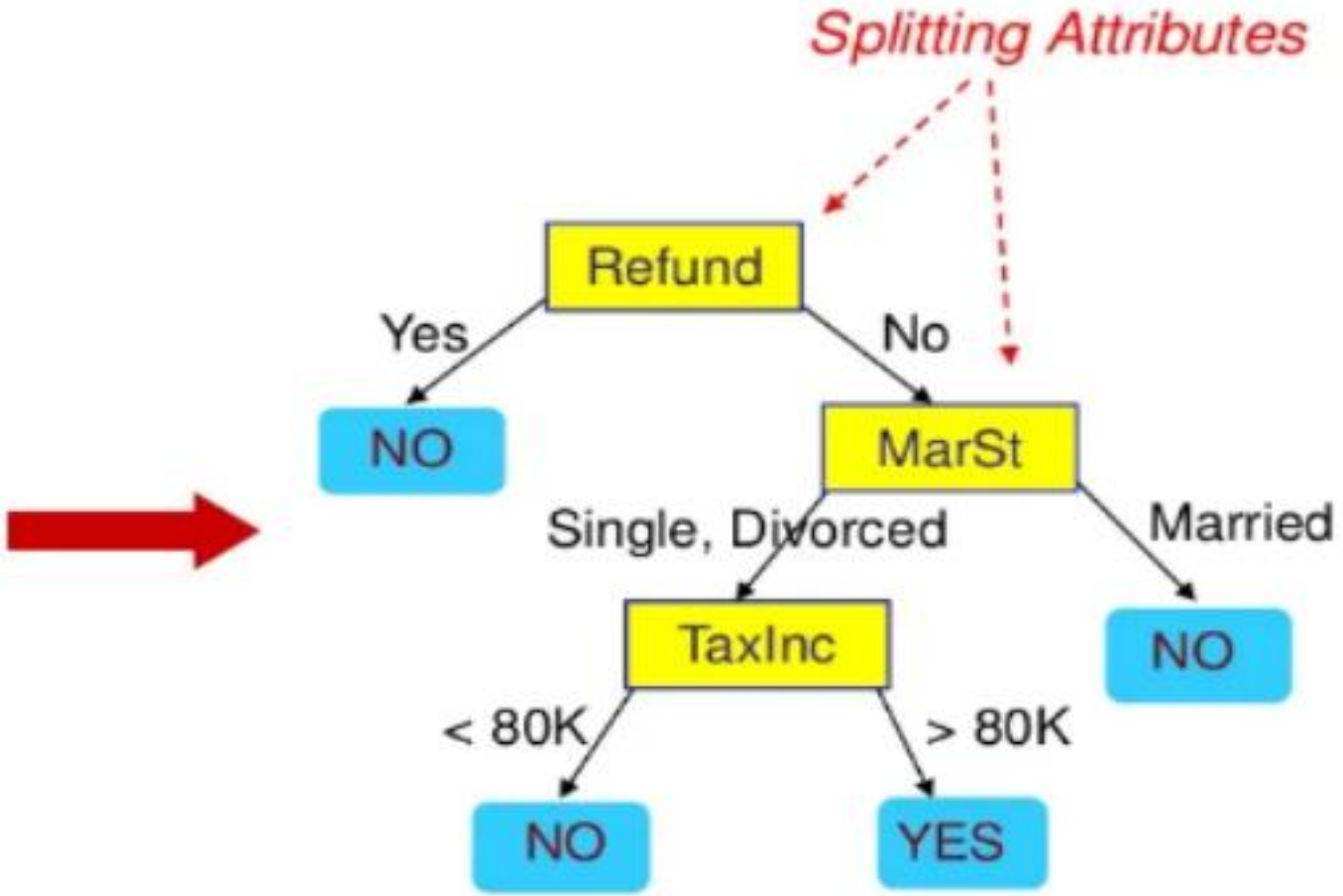
– Use of decision tree: Classifying an unknown sample

– Test the attribute values of the sample against the decision tree

Decision Tree for Cheat or not Cheat

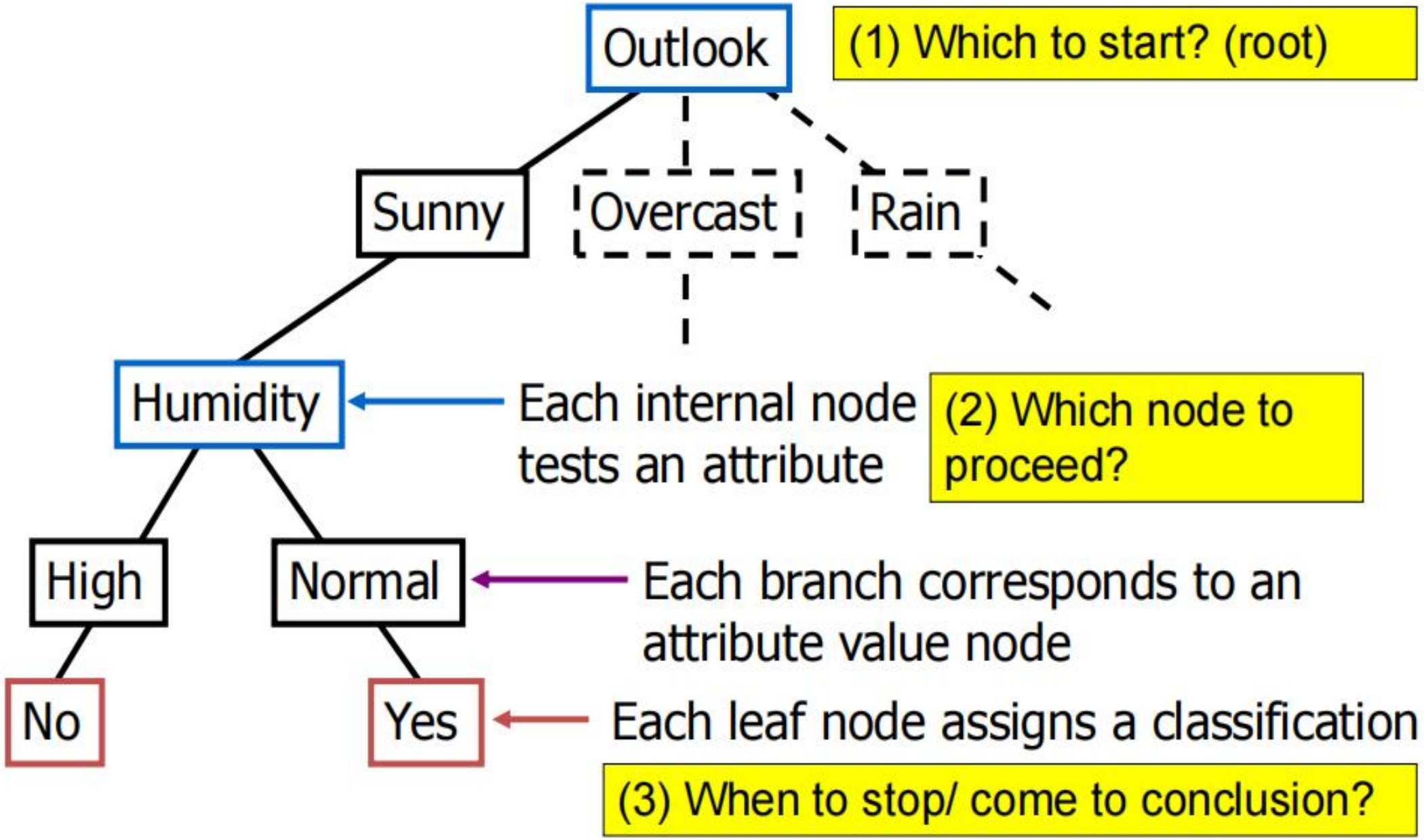
	categorical	categorical	continuous	class
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single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

Training Data



Model: Decision Tree

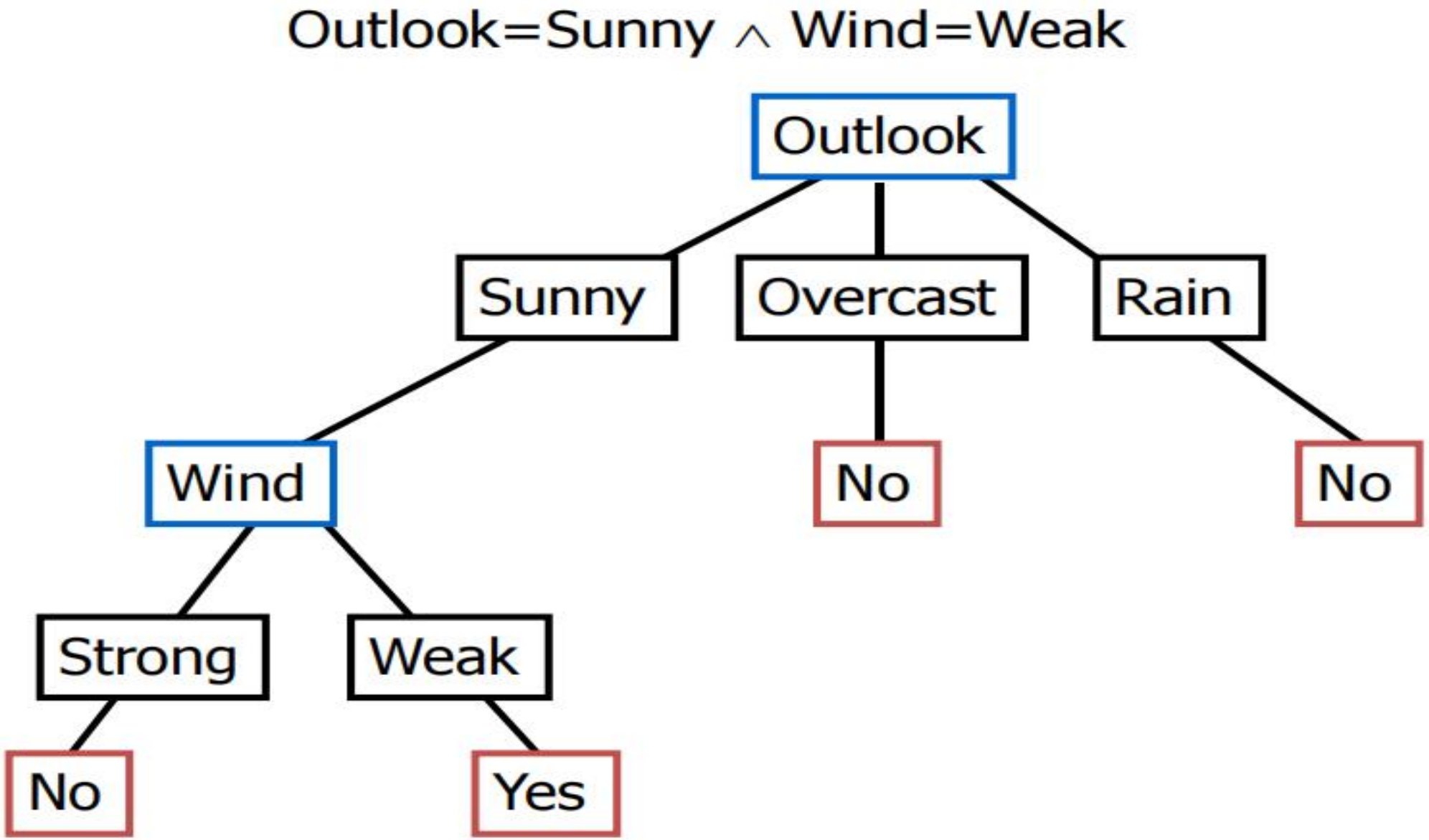
Decision Tree for PlayTennis



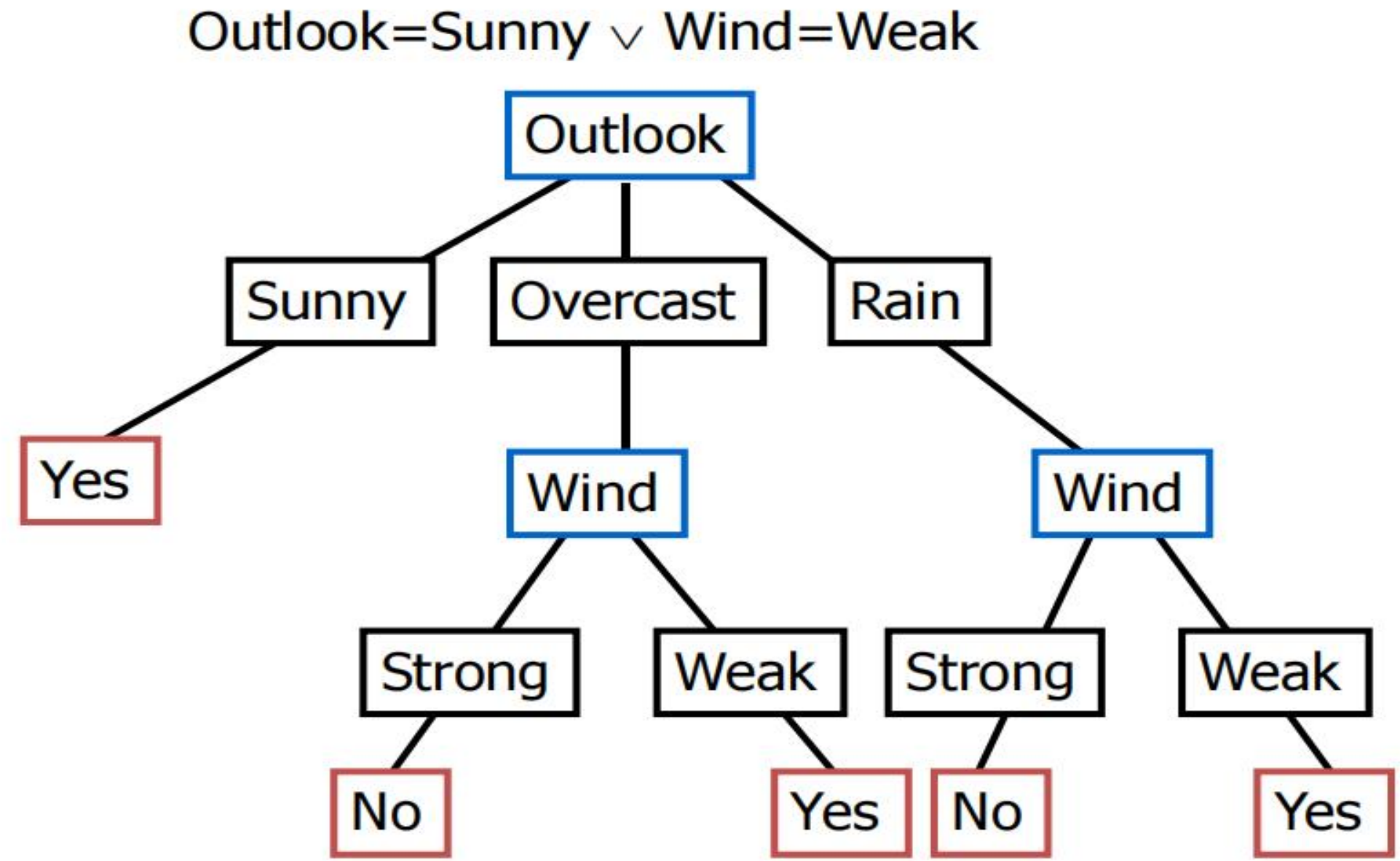
Play-Tennis Data

DAY	OUTLOOK	TEMP	HUMIDITY	WIND	PLAY
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

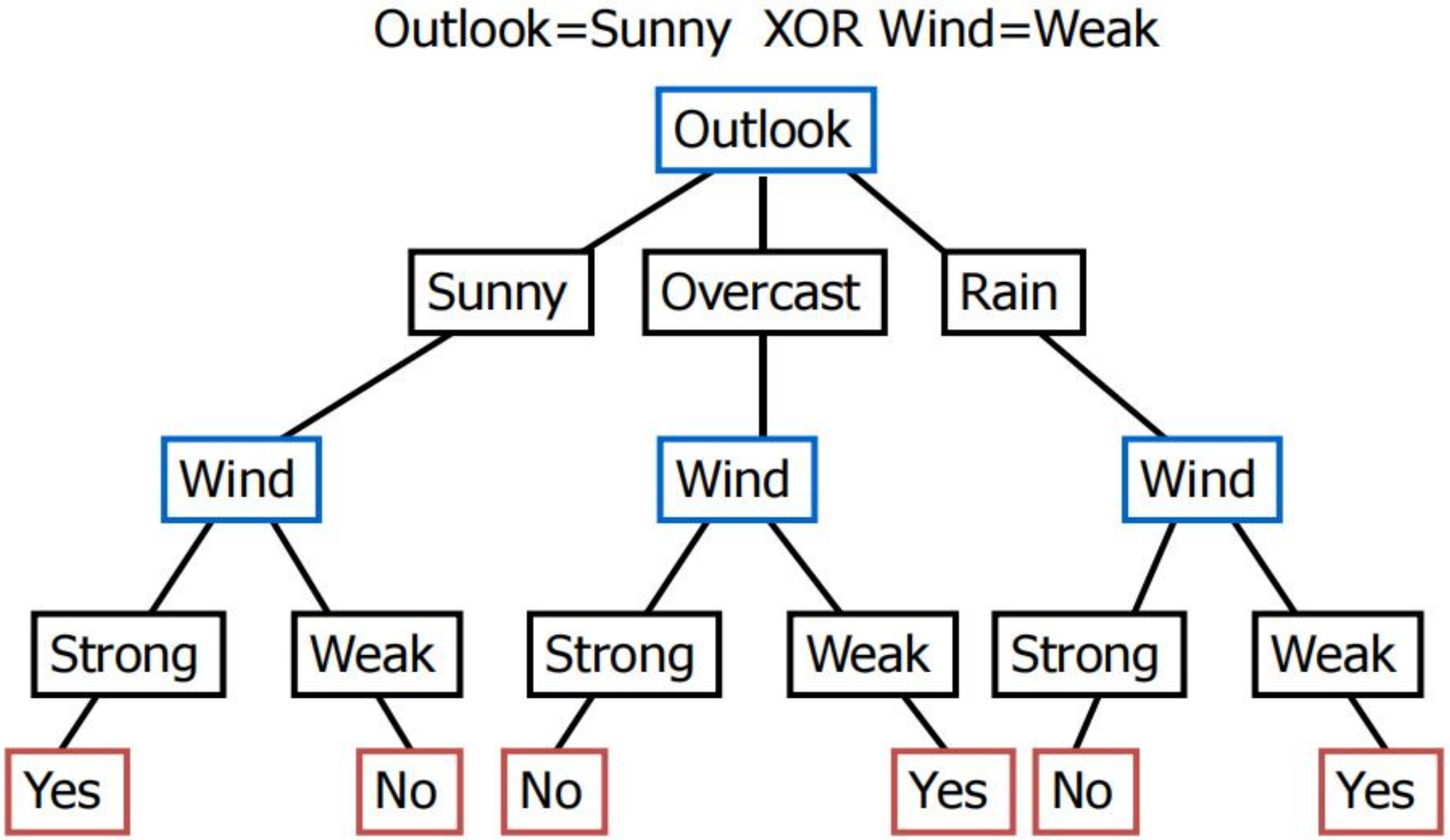
Decision Tree for Conjunction



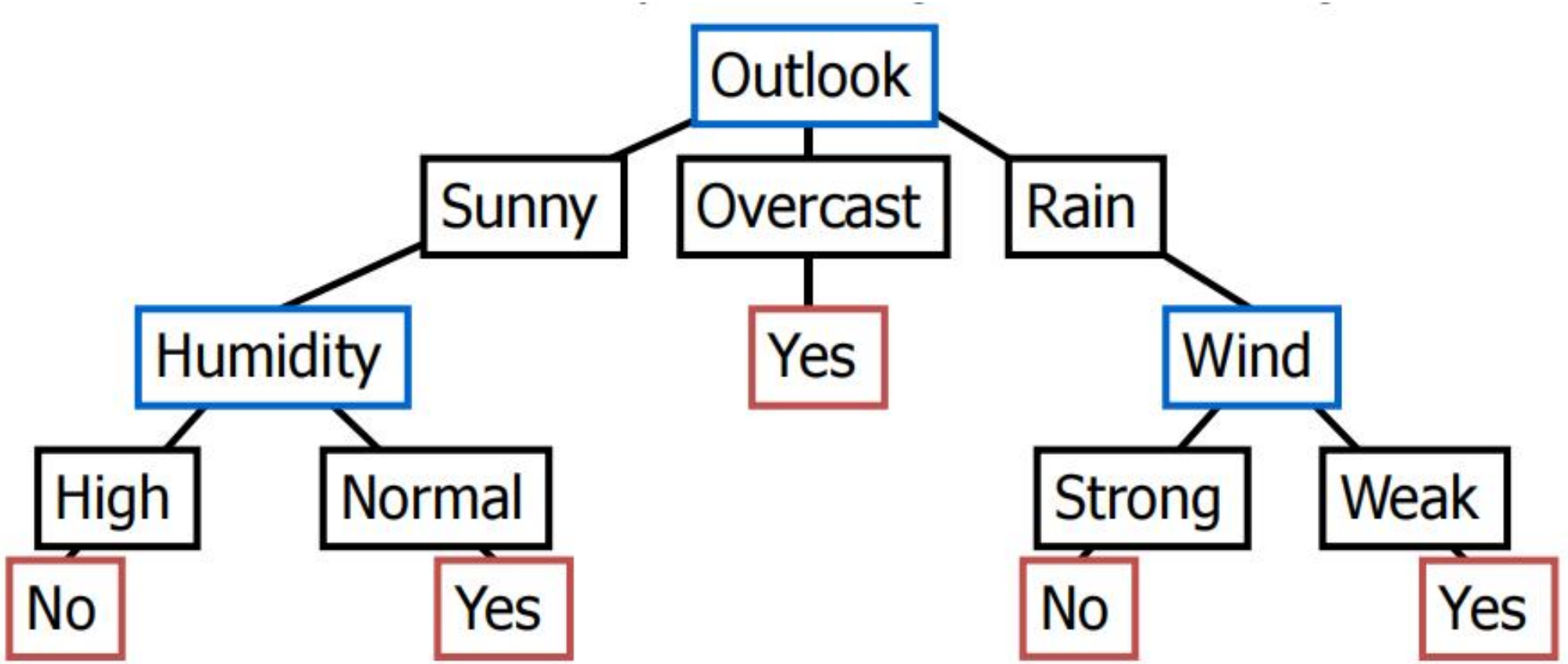
Decision Tree for Disjunction



Decision Tree for XOR



Decision Tree for Disjunction of Conjuction

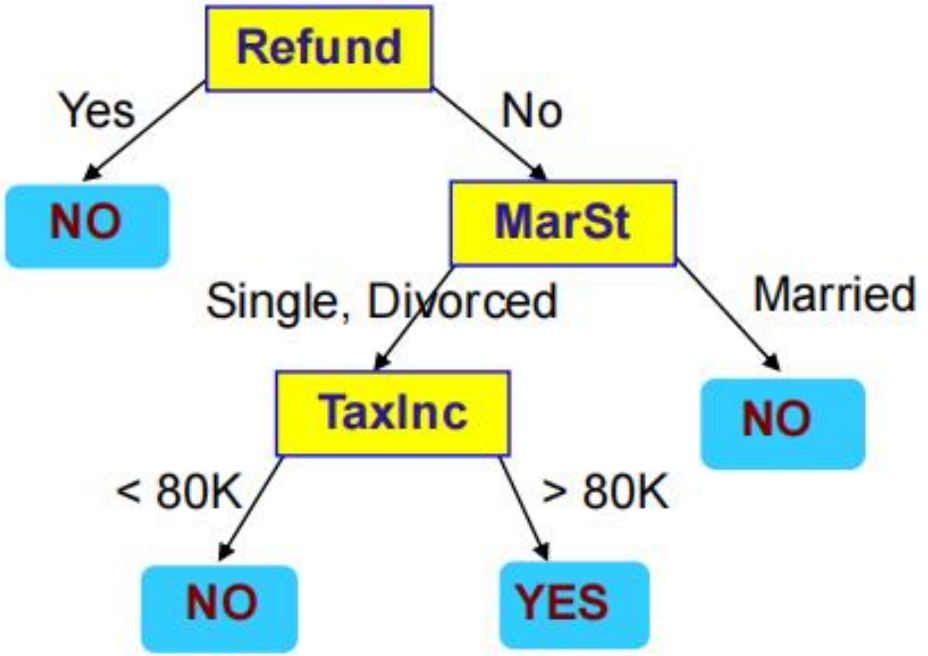


(Outlook=Sunny $\wedge$ Humidity=Normal)
$\vee$ (Outlook=Overcast)
$\vee$ (Outlook=Rain $\wedge$ Wind=Weak)

Example of a Decision Tree

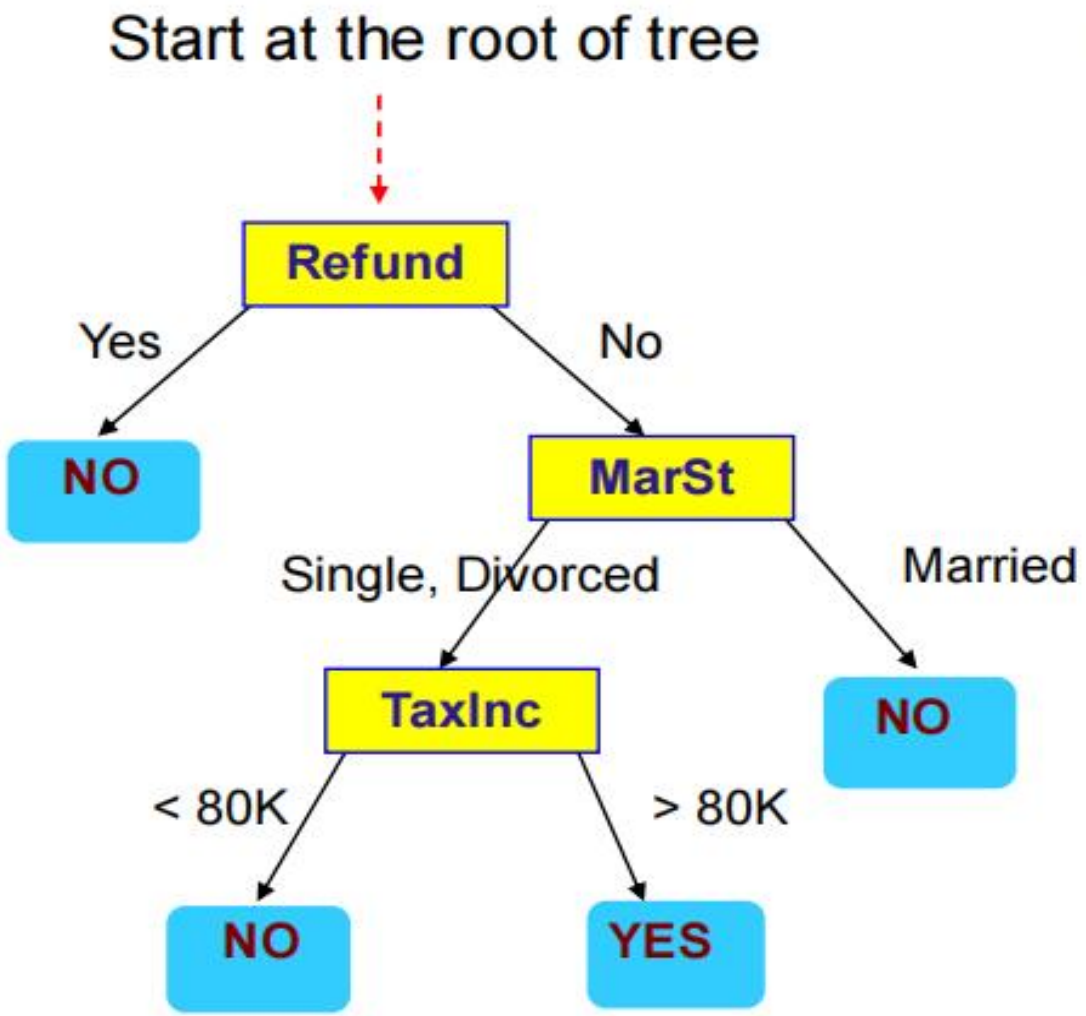
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data



Model: Decision Tree

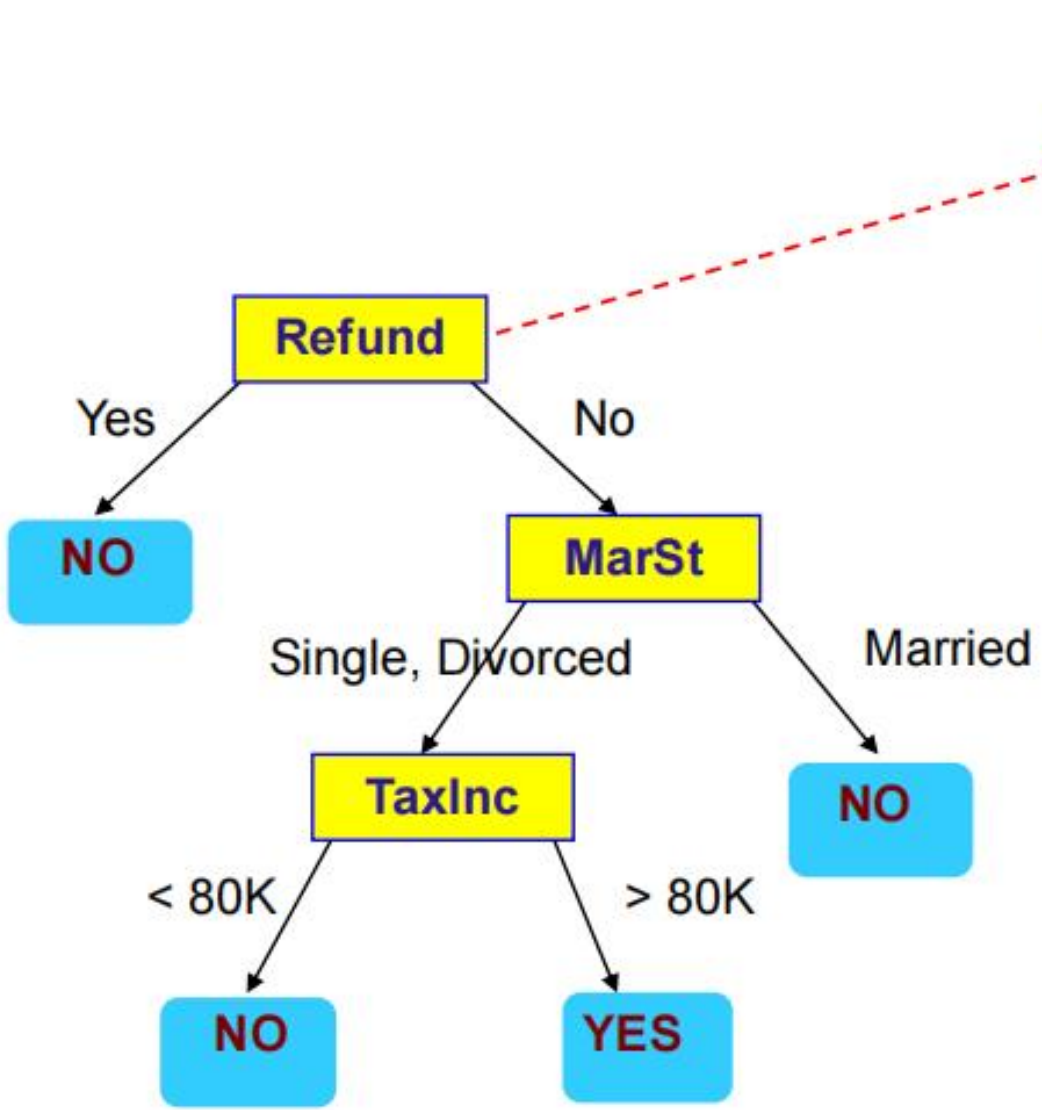
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

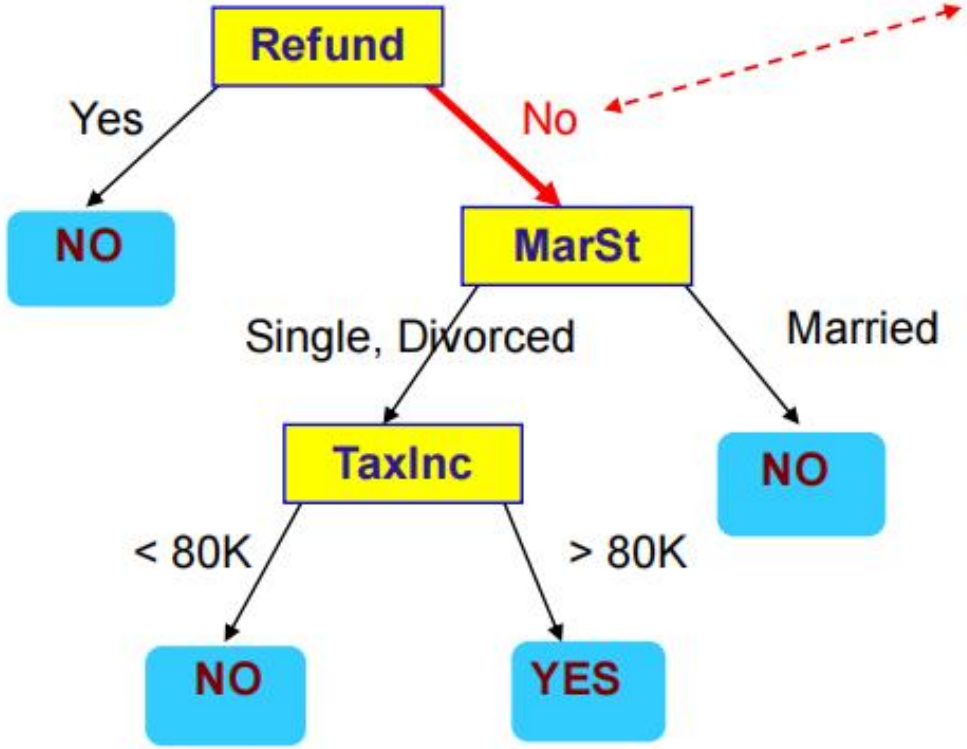
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

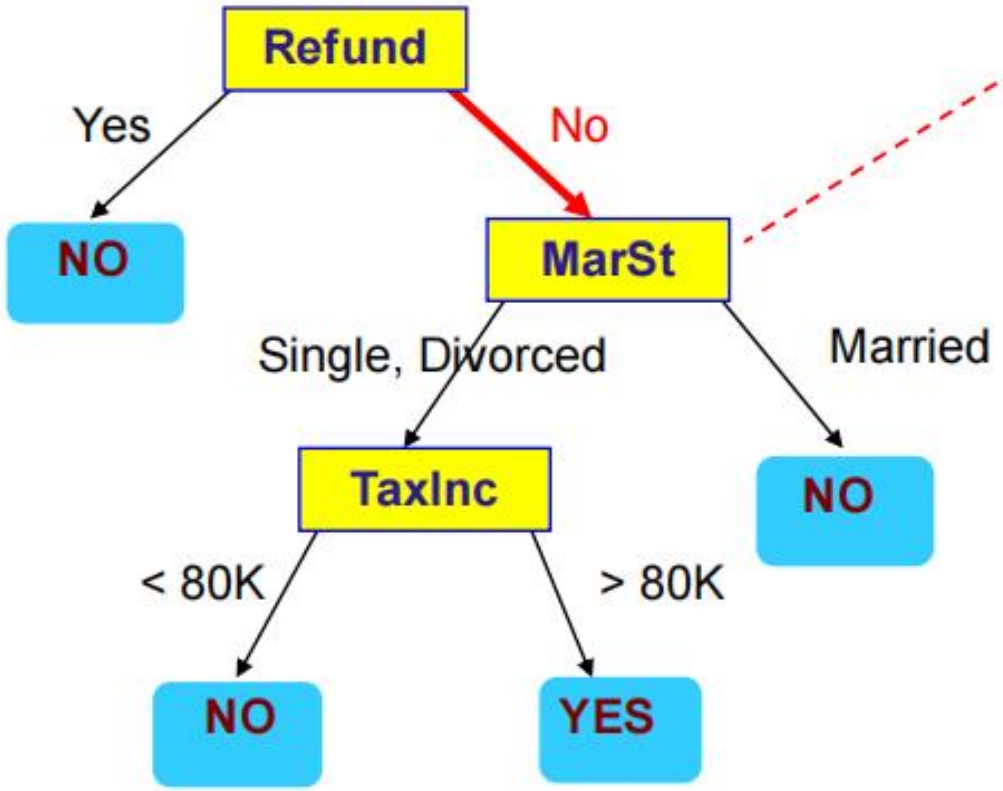
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

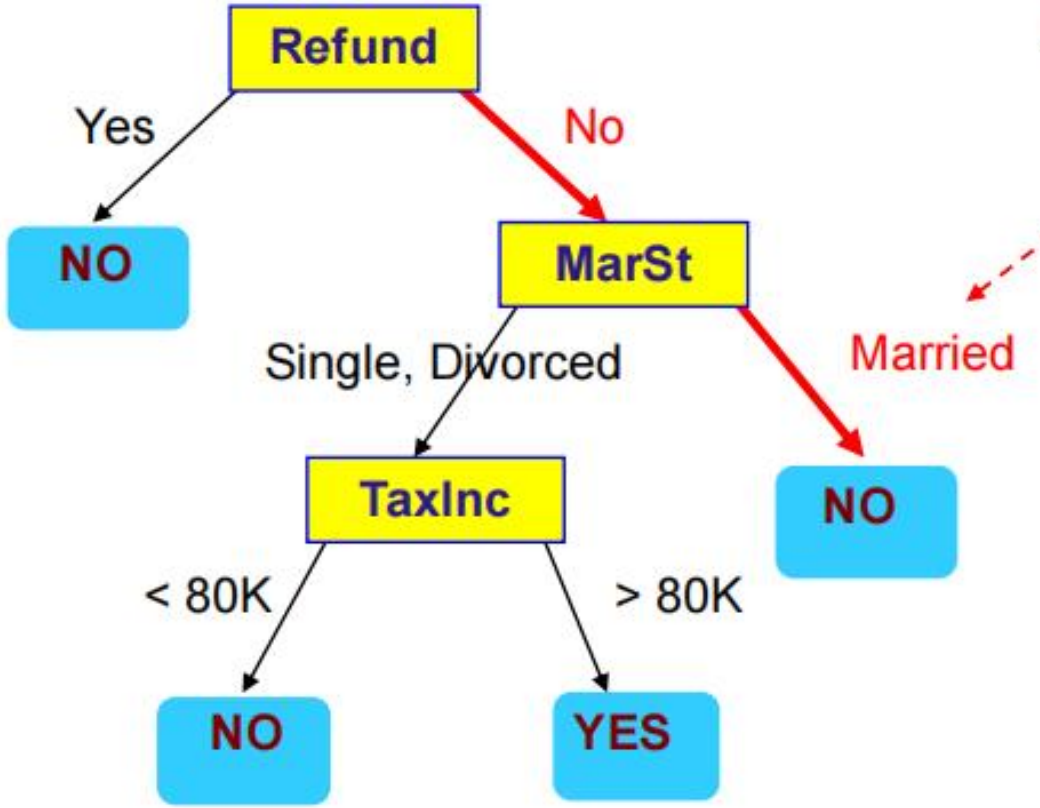
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

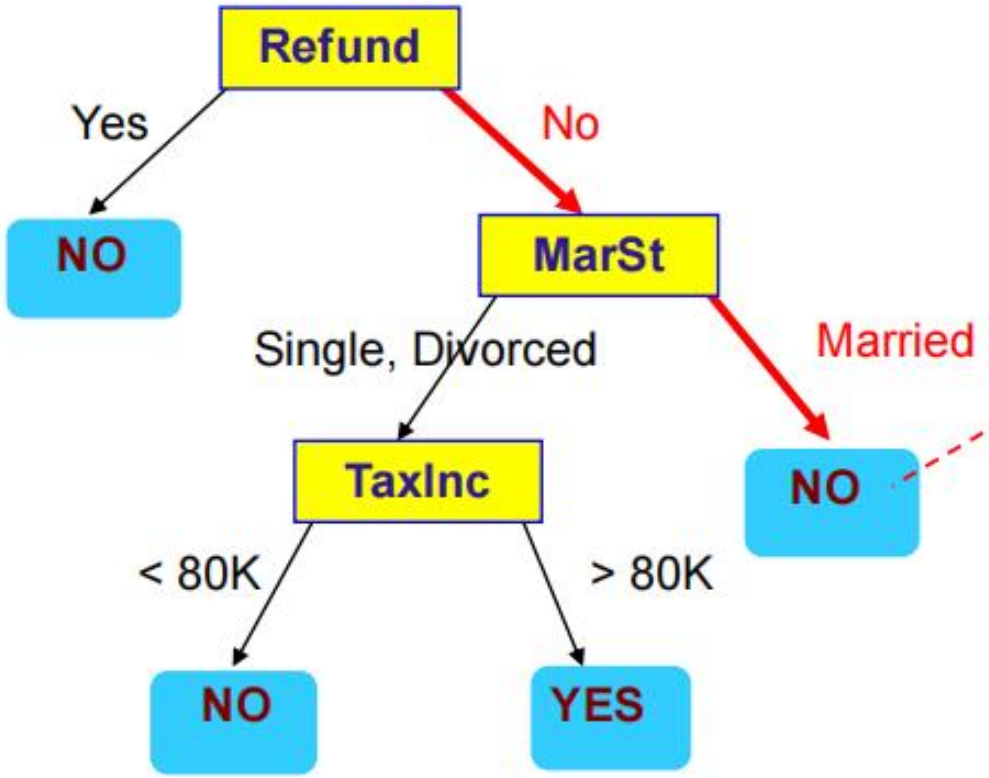
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Assign Cheat to “No”

Decision Tree Algorithm

Principle

- Basic algorithm (adopted by ID3, C4.5 and CART): a **greedy algorithm**
- Tree is constructed in a top-down recursive divide-and-conquer manner
- Attributes are categorical (if continuous-valued, they are discretized in advance)
- Choose the *best* attribute(s) to split the remaining instances and make that attribute a decision node

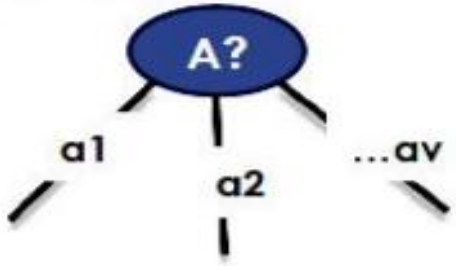
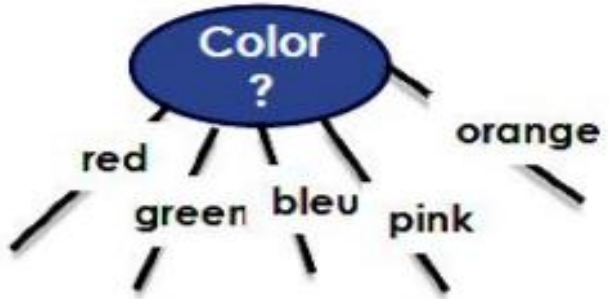
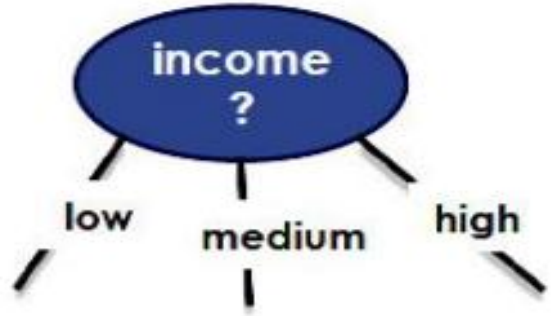
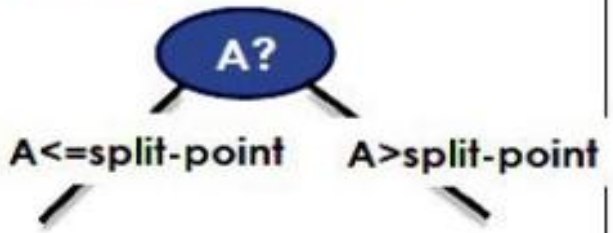
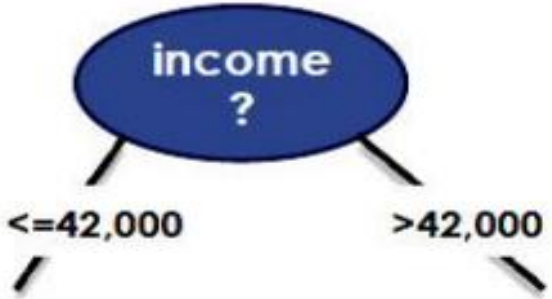
Iterations

- At start, all the training tuples are at the root
- Tuples are partitioned recursively based on selected attributes
- Test attributes are selected on the basis of a heuristic or statistical measure (e.g., **information gain**)

Stopping conditions

- All samples for a given node belong to the same class
- There are no remaining attributes for further partitioning – **majority voting** is employed for classifying the leaf
- There are no samples left

Three Possible Partition Scheme

Partitioning scenarios	Examples
Discrete-valued 	 
Continuous-valued 	
Discrete-valued+ binary tree 	

Attribute Selection Measures

How to choose An Attribute?

- An attribute selection measure is a heuristic for selecting the splitting criterion that "best" separates a given data partition, D, of class labeled training tuples into individual classes.

Ideally

  - Each resulting partition would be pure
  - A pure partition is a partition containing tuples that all belong to the same class

- **Attribute selection measures (splitting rules)**
  - Determine how the tuples at a given node are to be split
  - Provide ranking for each attribute describing the tuples
  - The attribute with highest score is chosen
  - Determine a split point or a splitting subset

- **Methods**
  - Information gain (ID3 (Iterative Dichotomiser 3) /C4.5)
  - Gain ratio
  - Gini Index (IBM IntelligentMiner)

Atribute Selection Measure

- Attribute selection measure finds the best attributes in sequence which can give a smallest decision tree. It provide a splitting rule that determine how the tuple at a give node can be best split.
- On the basis of sample dataset/traning dataset attribute selection measure provide a ranking for each attibute.
- The attribute with best rank/score for the measure is choosen as the splitting attribute for the given tuples.
- Three popular attribute selection measure:
  - Information Gain
  - Gain Ratio
  - Gini Index

Information Gain Method

- **Entropy/Information** is the probabilistic measure of uncertanity/variance/randomness present in the dataset/attribute.
- **Information Gain** is the decrease/increase in Entropy value when the node is split. It is the measure of a reduction of uncertainty/variance.
- Based on the computed values of Entropy and Information Gain teh best attribute at the respective level is choosen

Information Gain Method

To find the best feature that serves as a root node in terms of information gain:

1. Find the Expected Entropy/Information of Dataset: the average amount of information ned to identify the class label of a tuple in D.

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

- **m: number of Classes or labels**
- **p_i: Probability of Class i**

2. Find the Etropy/Information of individual feature/attribute

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

- **v: no. of variations of attribute A**
- **p_i: Probability of Class i**

3. Find the information gain of individual attribute(s).

- **A is the particular Attribute**

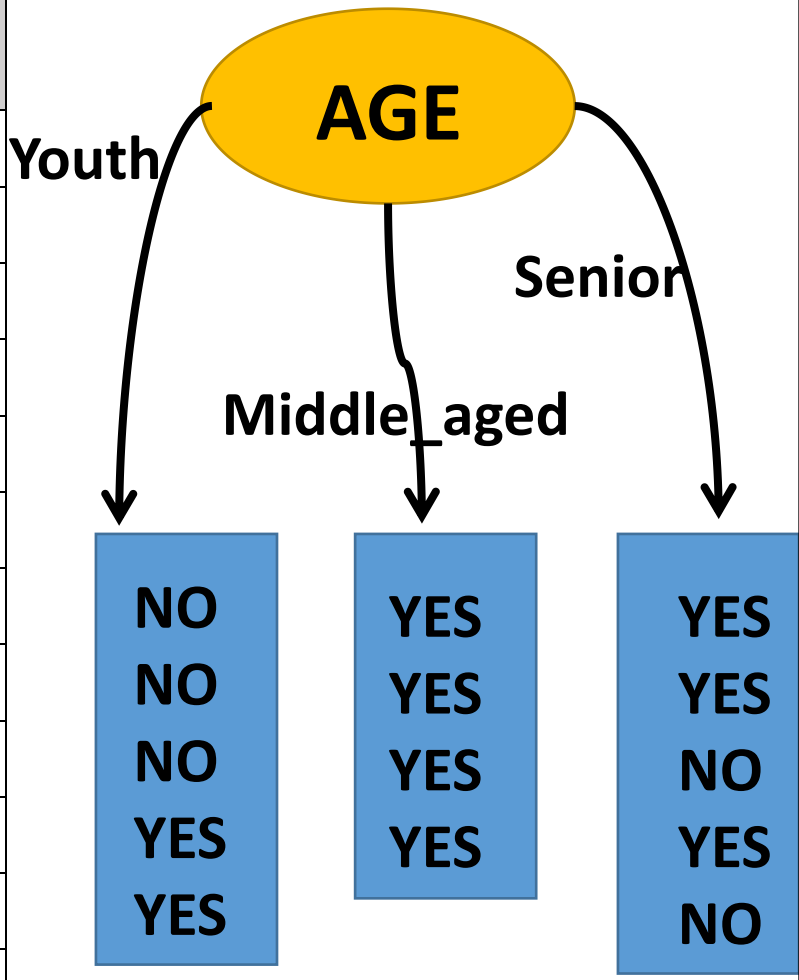
$$Gain(A) = Info(D) - Info_A(D)$$

3. Choose the best Information gain i.e Information gain with highest value.

Repeat this process to constuct the complete decision tree.

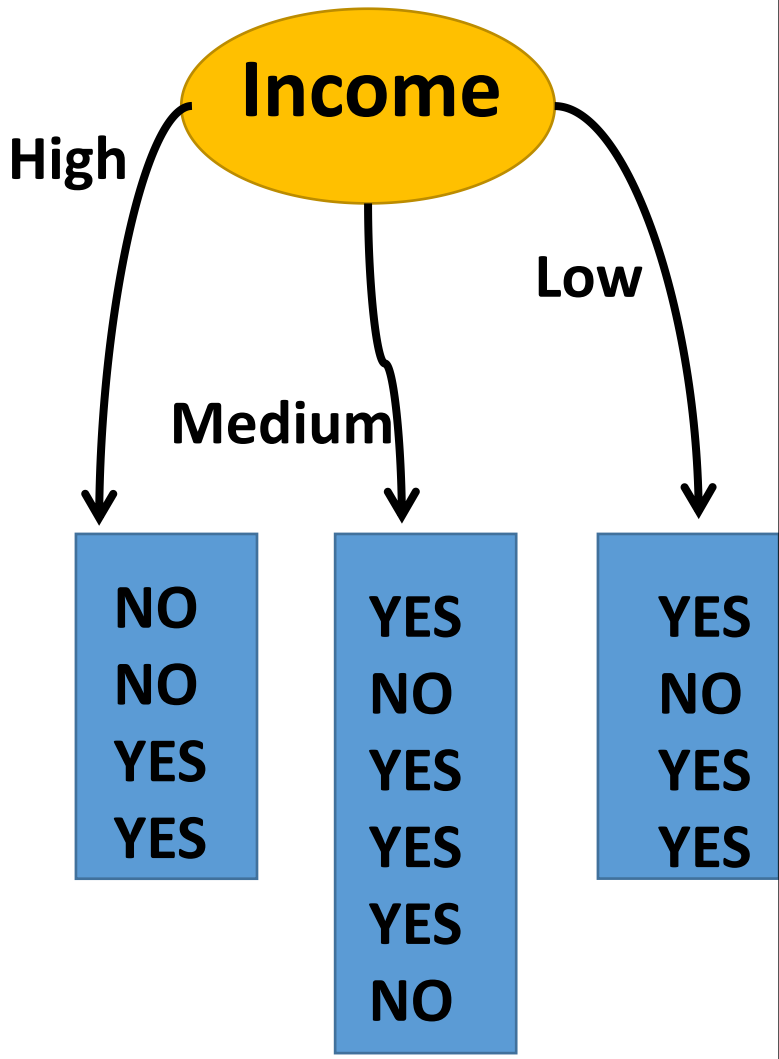
Which Attribute to Select

RID	Age	Income	Student	Credit_Rating	Class: Buys Computer
1	Youth	High	No	Fair	No
2	Youth	High	No	Excellent	No
3	middle_aged	High	No	Fair	Yes
4	senior	Medium	No	Fair	Yes
5	senior	Low	Yes	Fair	Yes
6	senior	Low	Yes	Excellent	No
7	middle_aged	Low	Yes	Excellent	Yes
8	Youth	Medium	No	Fair	No
9	Youth	Low	Yes	Fair	Yes
10	senior	Medium	Yes	Fair	Yes
11	Youth	Medium	Yes	Excellent	Yes
12	middle_aged	Medium	No	Excellent	Yes
13	middle_aged	High	Yes	Fair	Yes
14	senior	Medium	No	Excellent	No



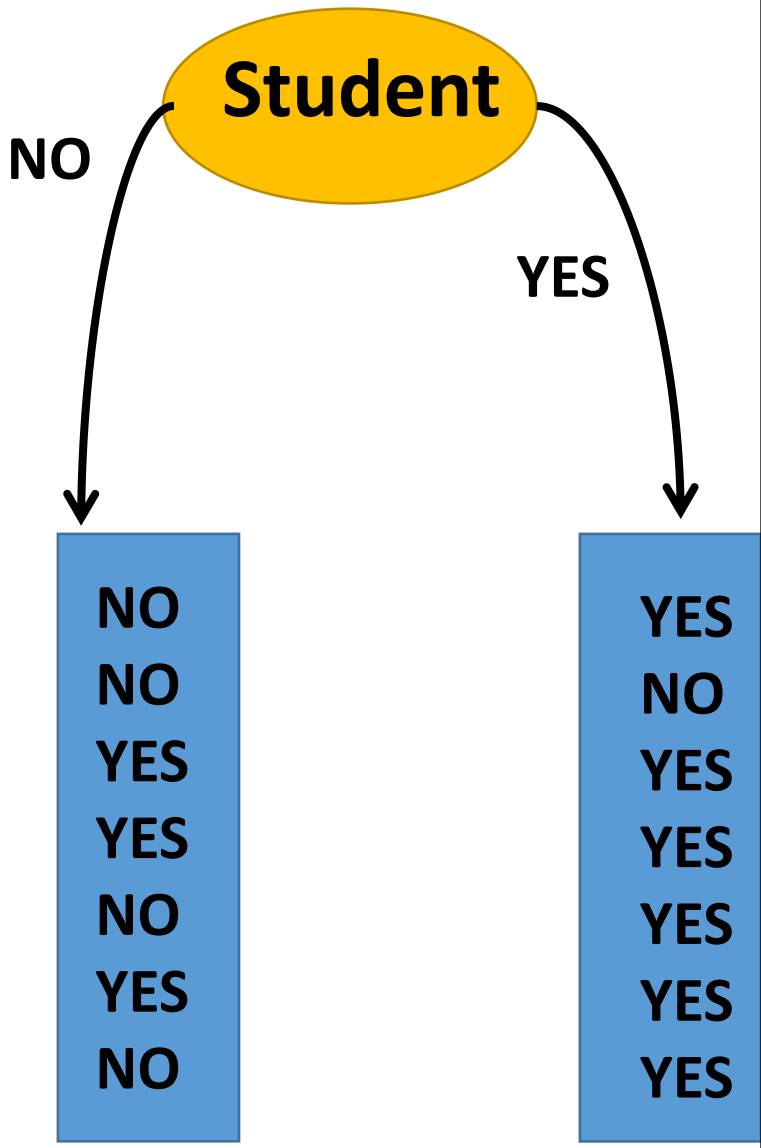
Which Attribute to Select

RID	Age	Income	Student	Credit_Rating	Class: Buys Computer
1	Youth	High	No	Fair	No
2	Youth	High	No	Excellent	No
3	middle_aged	High	No	Fair	Yes
4	senior	Medium	No	Fair	Yes
5	senior	Low	Yes	Fair	Yes
6	senior	Low	Yes	Excellent	No
7	middle_aged	Low	Yes	Excellent	Yes
8	Youth	Medium	No	Fair	No
9	Youth	Low	Yes	Fair	Yes
10	senior	Medium	Yes	Fair	Yes
11	Youth	Medium	Yes	Excellent	Yes
12	middle_aged	Medium	No	Excellent	Yes
13	middle_aged	High	Yes	Fair	Yes
14	senior	Medium	No	Excellent	No



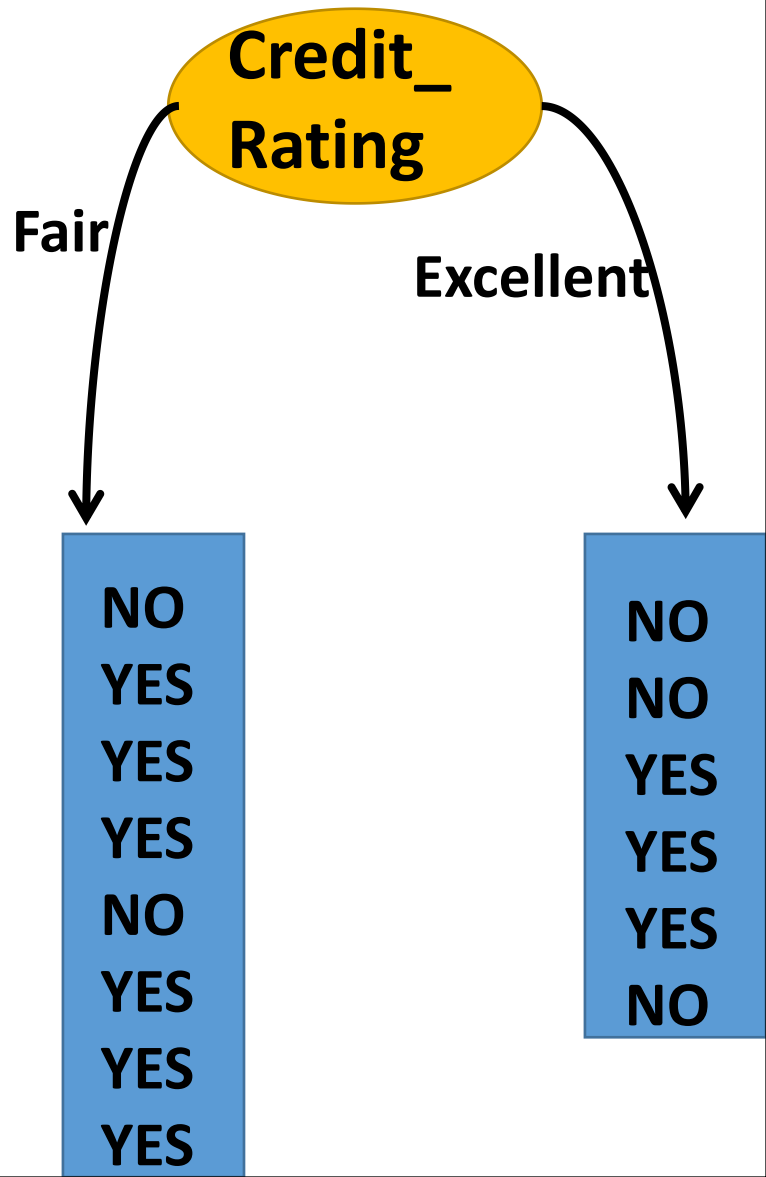
Which Attribute to Select

RID	Age	Income	Student	Credit_Rating	Class: Buys Computer
1	Youth	High	No	Fair	No
2	Youth	High	No	Excellent	No
3	middle_aged	High	No	Fair	Yes
4	senior	Medium	No	Fair	Yes
5	senior	Low	Yes	Fair	Yes
6	senior	Low	Yes	Excellent	No
7	middle_aged	Low	Yes	Excellent	Yes
8	Youth	Medium	No	Fair	No
9	Youth	Low	Yes	Fair	Yes
10	senior	Medium	Yes	Fair	Yes
11	Youth	Medium	Yes	Excellent	Yes
12	middle_aged	Medium	No	Excellent	Yes
13	middle_aged	High	Yes	Fair	Yes
14	senior	Medium	No	Excellent	No



Which Attribute to Select

RID	Age	Income	Student	Credit_Rating	Class: Buys Computer
1	Youth	High	No	Fair	No
2	Youth	High	No	Excellent	No
3	middle_aged	High	No	Fair	Yes
4	senior	Medium	No	Fair	Yes
5	senior	Low	Yes	Fair	Yes
6	senior	Low	Yes	Excellent	No
7	middle_aged	Low	Yes	Excellent	Yes
8	Youth	Medium	No	Fair	No
9	Youth	Low	Yes	Fair	Yes
10	senior	Medium	Yes	Fair	Yes
11	Youth	Medium	Yes	Excellent	Yes
12	middle_aged	Medium	No	Excellent	Yes
13	middle_aged	High	Yes	Fair	Yes
14	senior	Medium	No	Excellent	No



Information Gain Approach

Step 1: Find the Expected Entropy/Information of Dataset:

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

- **m: number of Classes or labels=2**
- **p_i: Probability of Class i**

$$Info(D) = -\left(\frac{9}{14} log_2 \frac{9}{14} + \frac{5}{14} log_2 \frac{5}{14}\right) = 0.94$$

Class YES of D **Class NO of D**

Information Gain Approach

Step 2: Iteratively find the **Best Atrribute** for **each level** of the Decision ree.

Iteration 1: Find Best Attribute for 0th level by computing **Gain of each Attribute**

a. Age attribute: $Entropy_{Age}(D) or\ Info_{Age}(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$

- **m: number of Classes or labels=2**
- **p_i: Probability of Class i**

$$= \frac{5}{14}\left(-\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5}\right) + \frac{4}{14}\left(-\frac{4}{4}log_2\frac{4}{4}\right) + \frac{5}{14}\left(-\frac{3}{5}log_2\frac{3}{5} - \frac{2}{5}log_2\frac{2}{5}\right) = 0.694$$

Class Variation in Youth

Class Variation in Middle_aged

Class Variation in Senior

Gain(Age)= Entropy (D) - $Entropy_{Age}$(D) = 0.940 - 0.694 = 0.246

Information Gain Approach

Step 2: Iteratively find the **Best Atrribute** for **each level** of the Decision ree.

Iteration 1: Find Best Attribute for For 0^{th} level Cont...

Similarly find the gain of attribute “income”, “student”, “credit_rating”

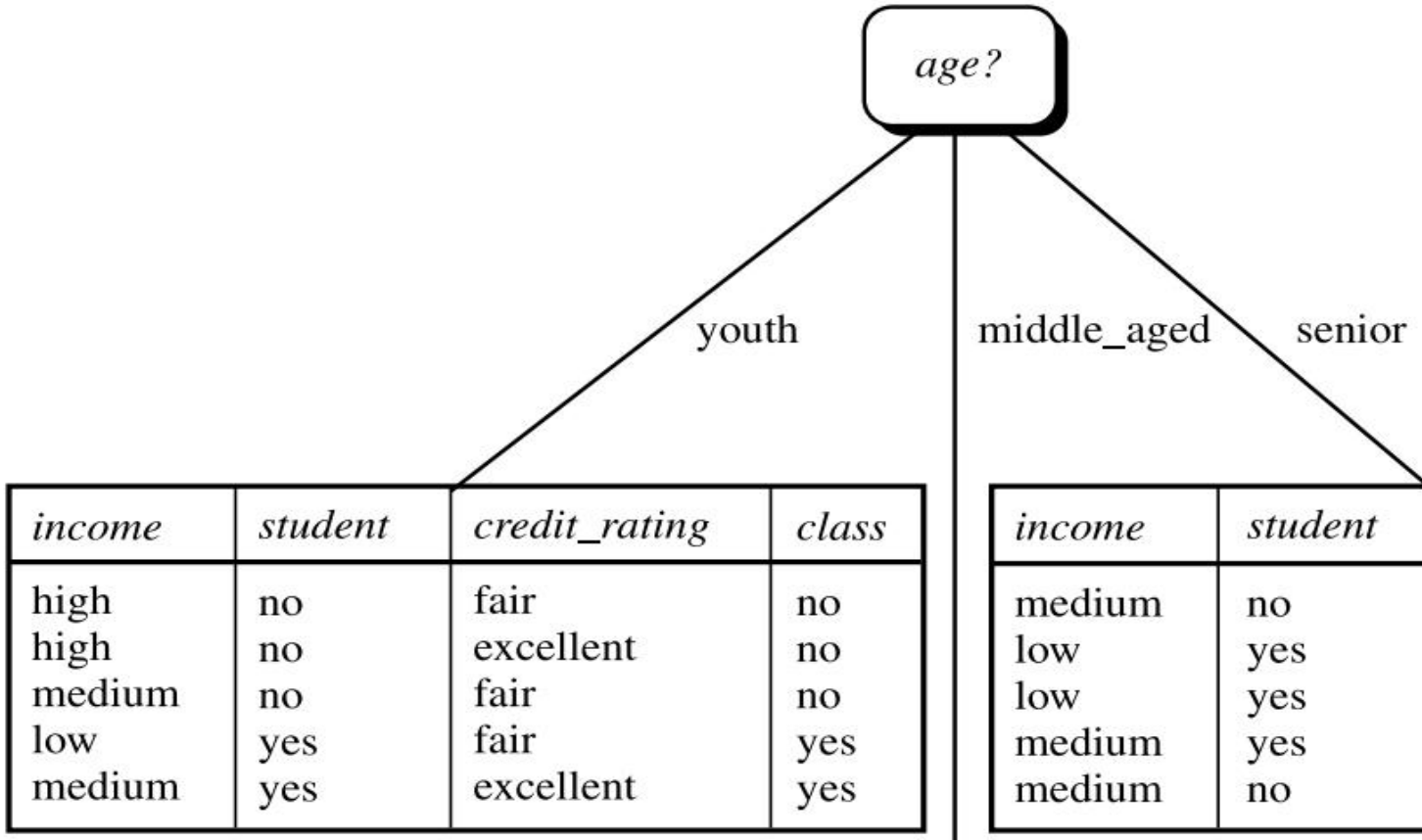
b. Gain of income = 0.029

c. Gain of student = 0.151

d. Gain of credit_rating = 0.048

Age has highest information gain from rest of the attribute, and hence is seleceted as splitting attribute.

Level 0 Decision Tree using 1st Splitting Attribute Age



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

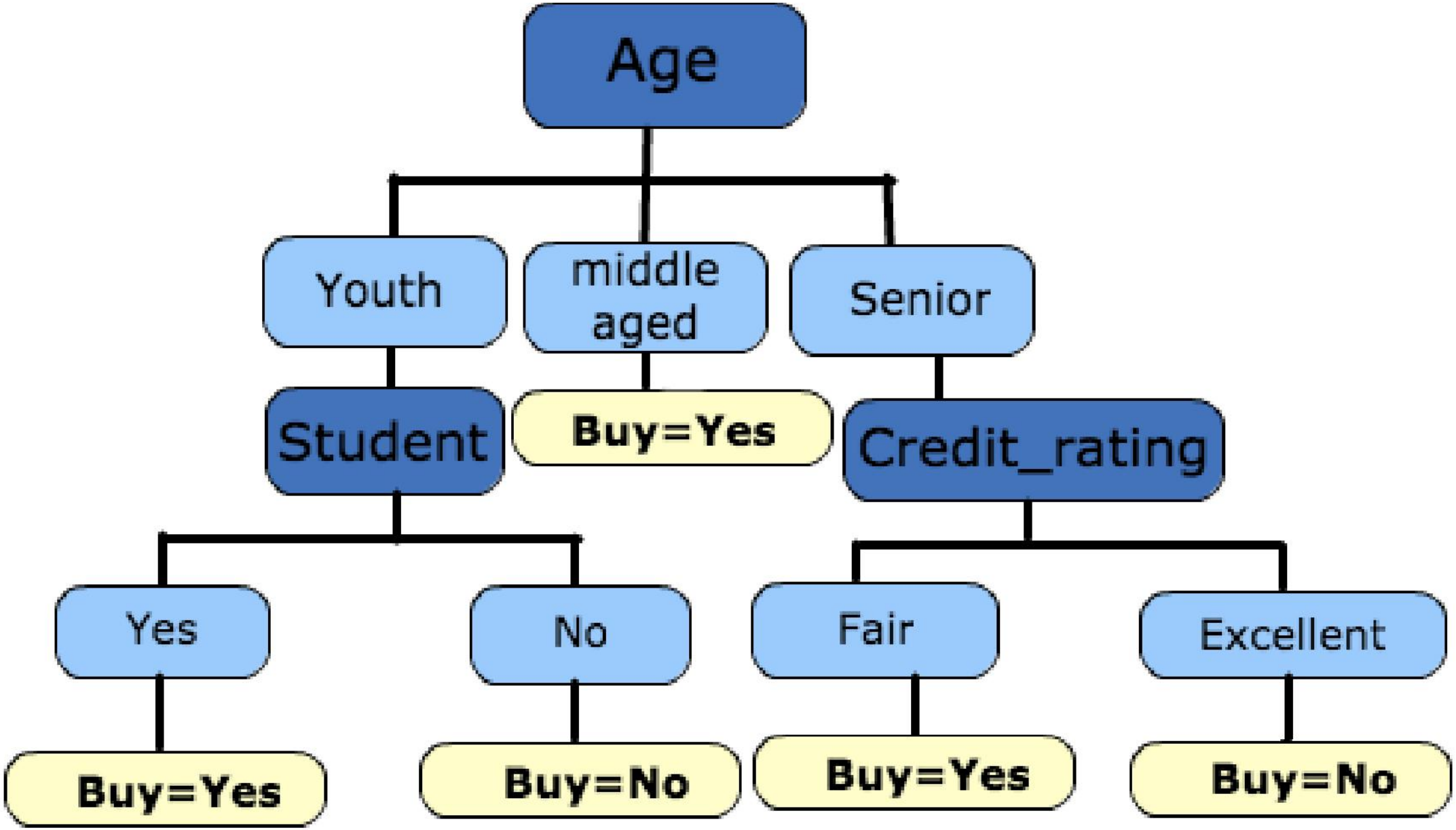
income	*student*	*credit_rating*	*class*
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

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

- **Tupple Falling on age=middle_aged all belong to the same class "YES" and hence a leaf can be created for that branch.**

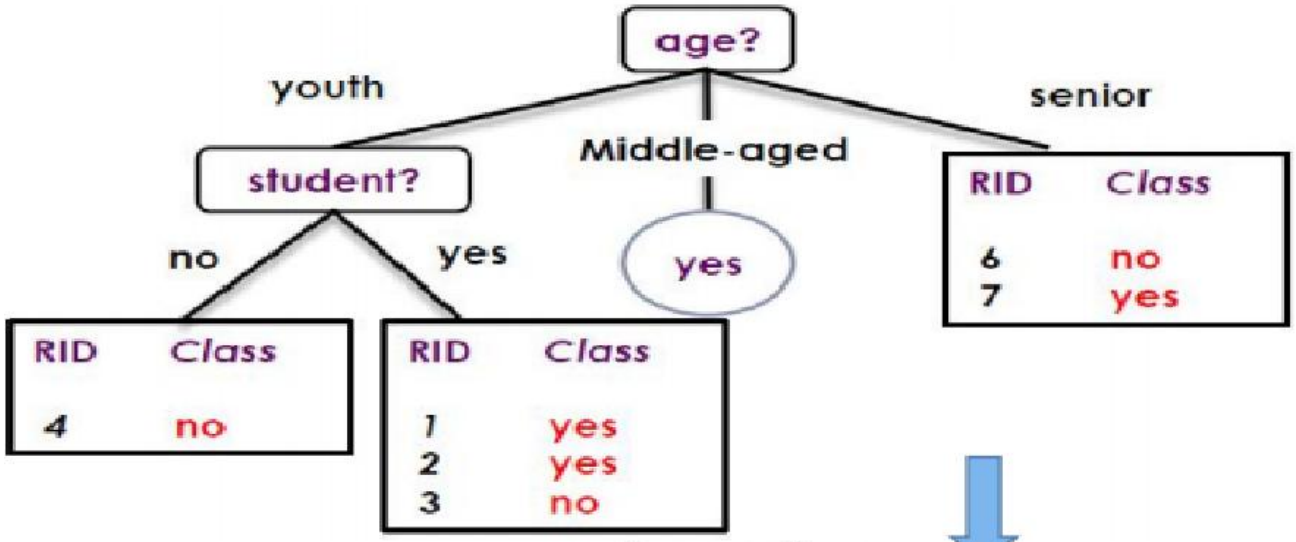
- **continue the tree construction with finding the best gain for data-partion age=youth and age=senior**

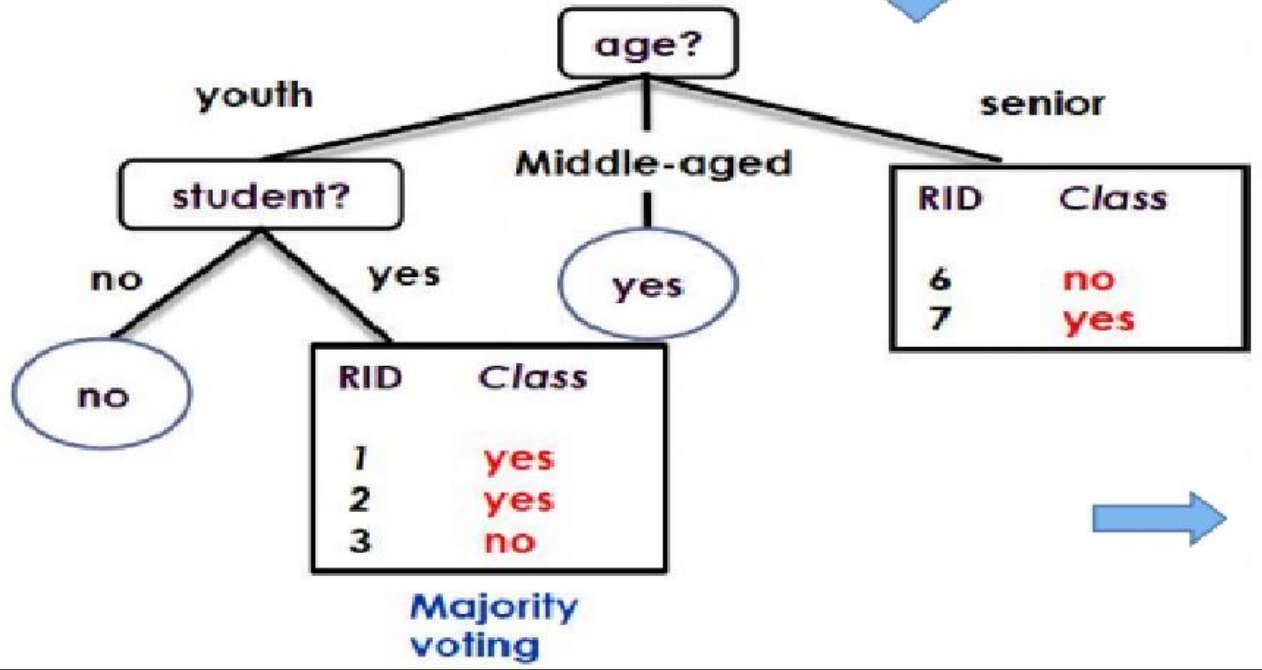
Final Decision Tree

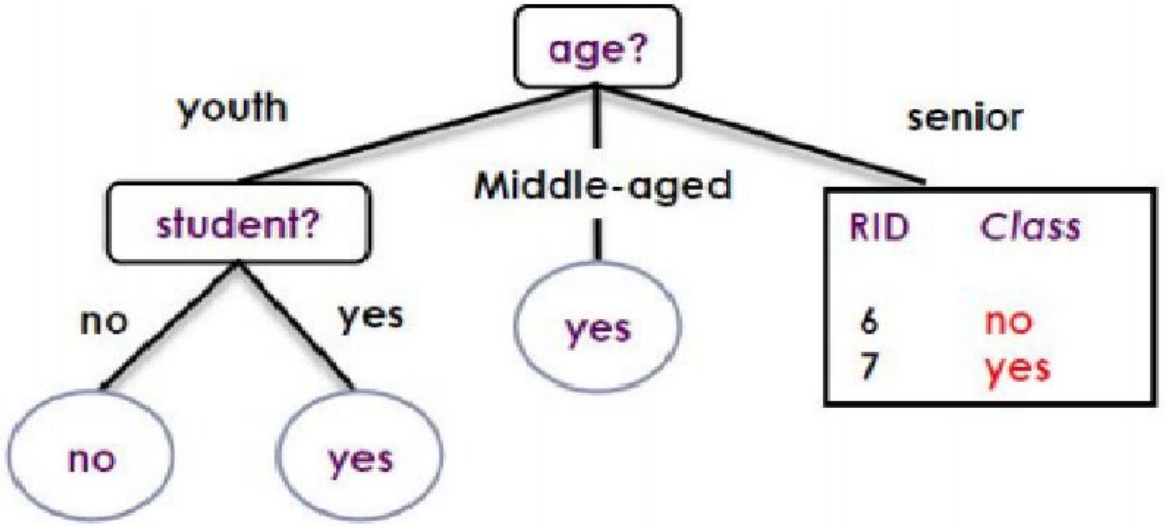


Example of Majority Voting

RID	age	student	credit-rating	Class: buys_computer
1	youth	yes	fair	yes
2	youth	yes	fair	yes
3	youth	yes	fair	no
4	youth	no	fair	no
5	middle-aged	no	excellent	yes
6	senior	yes	fair	no
7	senior	yes	excellent	yes







Extracting Classification Rule from Trees

- Represent the knowledge in the form of IF-THEN rules
- One rule is created for each path from the root to leaf.
- Each attribute-value pair along a path forms a conjuction
- The leaf node holds the class prediction

Example:

- If (Age == Youth) AND (Student == No) then buys_computer=”No”
- If (Age == Youth) AND (Student == No) then buys_computer=”Yes”
- If (Age == middle_aged) then buys_computer=”Yes”
- If(Age == senior) AND credit_rating == “Excellent” then buys_computer=”No”
- If(Age == senior) AND credit_rating == “fair” then buys_computer=”Yes”

Drawback of Information gain as attribute selection measure

- The information gain measure is biased towards tests with many outcomes. i.e. it prefers to select attribute having a large number of values.
- For instance, for the this dataset information gain prefers to split on attribute "Table income" that would result in a large number of pure partitions but each one containg one tuple.

$Info_{income}(D)=0$ [variation = 0]

=> $Gain_{income}(D) = Info(D)-Info_{income}(D)=Info(D)$

- Hence, Information gain w.r.t. income is maximal.

Solution:

C4.5, a successor of ID3

Tid	Refund	Marital Status	Taxable Income	Cheat
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

Gain Ratio

- C4.5, a successor of ID3, uses an extension to information gain known as gain ratio, which attempts to overcome this bias.
- It attempts a kind of normalization to information gain using a "*split information*" value defined analogously with *Info*(D) as follows.

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

- If there are "*v*" partitions in attribute A then SlpitInfo finds the number of tuples having that outcome w.r.t. total number of tuple.
- Now, to compute the best splitting attirbute the criteria "GainRatio" will be used instead of "Gain" which intern uses "SplitInfo".

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

- Conclusion: Gain Ratio applies a kind of normalization to information gain using "split information" value. the attribute with max gain ratio is selected as splitting attribute

Example of Gain Ratio

In the process of selecting best partition attribute nstead of finding gain of each attribute, now find the gain ratio of each attribute and then find the max gain ration

RID	Age	Income	Student	Credit_ Rating	Class: Buys Computer
1	-	High	-	-	No
2	-	High	-	-	No
3	-	High	-	-	Yes
4	-	Medium	-	-	Yes
5	-	Low	-	-	Yes
6	-	Low	-	-	No
7	-	Low	-	-	Yes
8	-	Medium	-	-	No
9	-	Low	-	-	Yes
10	-	Medium	-	-	Yes
11	-	Medium	-	-	Yes
12	-	Medium	-	-	Yes
13	-	High	-	-	Yes
14	-	Medium	-	-	No

Let computed Gain of attribute income = 0.029

Attribute income is having tree partitions:

Low, Medium, High

Low -> 4 tuples

Medium -> 6 tuples

High -> 4 tuples

$$SplitRatio_{income}(D)$$

$$= \underbrace{\frac{4}{14} log_2 \frac{4}{14}}_{\text{LOW}} + \underbrace{\frac{6}{14} log_2 \frac{6}{14}}_{\text{MEDIUM}} + \underbrace{\frac{4}{14} log_2 \frac{4}{14}}_{\text{HIGH}} = 1.557$$

$$GainRatio = \frac{Gain_A(D)}{SplitRatio_A(D)} = \frac{0.029}{1.557} = 0.019$$

Gini Index

- Gini Index is used in CART algorithm. It measure the impurity of database (D) and tries to minimizes the impurity
- It assumes all attributes to be categorical/continous valued and tries to find the best binary splitting position that gives minimum Gini Index and then find the best splitting attribute that can best reduce the impurity of the dataset D.

  - Step1: Finds the impurity of dataset D $[Gini(D)]$
  - Step 2: Find best binary splitting position of each attribute that gives the minimum Gini Index of that attribute. [Min of $Gini_A(D)$]
  - Step 3: Find the reduction in impurity due to each attribute [due to best splitting pos] and choose the best splitting attribute which can maximizes the reduction of impurity of dataset D. [Max of ΔGini(A)]

Gini Index

Step1: Finds the impurity of dataset D $\text{Gini(D)} = 1 - \sum_{i=1}^{m} P_i^2$

m: number of class

P_i: Probability of i^{th} class

Step 2: Find the best splitting position of each attribute that minimizes $\text{Gini}_A(D)$

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

- For instance income={Low, Medium, High} Number of splitting position= (2^v-2)
- Find the minimum $\text{gini}_A(D)$ [best binary splitting position] from these split positions for income.
- Similarly find best splitting position for all other attributes using $\text{gini}_A()$ index

Step3: Compute the reduction of impurity gini(A) of each attribute as follows and consider the attribute that maximizes the reduction impurity.

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

Advantages and Shortcomings of Decision Tree Classifications

- A decision tree construction process is concerned with identifying the splitting attributes and splitting criterion at every level of the tree.

- Major strengths are:
  - Decision tree able to generate understandable rules.
  - They are able to handle both numerical and categorical attributes.
  - They provide clear indication of which fields are most important for prediction or classification.

- Weaknesses are:
  - The process of growing a decision tree is computationally expensive. At each node, each candidate splitting field is examined before its best split can be found.
  - Some decision tree can only deal with binary-valued target classes.

Other Attribute Selection Measures

- CHAID: a popular decision tree algorithm, measure based on χ^2 test for independence
- C-SEP: performs better than info. gain and gini index in certain cases
- G-statistics: has a close approximation to χ^2 distribution
- MDL (Minimal Description Length) principle (i.e., the simplest solution is preferred):
  - The best tree as the one that requires the fewest # of bits to both (1) encode the tree, and (2) encode the exceptions to the tree
- Multivariate splits (partition based on multiple variable combinations)
  - CART: finds multivariate splits based on a linear comb. of attrs.
- Best attribute selection measure
  - Most give good results, none is significantly superior than others

Classifier -II
Naive Bayesian Classifier

Classification by Naive Bayesian Classifier

- **Bayesian Classifier** is a statistical classifier based on Bayes Theorem which can predict the probability that a given tuple belongs to a particular class.
- **Naive Bayesian Classifier** is a simple Bayesian classifier that exhibits high accuracy and speed compaared to Decision tree & Neural Network classifier.
- **Naive Assumption: [Class-Conditional Independence]:** The effect of an attribute value on a particular class is independent of other attribute value.

[Simplify the computation] => Naive

Introduction to Bayesian Classification

- Statistical method for classification
- Supervised Learning method.
- Assumes an underlying probabilistic model, the Bayes theorem
- Can solve problems involving both categorical and continuous valued attributes.
- Named after Thomas Bayes, who proposed the Bayes Theorem

Probability Basics: Prior Probability

A prior probability is the probability that an observation will fall into a group before one collect the data.

Example:

1. John conducts a single coin toss. What is the a priori probability of landing a head?

   A priori probability of landing a head = 1 / 2 = 50%.

2. A six-sided fair dice is rolled. What is the a priori probability of rolling a 2, 4, or 6, in a dice roll?

   The number of desired outcomes is 3 (rolling a 2, 4, or 6), and there are 6 outcomes in total.

   A priori probability of rolling a 2, 4, or 6 = 3 / 6 = 50%.

Probability Basics: Posterior Probability

A posterio probability / Conditional probability is the probability of an event occurring given that another event has already occurred.

For example, we might be interested in finding the probability of some event "A" occurring after we account for some event "B" that has just occurred.

The formula to calculate a posterior probability of A occurring given that B occurred:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} = \boxed{\frac{P(A) \times P(B \mid A)}{P(B)}}$$

Bayes' Theorem

Where, A, B = Events

P(B|A):- The probability of B occuring given that A has happened

P(A) & P(B):- The prior probability of event A & event B

Probability Basics: Example Calculating Posterior Probability

A forest is composed of 20% Oak trees and 80% Maple trees. Suppose it is known that 90% of the Oak trees are healthy while just 50% of the Maple trees are healthy.

Q:- Suppose that from a distance one can tell that a particular tree is healthy, what is the probability that the tree is an Oak tree?

$$P(A|B) = \frac{P(A) \times P(B|A)}{P(B)} \quad => \quad P(Oak|Healthy) = \frac{P(Oak) \times P(Healthy|Oak)}{P(Healthy)}$$

P(Oak): Probability that a tree is an oak is 0.2 (20%)

P(Healthy): Probability that a given tree is healthy= (0.2*0.9)+(0.8*0.5)=0.58

P(Healthy|Oak): The probability that a tree is healthy given that it's an Oak tree is 0.9 [90%]

$$P(Oak|Healthy) = \frac{0.2 \times 0.9}{0.58} = 0.3103$$

Probability Basics: Posterior Probability contd...

For an intuitive understanding of this probability, suppose the following grid represents this forest with 100 trees. Exactly 20 of the trees are Oak trees (20%) and 18 (90%) of them are healthy. The other 80 (80%) trees are Maple and 40 (50%) of them are healthy.

(O = Oak, M = Maple, Green = Healthy, Red = Unhealthy)

O	O	M	M	M	M	M	M	M	M
O	O	M	M	M	M	M	M	M	M
O	O	M	M	M	M	M	M	M	M
O	O	M	M	M	M	M	M	M	M
O	O	M	M	M	M	M	M	M	M
O	O	M	M	M	M	M	M	M	M
O	O	M	M	M	M	M	M	M	M
O	O	M	M	M	M	M	M	M	M
O	O	M	M	M	M	M	M	M	M
O	O	M	M	M	M	M	M	M	M

Out of all exactly 58 of them are healthy and 18 of these healthy ones are Oak trees. Thus, if a healthy tree has selected then the probability that it's an Oak tree is 18/58 = 0.3103.

Bayes' Theorem [useful in finding posterior probability]

Given:

- **X:** Data tuple described by measurement made on a set on n attributes
- **H:** Some hypothesis that the data tuple "X" belongs to class C

Problem: Determine P(H|X):- Given the evidence/observed data tuple "X", find the probability that the hypothesis H holds. i.e. Probability that tuple"X" belongs to class C

According to Bayes Theorem $$P(H|X) = \frac{P(X|H) \times P(H)}{P(X)}$$

P(H|X):- Posterior Probability of class "H" where attribute values $<x_1, \ldots x_n>$ are given.

P(H): Prior Probability of class "H" regardless of attributes values $<x_1, \ldots x_n>$

P(X|H): Posterior Probability of attribute values $<x_1, \ldots x_n>$ where class "H" is given

P(X): Prior Probability of attribute values $<x_1, \ldots x_n>$ regardless of class "H"

An Example of Bayes' Theorem

Example: Let customers are only described by attributes
age, income and credit_rating

- X: 35 years old customer with an income 40K and fair Credit_rating
- H: Hypothesis that the customer will buy a computer

Determine:-

Likelihood Prior

$$P(Y|X_1,\ldots,X_n) = \frac{P(X_1,\ldots,X_n|Y)P(Y)}{P(X_1,\ldots,X_n)}$$

Normalization Constant

An Example of Bayes' Theorem contd...

According to Bayes Theorem $$P(H|X) = \frac{P(X|H) \times P(H)}{P(X)}$$

Let hypothesis H is buying computer = "YES"

P(H|X)**:-** Posterior Probability that the ccustomer will buy computer="YSE" given that age = 35, income =40K and credit_rating = fair.

P(H): Prior Probability of class Yes regardless of attributes values <age, income, credit_rating>

P(X|H): Posterior Probability of attribute values <age=35, income=40K credit_rating = fair> where class buying computer ="YES" is given.

P(X): Prior Probability of attribute values <age=35, income=40K credit_rating = fair> regardless of class buying computer

Naive Bayes Classifier

- Given: D: - Set of tuples.

  X: -<x_1, ...x_n> where x_i is the value of attribute A_i

  m: Number of classes - C_1,..C_m

- Bayesian classifier predicts X belongs to Class C_i iff

$$P(Ci|X) > P(Cj|X) \quad \text{for } 1 \leq j \leq m \ \& \ j \neq i$$

or

Maximize Posterior Hypothesis: $$P(C_i|X) = \frac{P(X|Ci) \times P(Ci)}{P(X)}$$

Naive Bayes Classifier

In the maximization of hypothesis $P(C_i|X) = \frac{P(X|C_i) \times P(C_i)}{P(X)}$,

Find three values **1. P(X) 2. P(C_i) 3.P(X|C_i)**

Naive Assumption: [Class-Conditional Independence]: The effect of an attribute value on a particular class is independent of other attribute value.

1. Due to conditional independence of the attribute P(X)= P(x_1,...x_n) =1
2. P(C_i): Probability that class i is occuring in the dataset

   $P(C_i) = \frac{|C_{i,D}|}{|D|}$ where $|C_{i,D}|$ is the number of tuple containing class i
3. P(X|C_i)= P(x_1, ..., x_n | C_i).

Due to conditional independence, $P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$ = P(x_1|C_i)×...×P(x_n|C_i)

As P(X) is 1, Hence only maximize P(X|C_i) × P(C_i)

Problem

RID	Age	Income	Student	Credit_Rating	Class: Buys Computer
1	Youth	High	No	Fair	No
2	Youth	High	No	Excellent	No
3	middle_aged	High	No	Fair	Yes
4	senior	Medium	No	Fair	Yes
5	senior	Low	Yes	Fair	Yes
6	senior	Low	Yes	Excellent	No
7	middle_aged	Low	Yes	Excellent	Yes
8	Youth	Medium	No	Fair	No
9	Youth	Low	Yes	Fair	Yes
10	senior	Medium	Yes	Fair	Yes
11	Youth	Medium	Yes	Excellent	Yes
12	middle_aged	Medium	No	Excellent	Yes
13	middle_aged	High	Yes	Fair	Yes
14	senior	Medium	No	Excellent	No

For the given training dataset let the new customer X value as follows. Find the probability that customer will buy computer or not

or

Find the query X belongs to which class?

X = (Age=Youth

income = medium

student = Yes

credit_rating = Fair)

Problem

Step-1: Find Class Prior Probability $P(C_i)$ for i=1,2

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

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

Step-2: Find Conditional Probability of the query X or tuple X given is the class label.

Conditional Probability of X w.r.t. Class Yes

$$P(X|\text{Yes}) = \prod_{k=1}^{n} P(x_k|Yes)$$

= P(x1: age=youth|Yes) ×
P(x2: income=medium|Yes) ×
P(x3: student=yes|Yes) ×
P(x4: credit_rating=fair|Yes)

$$= \frac{2}{9} \times \frac{4}{9} \times \frac{6}{9} \times \frac{6}{9} = 0.22 \times 0.44 \times 0.66 \times 0.66 = 0.044$$

Conditional Probability of X w.r.t. Class No

$$P(X|\text{Yes}) = \prod_{k=1}^{n} P(x_k|No)$$

= P(x1: age=youth|No) ×
P(x2: income=medium|No) ×
P(x3: student=yes|No) ×
P(x4: credit_rating=fair|No)

$$= \frac{3}{5} \times \frac{2}{5} \times \frac{1}{5} \times \frac{2}{5} = 0.6 \times 0.4 \times 0.2 \times 0.4 = 0.019$$

Problem

Step-3: Find the class that maximizes $P(X|C_i) \times P(C_i)$

Max{ P(X|buys_Comp=Yes) × p(buys_computer = Yes),
P(X|buys_Comp=No) × p(buys_computer = No)
= {0.44 × 0.643, 0.019 × 0.357}
= {0.028, 0.007} = 0.028 {Class "Yes" Maximizes the conditional probability of X}

Hence, the Naive Bayesian classifier predicts buys_comp is yes for given tuple X

Problem

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

For the given training dataset let the new customer X value as follows. Find the probability that Golf will be played or not?

Find the query X belongs to which class?

X = (Outlook=Rainy

Temp = Hot

Humidity = High

Windy = False)

Partial Solution

P(p) = 9/14
P(n) = 5/14

outlook	
P(sunny|p) = 2/9	P(sunny|n) = 3/5
P(overcast|p) = 4/9	P(overcast|n) = 0
P(rain|p) = 3/9	P(rain|n) = 2/5
temperature	
P(hot|p) = 2/9	P(hot|n) = 2/5
P(mild|p) = 4/9	P(mild|n) = 2/5
P(cool|p) = 3/9	P(cool|n) = 1/5
humidity	
P(high|p) = 3/9	P(high|n) = 4/5
P(normal|p) = 6/9	P(normal|n) = 2/5
windy	
P(true|p) = 3/9	P(true|n) = 3/5
P(false|p) = 6/9	P(false|n) = 2/5

Advantage

- Simple, fast and easy to implement.Needs less training data. Highly scalable.
- Handles continuous and discrete data.
- Easily updatable if training dataset increases

Uses

- Text Classification
- Spam Filtering
- Hybrid Recommender
- Online Application

Disavantages

- Assumption: Class conditional independence, therefore loss of accuracy
- Practically, dependecies exist among variables.

Example: Hospitals, patients profile, family history, didease symptoms etc

Major Issues

- **Violations of Independence Assumpyion:** It assumes that the effect of an attribute values on a given class is independent of the values of the other attribute.
  - ➢Solution: Bayesian beleief network classifier allows the representation of dependencies among subsets of attributes
- **Zero Conditional Probability problem:** If a given class and feature value never occur together in the training set then the frequency based probability esimate will be zero. This is problematic since it will wipe out all information in the other probabilities when they are multiplied.
  - ➢Therefore often it is desirable to incorporate a small sample correction in all possible option. Or consider the occurance/probability as 1 instead of zero which can be negligible in the consideration of maximum.

K-Nearest Neighbor

(Lazy Learners or Learning from your Neighbor)

Eager Learners Vs. Lazy Learners

Eager Learner

- Eager Learner approach of classification constructs a generalized classification model just after receiving the training tuples (much before receiving new/test tuple), and be ready and eager to classify the new unseen tuple.
- More work when training tuple is received and less work when making a classification.
- Less Expensive but not support incremental learning
- e.g. Decision Tree, Bayesian Classifier, Backpropagation, Association Rule Mining

Lazy Learner

- Lazy Learner approach stores the received training tuples and wait to the last minute of receiving the new/test tuple. Only when it receives the test tuple, it constructs the generalized classification model based on similarity among test tuple and training tuples.
- Less work (only store) when training tuple is received and more work when making a classification
- More Expensive but support incremental learning
- e.g. K-Nearest Neighbor, case-based learning

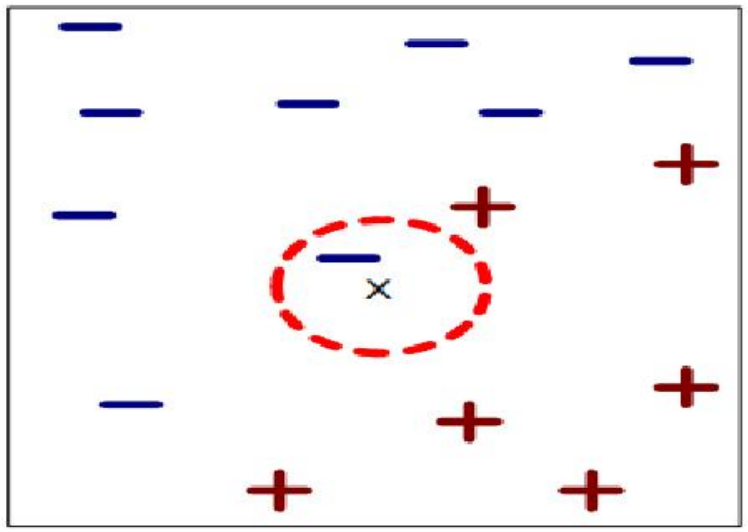
K-Nearest Neighbor Classifier

Nearest based classifiers are based on learning by analogy i.e. by comparing a given test tuples with traing tuples that are similar to it.

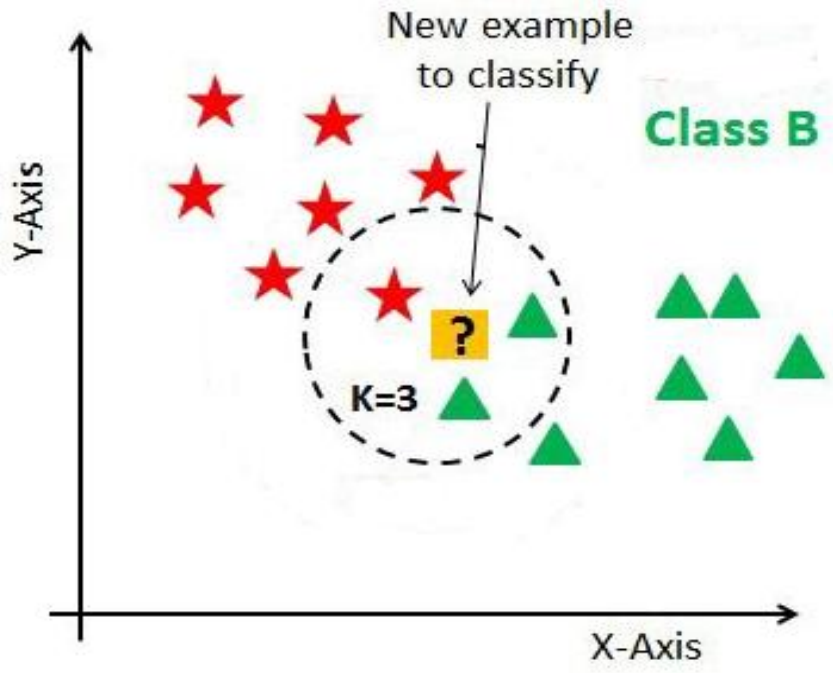
- Let the training tuples are described by n attributes. All instances of training tuples are correspond to points in the n-dimentional space and accordingly each tuple represents a point in an n-dimensional space
- A **k-nearest-neighbor classifier** searches the pattern space for the k training tuples that are closest to the unknown tuple.

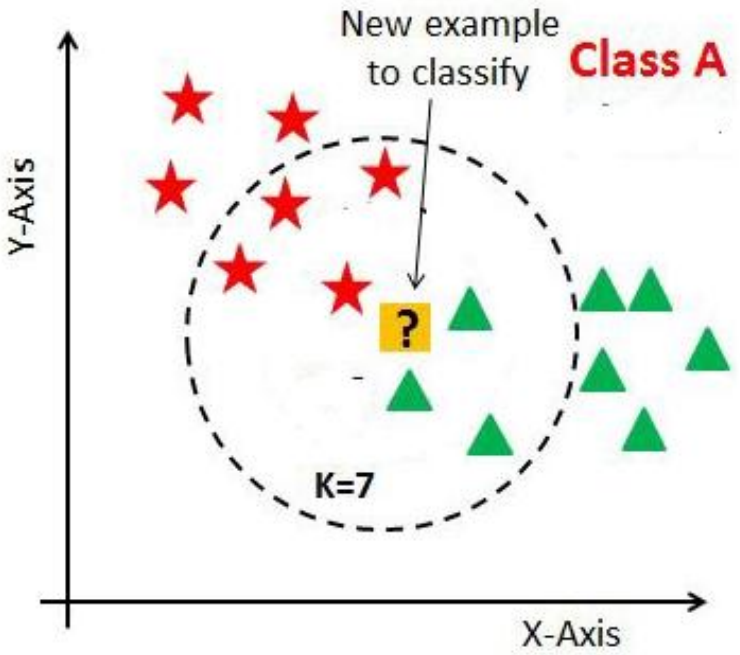
Prediction for test data is done on the basis of its neighbour

Properties of KNN



(a) 1-nearest neighbor





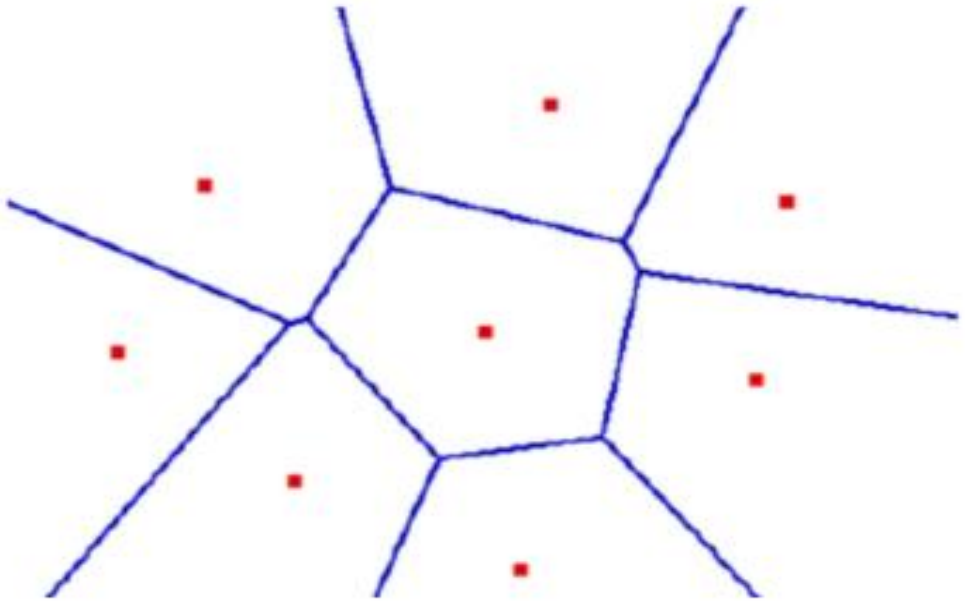
- How many neighbors should be consider? i.e. what is k?
- How do we measure distance?
- Should all points be weighted equally, or some points have more influence than others?

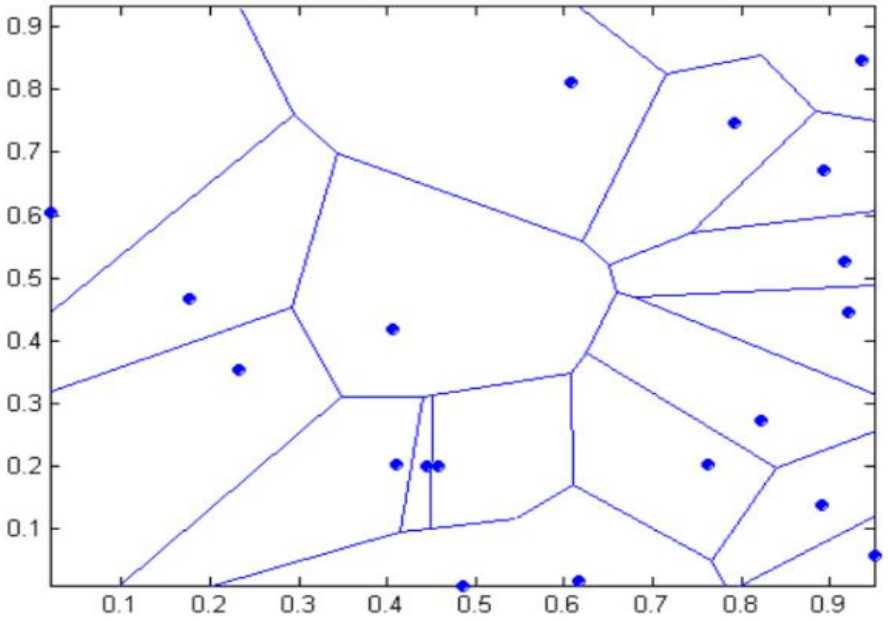
➢ How many neighbors should be consider? (what is k)

- At K=1, the KNN model tends to closely follow the training data and thus shows a high training score and low test score, indicating overfitting.
- The problem can be solved by tuning the value of k_neighbors parameter. As we increase the number of neighbors k, the model starts to generalize well, but increasing the value too much would again drop the performance.
- Therefore, it's important to find an optimal value of K, such that the model is able to classify well on the test data set with minimum error.
- Setting K value to the square root of number of training tuple or half of the square root of number of training tuple can lead to better result.
- For an even number of classes (e.g. 2 or 4) it is a good idea to choose a K value with an odd number to avoid a tie. And the inverse, use an even number for K when you have an odd number of classes

Decision Boundary: Voronoi Diagram

- Given a set of data, Voronoi diagram describes the areas that are nearest to any given point.
- Each line segment is at equidistance between two points of opposite class.

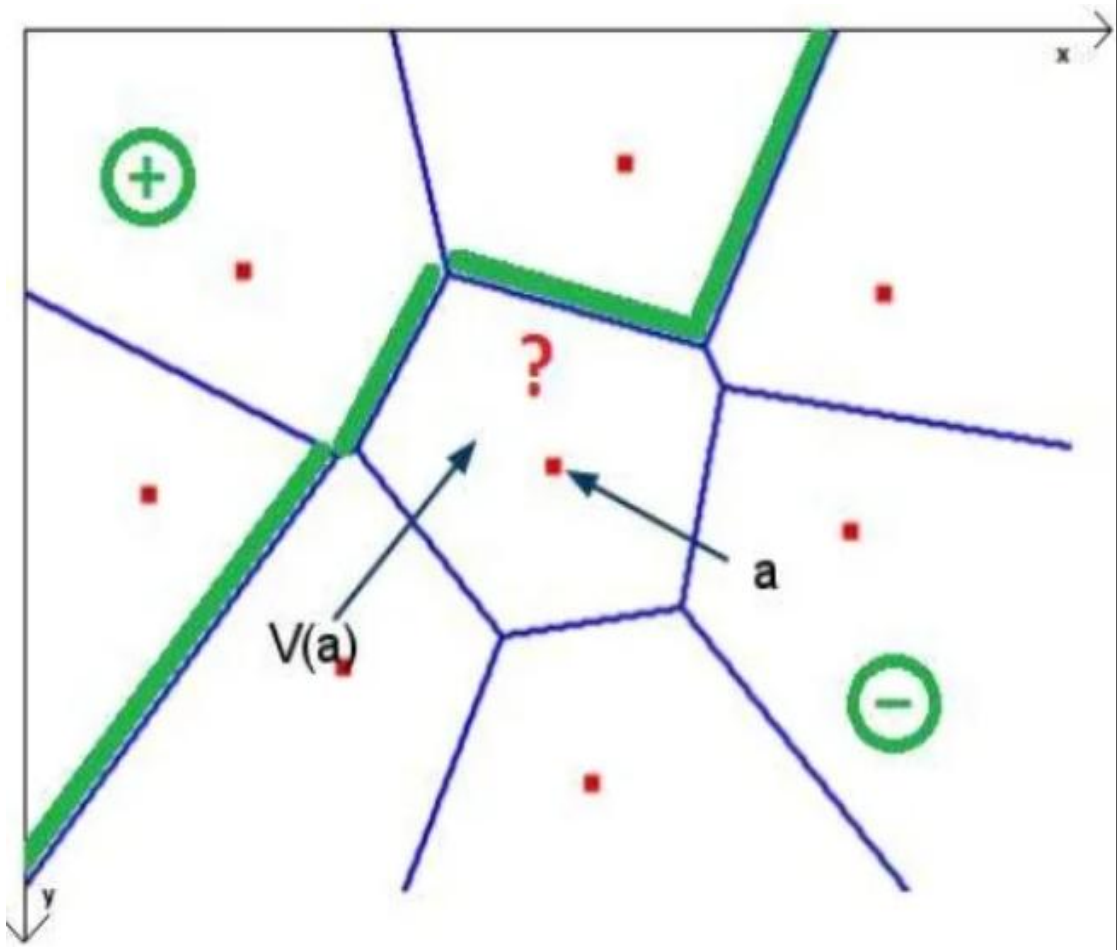




A Voronoi diagram divides the space into such cells that each cell contains one sample and every location within the cell is closer to that sample than to any other sample. .

Decision Boundary: Voronoi Diagram

- Let the requirement is to classify data “?” using the voronoi diagram.
- For this it is required to contstruct the voronoi diagram decision boundary.
- Let the data “?” is located in the voronoi area of “a” (diagram) which is in the negative side of the decision boundary or belongs to negative class. As this area contains all the points closest to “a” compared to other instance, the data “?” belongs to class to which a belongs, i.e. negative class.



How to Measure Distance

$$Euclidean\ distance:\ d(x,y) = \sqrt{\sum (x_i - yi)^2}$$

$$Squared\ Euclidean\ distance:\ d(x,y) = \sum (x_i - yi)^2$$

$$Manhattan\ distance:\ d(x,y) = \sum |(xi - yi)|$$

From these Euclidean distance is more preferable as it gives equal important to each feature

Example: 1

- Let given is the data from the questionaires survey with two attributes (X1: Acid durability and X2: Strength) to classify whether a special paper tissue is good or not. Classify the new paper tissue with features: X1=3 and X2=7.

X1: Acid Durability	X2: Strength	Y: Class
7	7	Bad
7	4	Bad
3	4	Good
1	4	Good

Example-1: Solution

Step 1: Intialize and Define K

K=1 Overfitting Problem

Number of tuples is even, hence consider K as odd.

Let K = 3 for this problem

Step 2: Compute distance between input sample and training samples using Euclidian distance measure.

$$dist((x, y), (a, b)) = \sqrt{((x - a)^2 + (y - b)^2)}$$

X1: Acid Durability	X2: Strength	Y: Class	Distance from (3, 7)
7	7	Bad	sqrt((7-3)^2 + (7-7)^2)=4
7	4	Bad	sqrt((7-3)^2 + (4-7)^2)=5
3	4	Good	sqrt((3-3)^2 + (4-7)^2)=3
1	4	Good	sqrt((1-3)^2 + (4-7)^2)=3.6

Example-1: Solution

Step 3: Determine K nearest neighbor those are at minimum distance and rank them.

X1: Acid Durability	X2: Strength	Y: Class	Distance from (3, 7)	Rank
7	7	Bad	4	3
7	4	Bad	5	4
3	4	Good	3	1
1	4	Good	3.6	2

Step 4: Apply Simple Mojority

There are 2 “Good” and 1 “Bad”. Thus the new paper tissue will be classified as Good due to simple majority

Example 2

Given is the customer detail as customer's age, income, credit card no. and their corresponding class. Find the class label for the new tuple/sample:

Customer: RRR, Age: 37, Income: 50K, Cr Card: 2, Class- ?

Customer	Age	Income	No. of Credit Cards	Class
XXX	35	35K	3	No
YYY	22	50K	2	Yes
ZZZ	63	200K	1	No
PPP	59	170K	1	No
QQQ	25	40K	4	Yes
RRR	**37**	**50K**	**2**	**????**

Example-2: Solution

Find the Euclidian Distance of new tuple from each training data

Customer	Age	Income	No. of Credit Cards	Class	Distance
XXX	35	35K	3	No	$\text{sqrt}((35-37)^2+(35-50)^2+(3-2)^2)=15.16$
+YYY	22	50K	2	Yes	$\text{sqrt}((22-37)^2+(50-50)^2+(2-2)^2)=15$
ZZZ	63	200K	1	No	$\text{sqrt}((63-37)^2+(200-50)^2+(1-2)^2)=152.2$
PPP	59	170K	1	No	$\text{sqrt}((59-37)^2+(170-50)^2+(1-2)^2)=122$
QQQ	25	40K	4	Yes	$\text{sqrt}((25-37)^2+(40-50)^2+(4-2)^2)=15.74$
RRR	**37**	**50K**	**2**	**????**	

Majority of {Yes, Yes, No} is Yes. Hence Class of new tuple is Yes

Should all Attributes be weighted equally, or Some Attributes have more influence than others?

Some attributes (like income, loan) outweighs some smaller ranged attributes (like binary or categorical or numeric like age) and thus, the result obtained can be a biased one.

Normalization is a solution to it: Max-Min Normalization generally used

Age	Loan	Default	Distance
25	$40,000	N	102000
35	$60,000	N	82000
45	$80,000	N	62000
20	$20,000	N	122000
35	$120,000	N	22000
52	$18,000	N	124000
23	$95,000	Y	47000
40	$62,000	Y	80000
60	$100,000	Y	42000
48	$220,000	Y	78000
33	$150,000	Y	8000
48	**$142,000**	**?**	

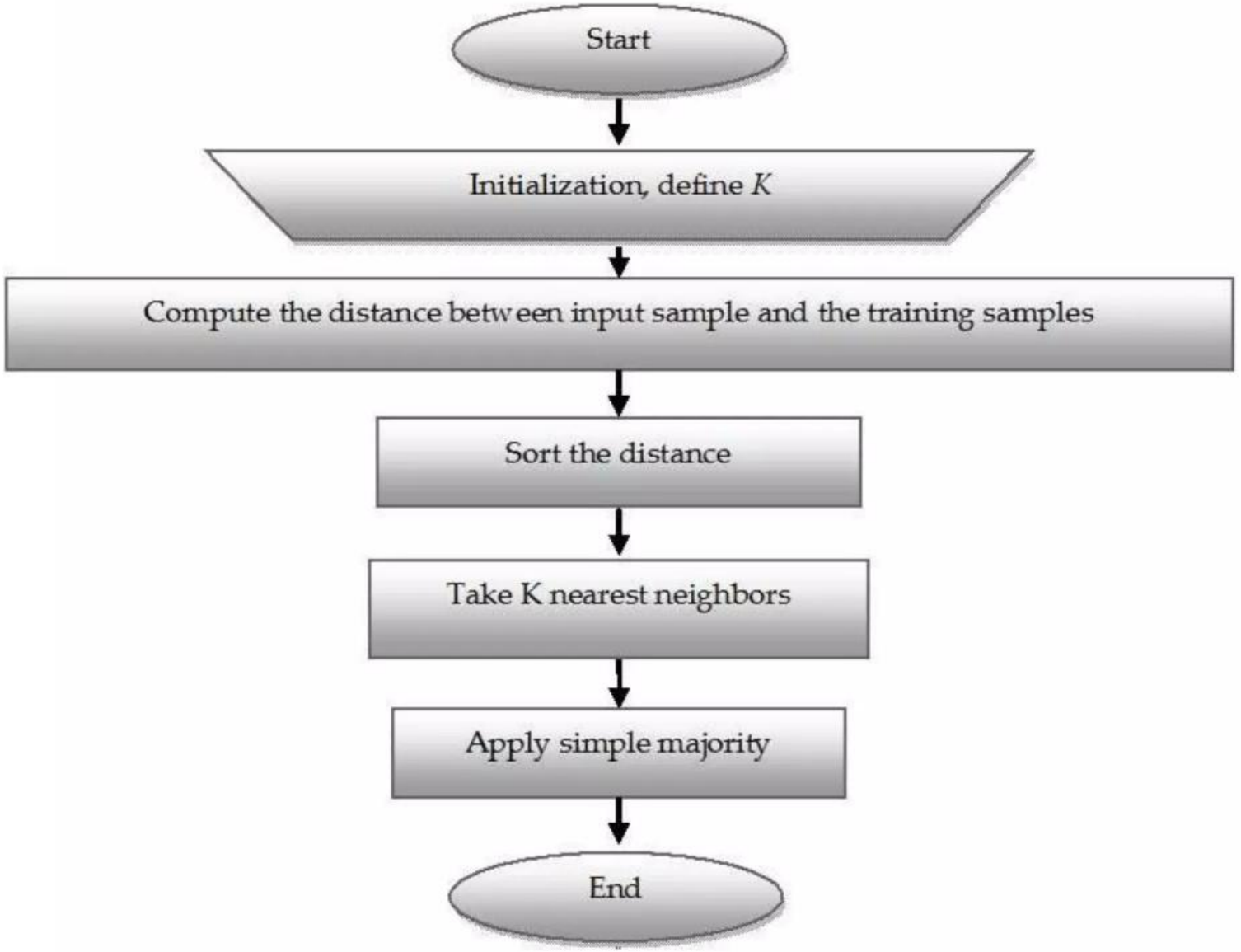
Solution : Normalization

Age	Loan	Default	Distance
0.125	0.11	N	0.7652
0.375	0.21	N	0.5200
0.625	0.31	N	0.3160
0	0.01	N	0.9245
0.375	0.50	N	0.3428
0.8	0.00	N	0.6220
0.075	0.38	Y	0.6669
0.5	0.22	Y	0.4437
1	0.41	Y	0.3650
0.7	1.00	Y	0.3861
0.325	0.65	Y	0.3771
0.7	**0.61**	**?**	

Standardized Variable

$$X_s = \frac{X - Min}{Max - Min}$$

KNN Classifier Algorithm



Advantage:

- No training period: There is no discriminative function or model build at the traing stage
- As there is no training period, any new data can be added seamlessly without impacting accuracy of the result.
- Easy to implement
- Best used where probability is unknown
- Useful for non-linear data as no assumption is required

Disadvantage:

- Computationally expensive and more space complexity
- Output depends on choosen K value whic can reduce accuracy for some K value
- Does not work well with large data set and high dimension
- Sensitive to noisy, missing and outlier data
- Need normlization

Application

- Used in classification and regression
- Used in pattern recognition
- Used in gene expression, protein-protein prediction, get 3D structure of the protein
- Used to measure document similarity

Classification by Backpropagation [Multilayer Perceptron Network]

Artificial Neural Networks : An Introduction

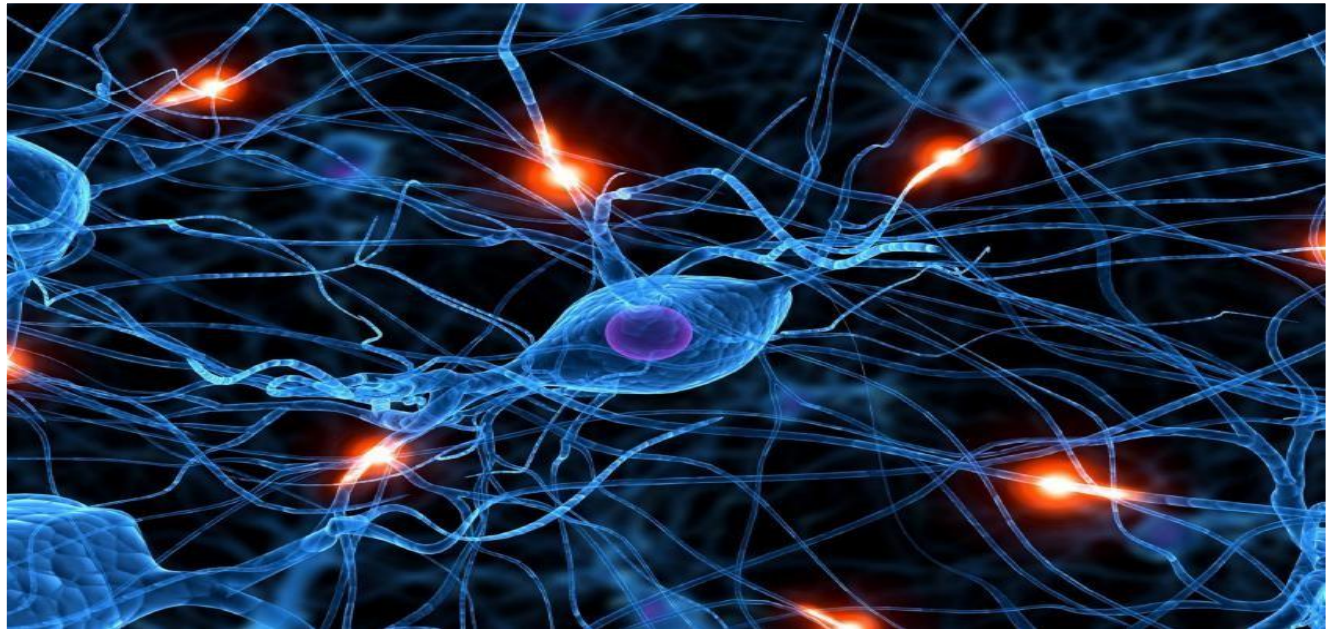
Artificial Neural Network (ANN)

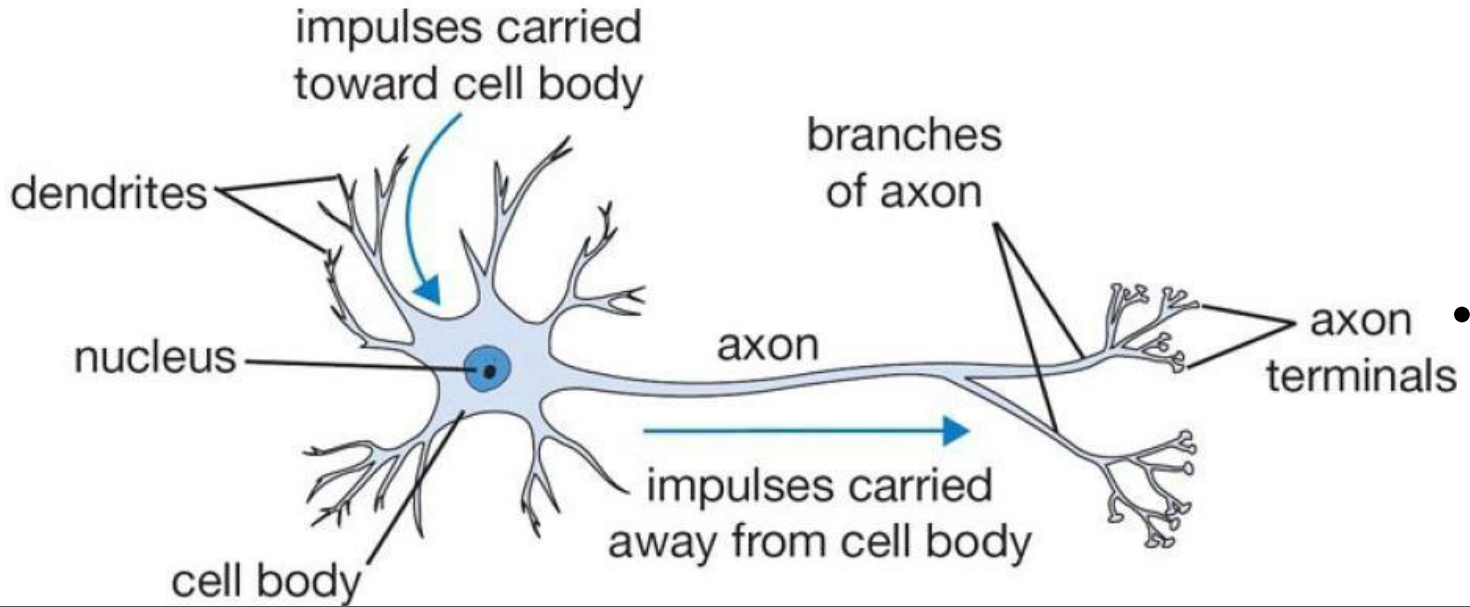
- ANN is a network model having a set of connected input-output units where each connection has a weight associated with it. The network/model tries to learn by its own through the adjustment of weights [Learning process].
- The alternative name of ANN: connectionist/ parallel distributed processing/ neural computation/ adaptive networks..

Multilayer Feed-Forward Neural Network

- The ANN model used for the classification purpose uses a multilayer feed forward neural network where the model iteratively learns a set of weights for prediction of the class label of the tuples.
- It consists of input layer, one or more hidden layer and an output layer.

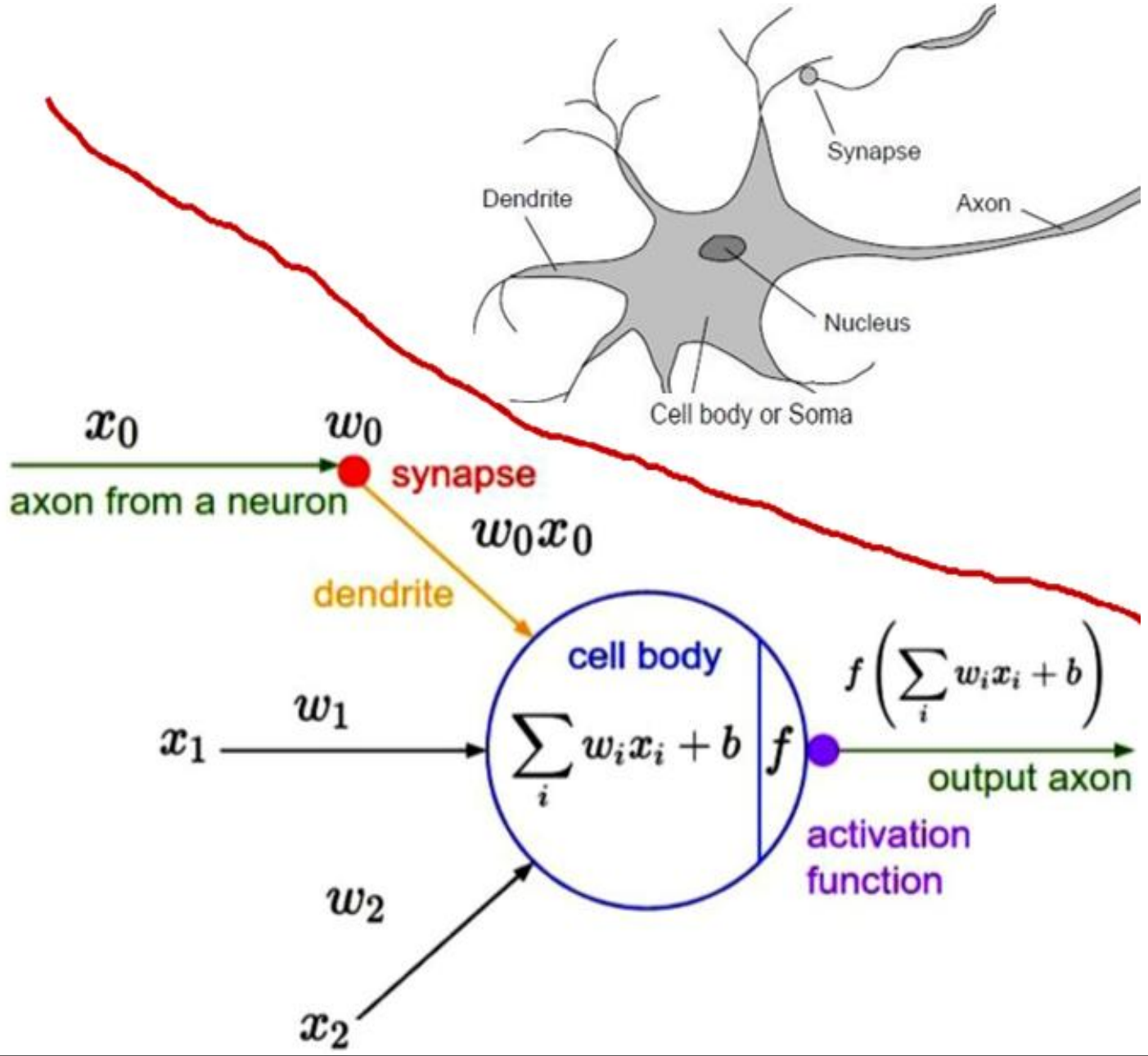
The Structure of Biological Neurons





- The brain consists of a complex network of cells called neurons.
- Neurons communicate by transimitting electrochemical signals throughout the network.
- Each input signal to a neuron can inhibit or excite the neuron. When the neuron is excited enough (exceeds a certain amount of threshold), it fires its own electrochemical signal.
- A neuron has a cell body, a branching input structure (the dendrIte) and a branching output structure (the axOn)

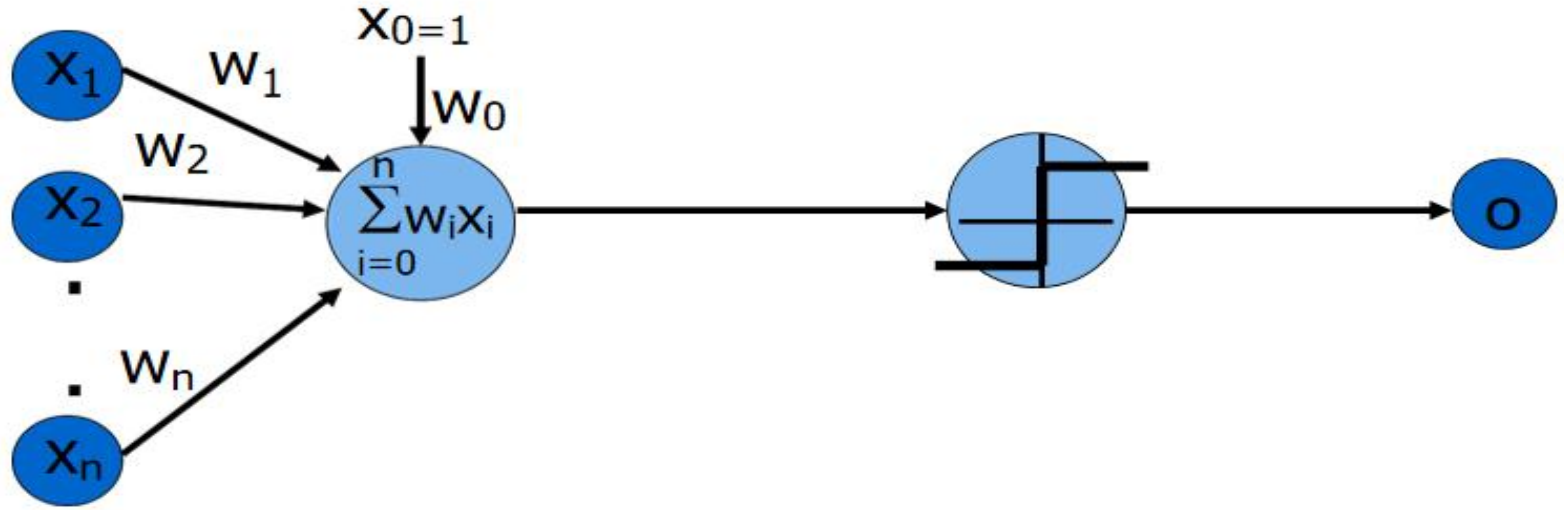
Properties of Artificial Neural Nets (ANNs)



- ANN posess a large number of processing elements called nodes/neurons which operate in parallel
- Neurons are connected with others by connection link. Each link is associated with weights and contains information about the input signal.
- Each neuron has an internal state of its own which is a function of the inputs that neuron receives- Activation function.
- The network Learn by tuning the connection weights

Important terminologies of ANNs

1. **Perceptron:** A Perceptron is a type of artificial neuron which takes several binary inputs $<x_1, x_2, \ldots, x_n>$, weights for respective inputs, bias value per node and produces a single binary output.



2. **Weight:** Each neuron is connected to every other neuron by means of directed links. Links are associated with weights. The weight value lies between 0 to 1.

3. **Bias** is another weight included by adding a component $x_0 = 1$ to the input vector X.

$$X=(1,X_1,X_2 \ldots X_i,\ldots X_n)$$

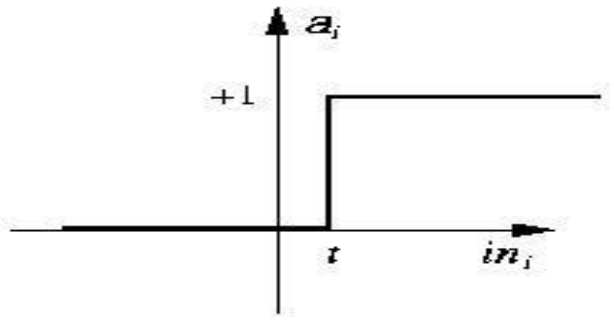
Bias is of two types: - Positive bias: increase the net input

- Negative bias: decrease the net input

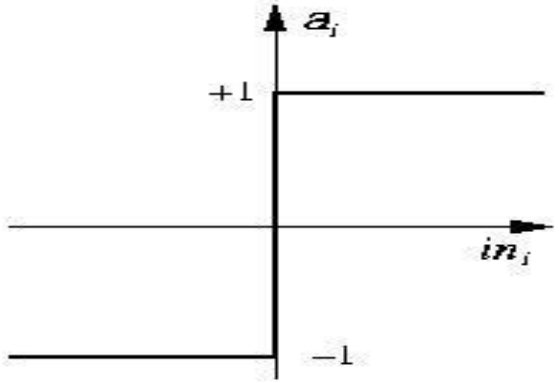
Important terminologies of ANNs...

4. Activation Function: It is used to calculate the output response of a neuron. Sum of the weighted input signal is applied with an activation to obtain the response.

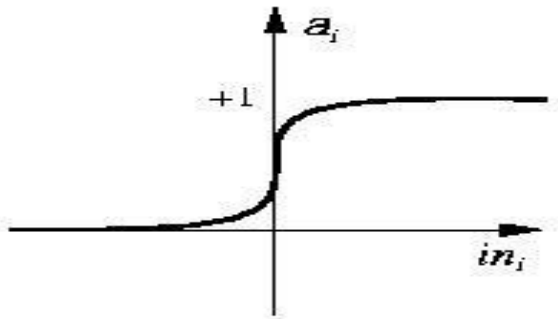
- Activation functions can be linear or non linear
- Types of Activation function
  - Identity function
  - Single/binary step function
  - Discrete/continuous **sigmoidal function**.



(a) Step function



(b) Sign function



(c) Sigmoid function

Important terminologies of ANNs...

5. Target Value (t): This is the known class value to which the perceptron output should match. When the network is under training phase, it is required to give the target value in addition to input, weight and bias value to compare the network's accuracy.

6. Error: The error value is the amount by which the value output by the network differs from the target value. For example, if we required the network to output 0 and its output as 1, then error =0-1= -1

7. Learning Rate (ɳ): Used to control the amount of weight adjustment at each step of training. Learning rate ranges from 0 to 1. It determines the rate of learning in each time step/ iteration.

It is a process by which a neural network adapts itself to a stimulus/model by making proper parameter adjustments, resulting in the production of desired response

Two kinds of learning

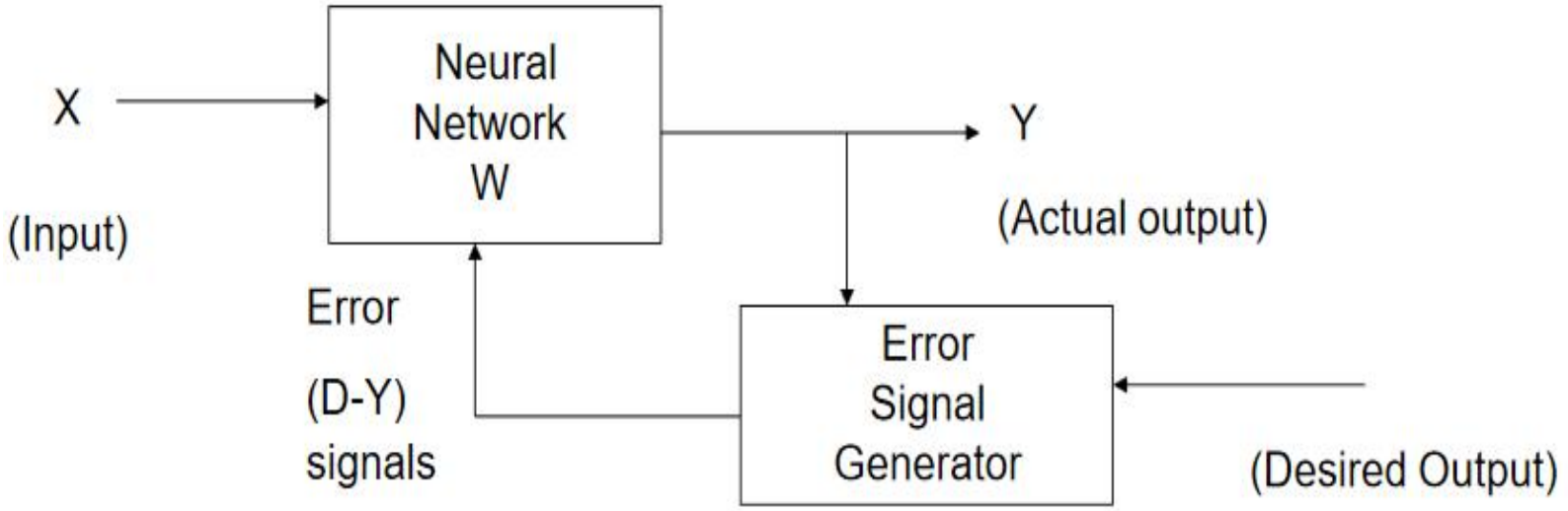
- **Parameter learning**:- connection weights are updated
- **Structure Learning**:- change in network structure

Important terminologies of ANNs...

5. Training: The process of modifying the weights of the connectors between network layers with the objective of achieving the expected output is called training a network.

Traing a network can be achieved through

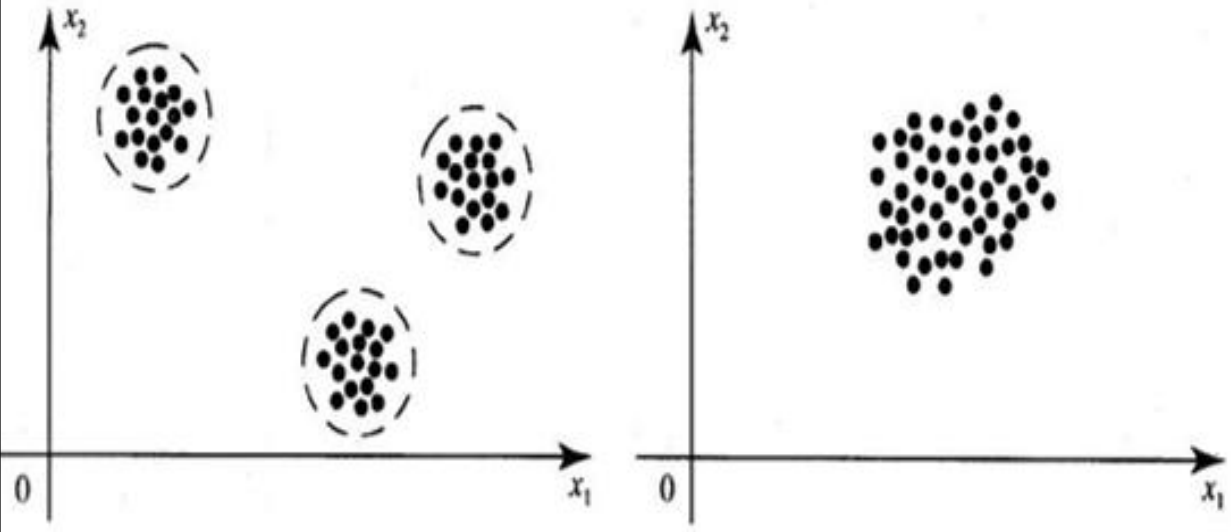
- Supervised learning
- Unsupervised learning
- Reinforcement learning



- **Supervised learning**

In supervised training, the network is trained by presenting it a sequence of training inputs (patterns), each with an associated target output value. Weights in the network are adjusted according to a learning algorithm.

Important terminologies of ANNs...

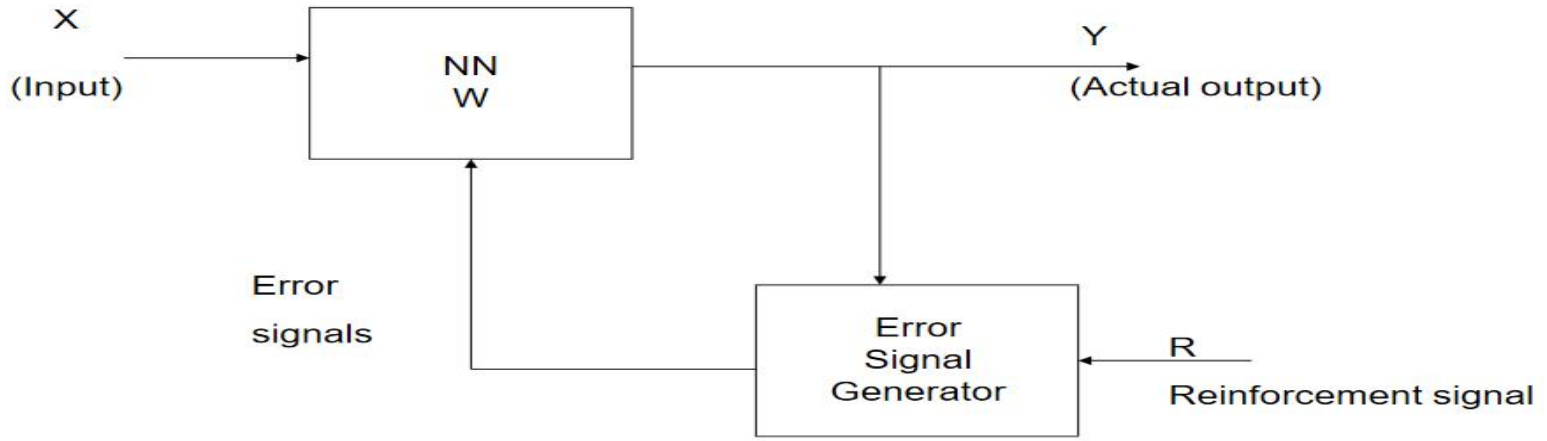


– **Unsupervised learning**

In unsupervised training, a sequence of training inputs is provided, but no target output values are specified. The weights are adjusted according to a learning algorithm. All similar input patterns are grouped together as clusters.

-- **Reinforcement learning**

If less information is available about the target output values (critic information). Learning based on this critic information is called reinforcement learning and the feedback sent is called reinforcement signal. Feedback in this case is only evaluative and not instructive



Important terminologies of ANNs...

8. Perceptron Learning Rule:

The perceptron compute the output **(Y)** on the basis weight(s), bias value and activation function and compare the computed output with the target output **(t)**

- If the computed output is correct (t = = Y) the weights w_i are not changed
- If the output is incorrect (t ≠ y) the weights w_i are changed by value Δw_i such that the output of the perceptron for the new weights will be closer to target t.

$$w_{ij} = w_{ij} + \Delta w_{ij}$$
$$\text{where } \Delta w_{ij} = \eta Err_j Y_i$$

- Err is the obtained error
- Yi is the output of lower ith unit
- η is Learning Rate (small constant <1)

The algorithm converges to the correct classification

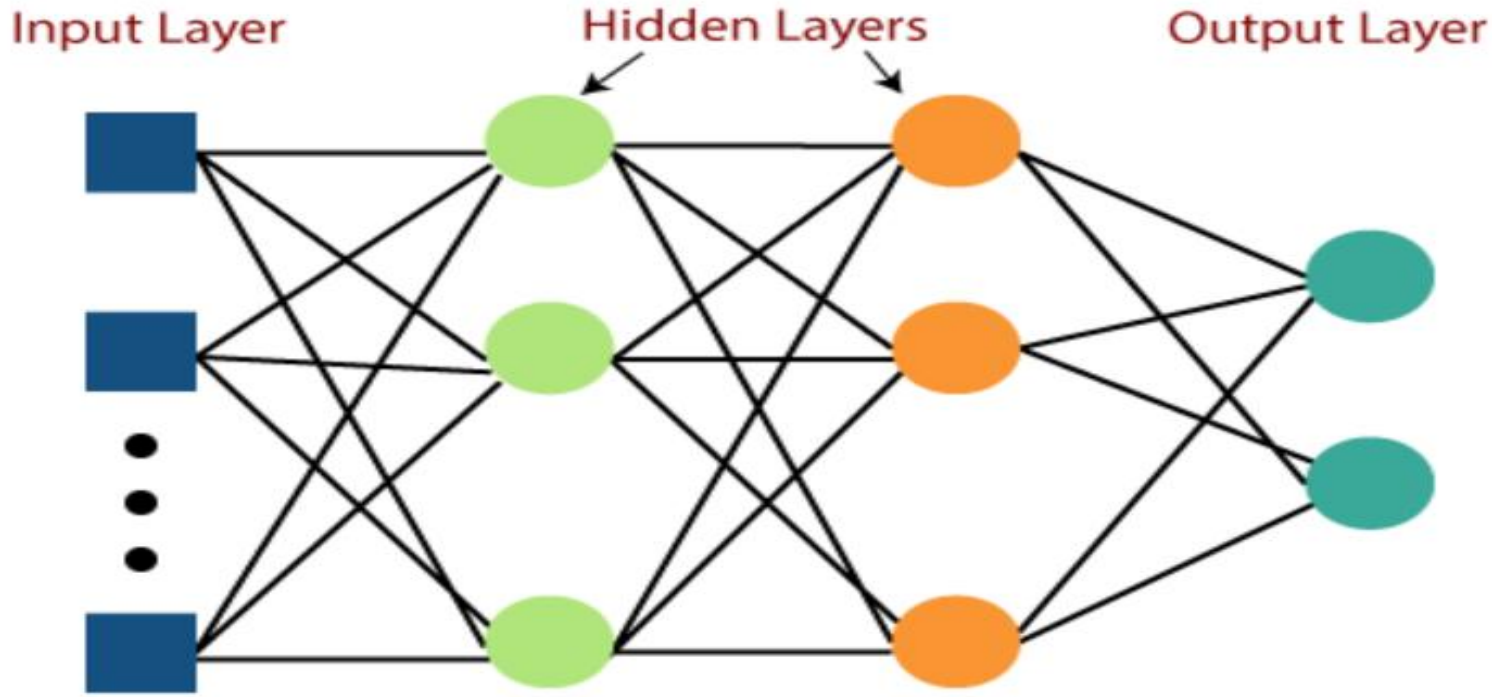
- if the training data is linearly separable
- and η is sufficiently small

Important terminologies of ANNs...

9. Multilayer Perceptron (MLP):

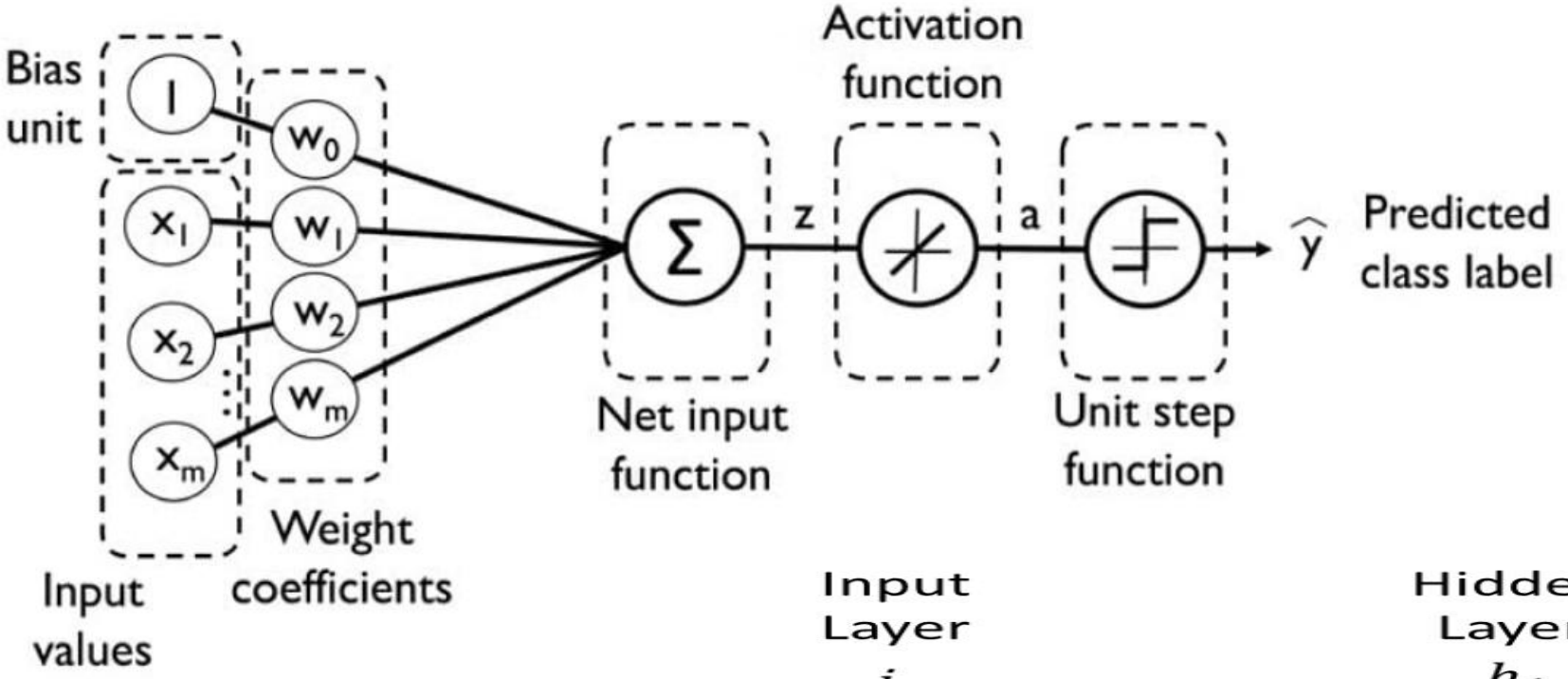
Multi-layer perception is a fully connected dense multiple layers neural network which transform any input dimension to the desired dimension.

A multi-layer perceptron has one input layer and for each input, there is one neuron(or node), it has one output layer with a single node for each output and it can have any number of hidden layers and each hidden layer can have any number of nodes. A schematic diagram of a Multi-Layer Perceptron (MLP) is depicted below.



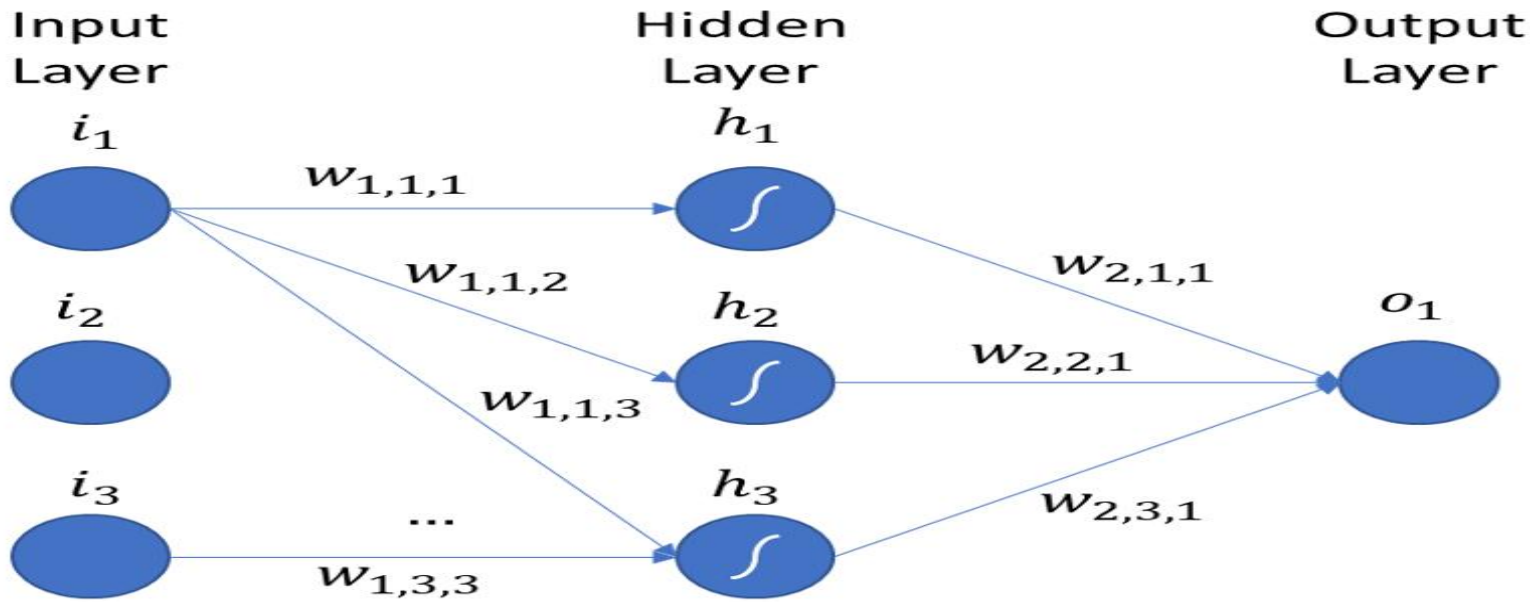
Important terminologies of ANNs...

Single Layer Perceptron Vs. Multilayer Perceptron



Single Layer Perceptron

Multi Layer Perceptron



Important terminologies of ANNs...

10. Learniing Algorithm (Backpropagation Algorithm):

The algorithm has mainly 2 phases.

- **Forward Pass Phase:** This phase computes the "functional signal" and does the feed forward propagation of input signals through network till the output neuron/node.
- **Backward Pass Phase:** This phase compute the "error signal" and propagate the error backwords through network starting from output units towards input node by updating the weight value such that the output of the perceptron for the updated weights will be closer to the target.

11. Epoch:

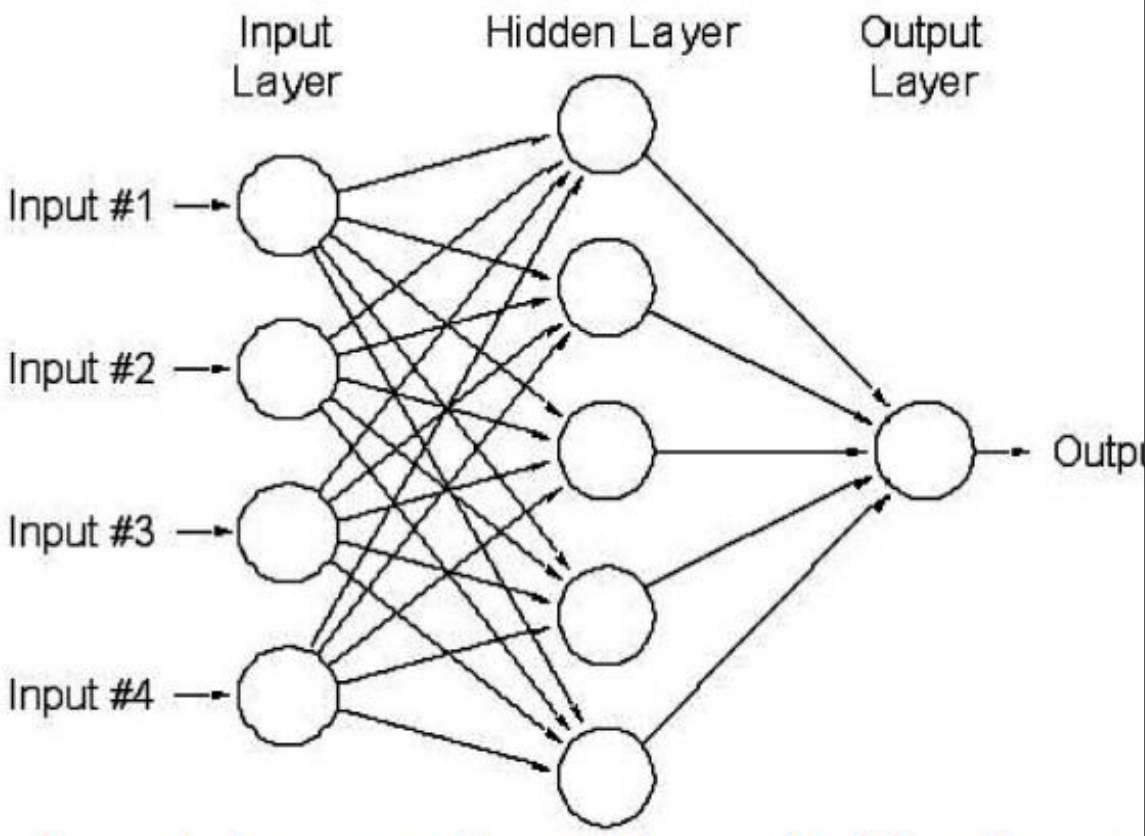
In the training phase one by one tuple is given as input to the MLP. The presentation of the entire training set to the neural network is called as one epoch.

For instance, for AND function an epoch consists of four sets of inputs being presented to the network (i.e. [0,0], [0,1], [1,0], [1,1])

Multilayer Neural Network
Backpropagation Method

Multi-Layer Neural Networks

- It is made up from an input, output and one or more hidden layers.
- Each node from input layer is connected to a node from hidden layer and every node from hidden layer is connected to a node in output layer.
- There is usually some weight associated with every connection. Input layer represents an the raw information that is fed into the network. This part of network is never changing its values.



- Every single input to the network is duplicated and send down to the nodes in hidden layer. Hidden Layer accepts data from the input layer.
- Output layer process information received from the hidden layer and produces an output. This output is than processed by activation function.
- The network is feed-forward in that none of the weights cycles back to an input unit or to an output unit of a previous layer

Number of Nodes and Layers

Number of Input Nodes:

Number of input nodes depends on training set in hand or number of nodes is equal to the number of features (columns) in training dataset. Some NN configurations add one additional node for a bias term.

Number of Output Nodes:

If the NN is a regressor, then the output layer has a single node. If the NN is a classifier, then it output nodes corresponds to per class label of the model.

Number of Hidden Layers:

The hidden layer configuration can be set by using just two rules:

(i) number of hidden layers equals one or two

(ii) number of neurons in that layer is the mean of the neurons in the input and output layers.

Backpropagation Algorithm

The training algorithm of backpropagation involves 4 steps

1. **Intialization of Weight:** Some small random weights (-1 to +1 or -0.5 to +0.5) are assigned to each connectors and some bias values are assigned to each computational unit (neurons).
2. **Feed Forward [Propagate the input forward]:**
   - The training tuple is fed to the network input unit layer (x_i) which propagate the received input (x_i) to the hidden layer as aj and then to the subsequent layers as follows.

Input from i^{th} unit to j^{th} unit $= I_j = \sum_i w_i x_i + \theta_i$

Where w_{ij} = Weight of connector i to j

x_i= Input from i^{th} unit

θ_j= Bias value of jth unit

   - Each j^{th} hidden unit [or output unit] compute the otput (Y_j) throuh the activation function (sigmoid) and propagate the output to the next unit [or outputs the response: class]

Output of j^{th} unit $= Y_j = f(I_j) = \frac{1}{1+e^{-Ij}}$

Backpropagation Algorithm ....

3. **Backpropagation of Errors:** Each unit (output/hidden) compares the computed output and targeted output to determine the **associated error** of that unit. The error is propagated backward by **updating the weights and biases** to rectify the error of the network prediction.

$$\text{Associated Error of } j^{th} \text{ unit}(Err_j) = \begin{cases} \text{if j is the output unit: } Err_j = Y_j(1-Y_j)(T_j-Y_j) \\ \text{if j is the hidden unit: } Err_j = Y_j(1-Y_j)\sum_k Err_k * w_{jk} \end{cases}$$

Where, T_j: Target output, Y_j : ComputedOutput, $Y_j(1-Y_j)$: Derivative of the sigmoid function
Err_k: Higher layer error, k: Higher layer index

Updating weight (W_{ij}): $W_{ij} = W_{ij} + \Delta W_{ij}$ Where $\Delta W_{ij} = \eta * Err_j * Y_i$

Updating Bias (θ_j): $\theta_j = \theta_j + \Delta\theta_j$ Where $\Delta\theta_j = \eta * Err_j$

Where, η: Given learning rate of network
Err_j: Error at j^{th} unit of higher layer
Y_i: Output from i^{th} unit of lower layer

4. **Iteratively updation of weights and bias value for one epoch.**

Advantage of Backpropagation

- Relatively simple implemenion.
- Mathematical formula used in the algorithm can be applied to any network.
- Computing time can be reduced if the weight choosen are small at the beginning.
- Well suited for continuous valued input and output
- High tolerant for noisy data.
- Ability to classify patterns for which they have not been trained

Drawback of Backpropagation

- Slow and inefficient. Can stuck in local minima resulting in sub-optimal solution.
- In case of large input, it is difficult to relate the output w.r.t. inputs.
- Require long training time.
- Poor interpretability: difficult to interpret the parameter values

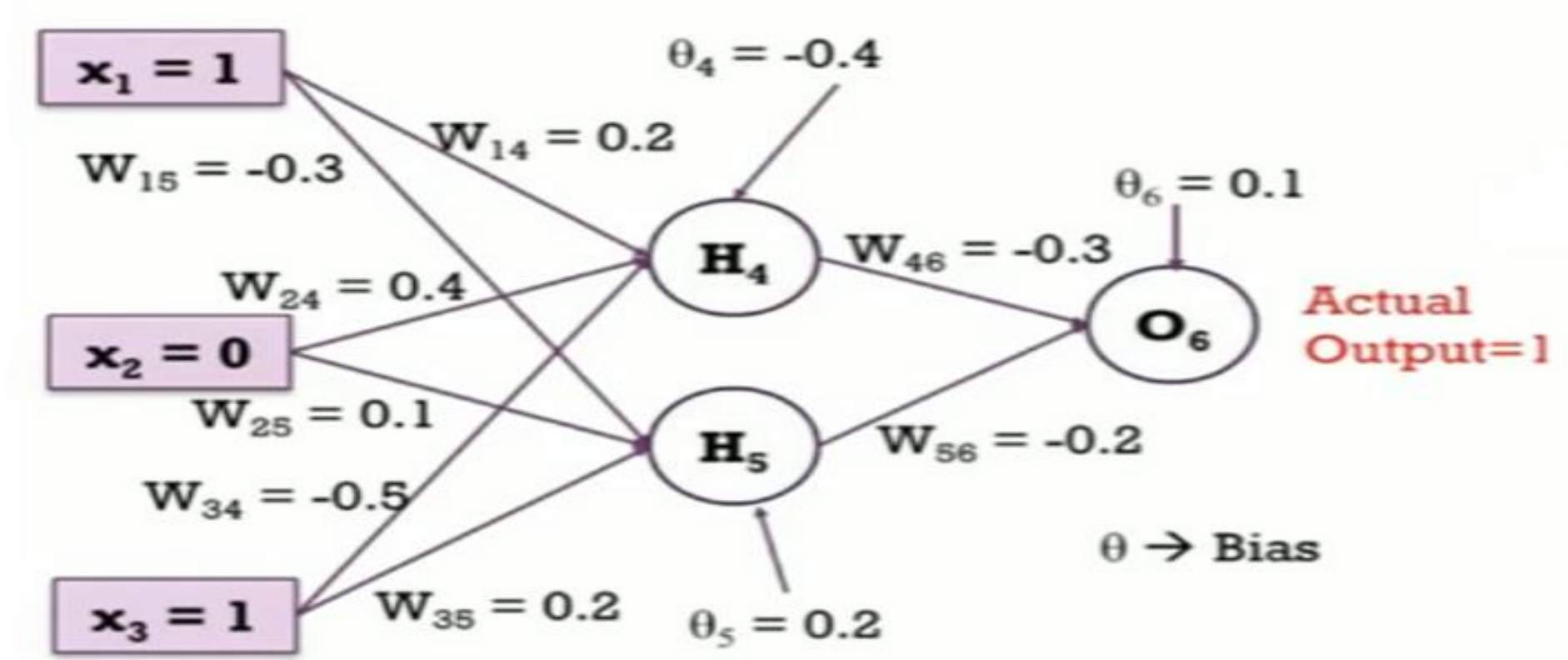
Application

- Successful in real world data: Handwritten character Recognition, pathology and laboratory medicine, tning a computer to pronounce.

Backpropagation Problem

Q: Assume that the neurons have a sigmoid activation function, perform a forward pass and backwd pass on the network to update the weights. Assume target output is 1 and learning rate is 0.9.

X1	X2	X3	W14	W15	W24	W25	W34	W35	W46	W56	$\theta 4$	$\theta 5$	$\theta 6$
1	0	1	0.2	-0.3	0.4	0.1	-0.5	0.2	-0.3	-0.2	-0.4	0.2	0.1



Backpropagation Problem...

Step 1 (Feed Forward) :- Compute Hidden Layer's & Output Layer's Input (Ij) and output (Yj)

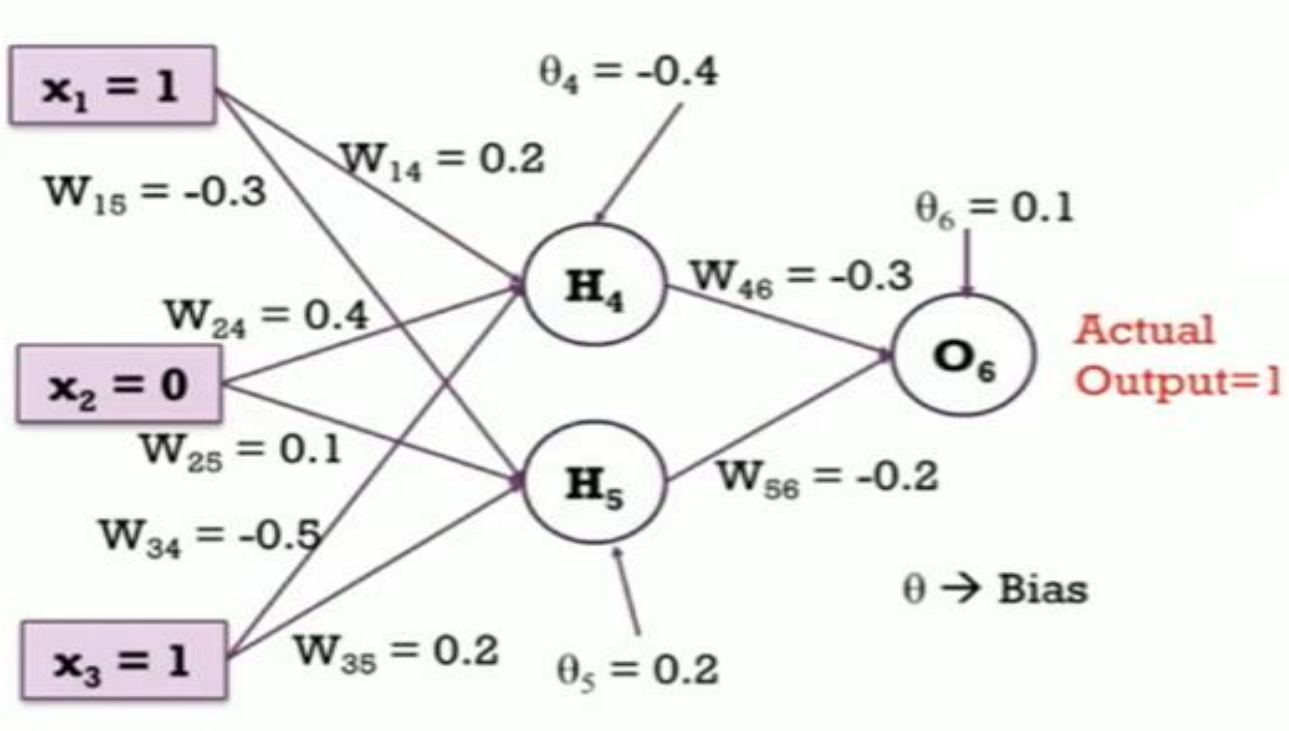
Input from i^{th} unit to j^{th} unit = $I_j = \sum_i w_i x_i + \theta_j$

$I_4 = (0.2*1 + 0.4*0 + -0.5*1) + -0.4 = -0.7$

$I_5 = (-0.3*1 + 0.1*0 + 0.2*1) + 0.2 = 0.1$

$I_6 = (W_{46}*Y_4 + W_{56}*Y_5) + \theta_6$

$= -0.3*0.332 + -0.2*.525 = -0.105$



Output of j^{th} unit = $Y_j = f(I_j) = \frac{1}{1+e^{-Ij}}$

$Y_4 = 1/1+e^{-0.7} = 0.332$

$Y_5 = 1/1+e^{-0.1} = 0.525$

$Y_6 = 1/1+e^{+0.105} = 0.474$

Backpropagation Problem...

Step 2 (Backpropagate) :- Update weight of each connector by evaluating the **corresponding error**

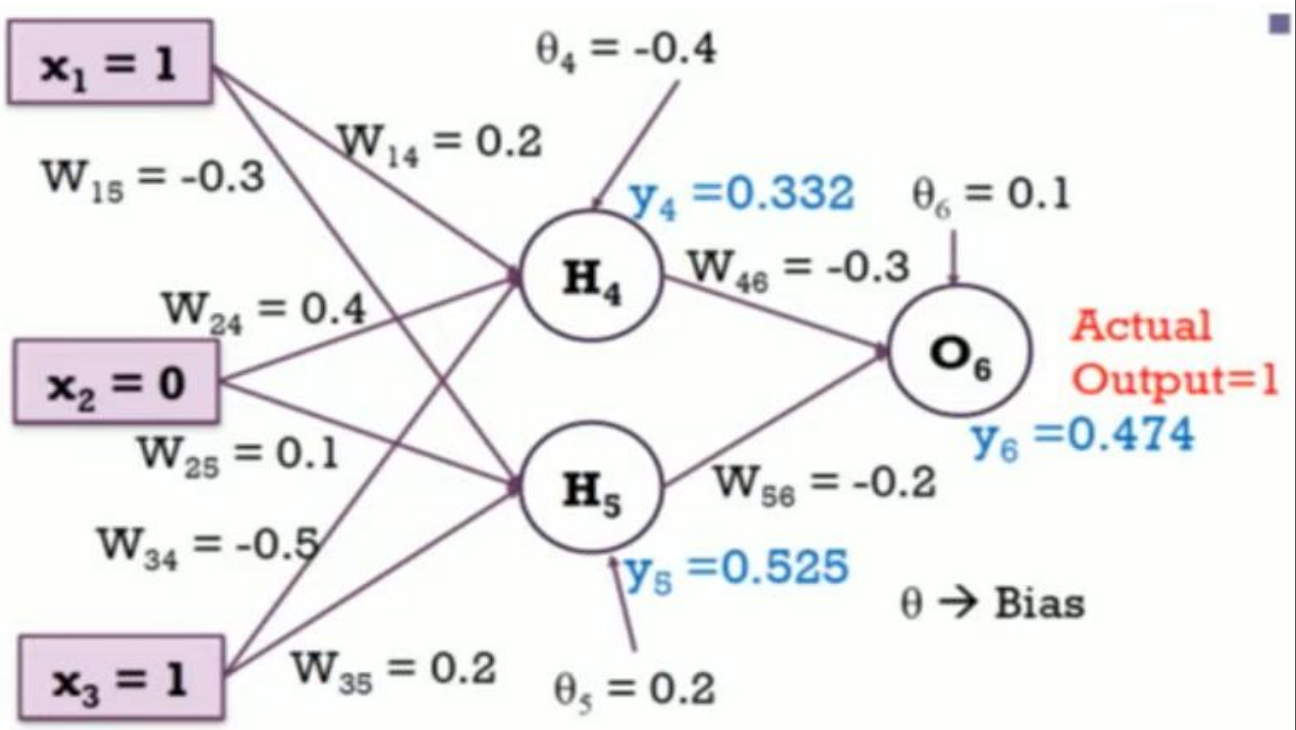
Updated Weight $W_{ij} = W_{ij} + \Delta W_{ij}$

Where $\Delta W_{ij} = \eta * Err_j * Y_i$

Update Weight W_{46} & W_{56} using Err_6

Update Weight W_{14} W_{24} & W_{34} using Err_4

Update Weight W_{15} W_{25} & W_{35} using Err_5



Backpropagation Problem...

Step 2 (Backpropagate) :-
(Compute Error)

$$\text{Associated Error of } j^{th} \text{ unit}(Err_j) = \begin{cases} \text{if j is the output unit: } Err_j = Y_j(1-Y_j)(T_j-Y_j) \\ \text{if j is the hidden unit: } Err_j = Y_j(1-Y_j)\sum_k Err_k * w_{jk} \end{cases}$$

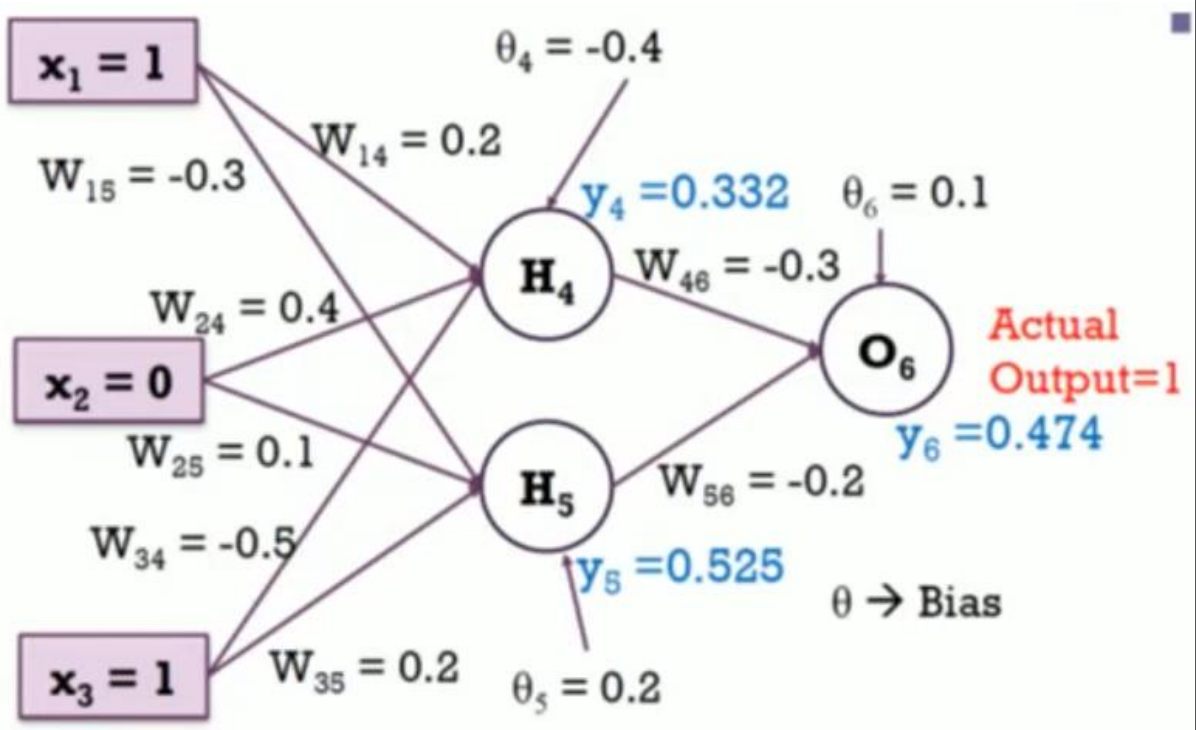
Output Layer Error:

$Err_6 = Y_6 * (1-Y_6) * (T_6-Y_6)$
$= 0.474 * (1-0.474) * (1-0.474) = 0.1311$

Hidden Layer Error:

$Err_4 = Y_4 * (1-Y_4) * W_{46} * Err_6$
$= 0.332 * (1-0.332) * -0.3 * 0.1311$
$= -0.0087$

$Err_5 = Y_5 * (1-Y_5) * W_{56} * Err_6$
$= 0.525 * (1-0.525) * -.2 * 0.1311$
$= -0.0065$



Backpropagation Problem...

Step 2 (Backpropagate) :-
(Update Weight and Biase)

Updated Weight $W_{ij} = W_{ij} + \Delta W_{ij}$

Where $\Delta W_{ij} = \eta * Err_j * Y_i$

Unit - 6

Update $W_{46} = W_{46} + \eta * Err_6 * Y_4$

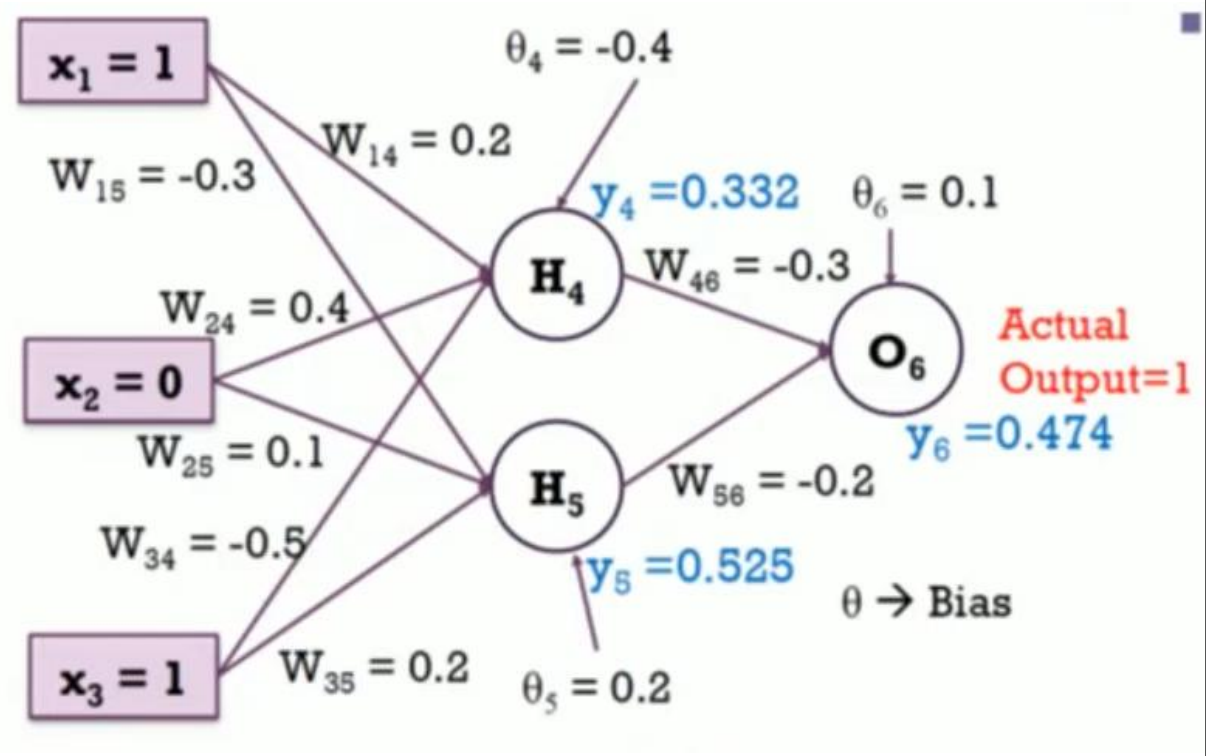
$= -0.3 + 0.9 * 0.1311 * 0.332$

$= -0.261$

Update $W_{56} = W_{56} + \eta * Err_6 * Y_5$

$= -0.2 + 0.9 * 0.1311 * 0.525$

$= -0.138$



Backpropagation Problem...

Step 2 (Backpropagate) :-
(Update Weight and Biase)

Unit - 4

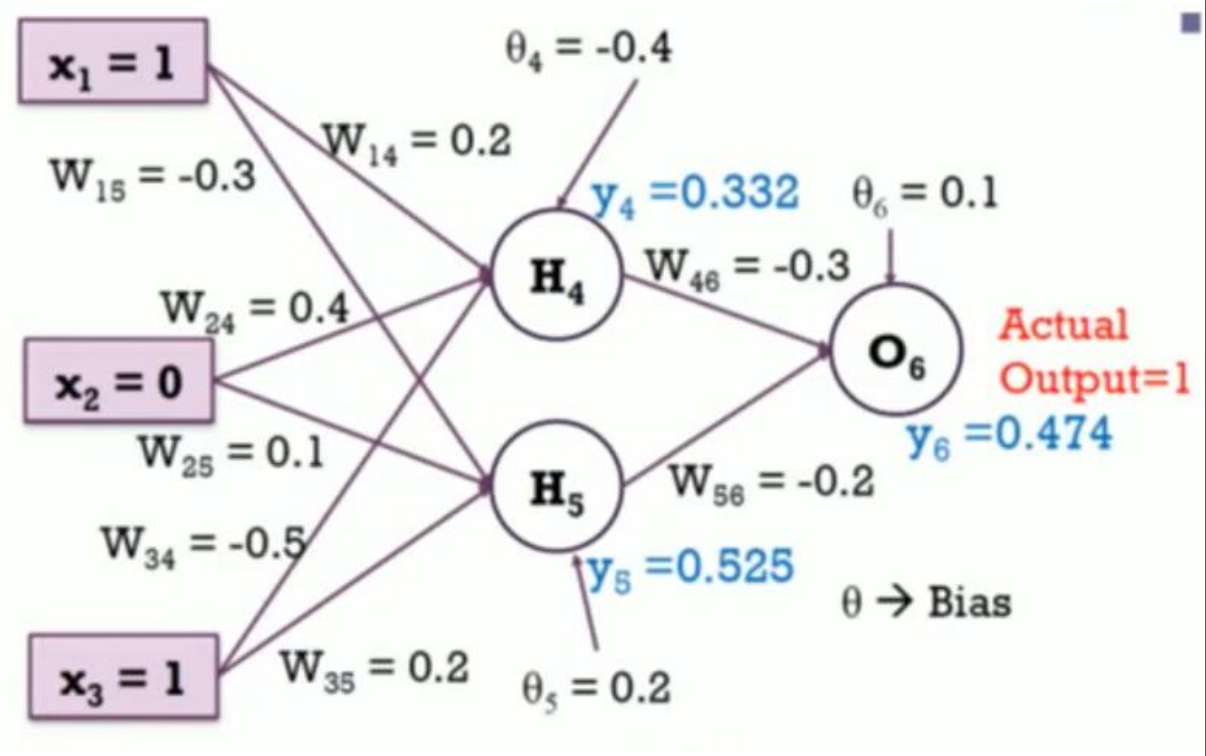
Updated Weight $W_{ij} = W_{ij} + \Delta W_{ij}$

Where $\Delta W_{ij} = \eta * Err_j * Y_i$

Update $W_{14} = W_{14} + \eta * Err_4 * Y_1$
$= -0.2 + 0.9 * -0.0087 * 1$
$= 0.192$

Update $W_{24} = W_{24} + \eta * Err_4 * Y_2$
$= 0.4 + 0.9 * -0.0087 * 0$
$= 0.4$

Update $W_{34} = W_{34} + \eta * Err_4 * Y_3$
$= -0.5 + 0.9 * -0.0087 * 1$
$= -0.508$



Backpropagation Problem...

Step 2 (Backpropagate) :-
(Update Weight and Biase)

Updated Weight $W_{ij} = W_{ij} + \Delta W_{ij}$

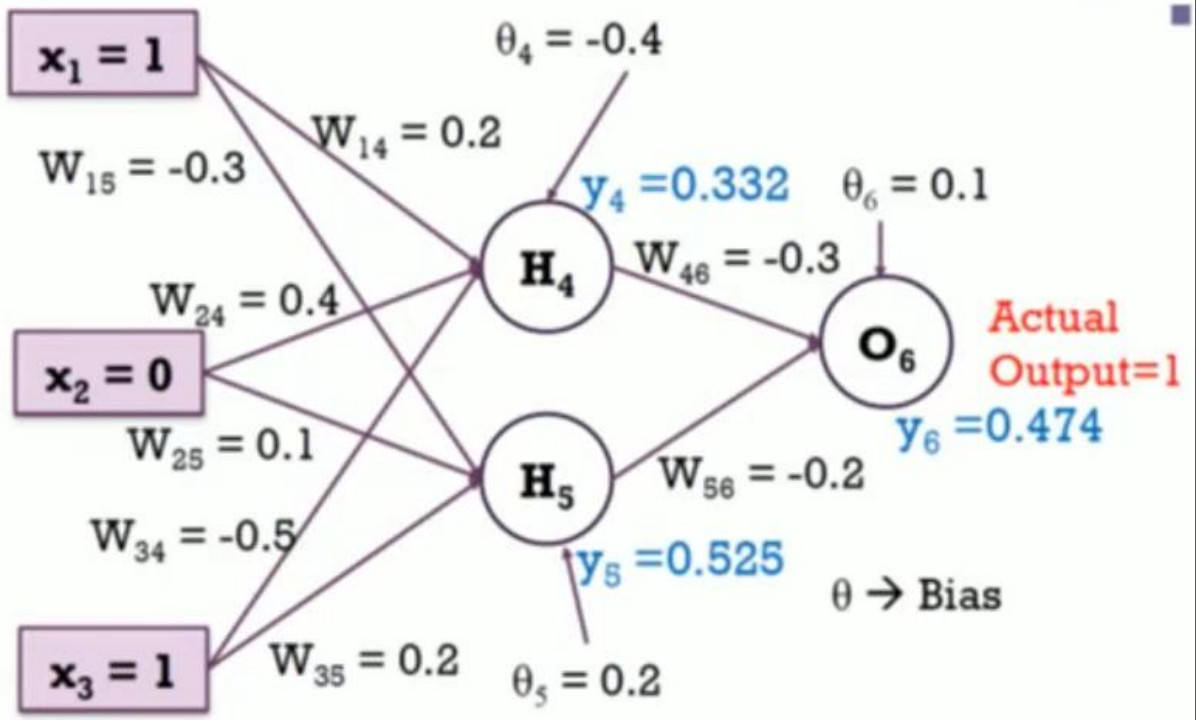
Where $\Delta W_{ij} = \eta * Err_j * Y_i$

Unit - 5

Update $W_{15} = W_{15} + \eta * Err_5 * Y_1$
$= -0.3 + 0.9 * -0.0065 * 1$
$= -0.306$

Update $W_{25} = W_{25} + \eta * Err_5 * Y_2$
$= 0.1 + 0.9 * -0.0065 * 0$
$= 0.1$

Update $W_{35} = W_{35} + \eta * Err_5 * Y_3$
$= 0.2 + 0.9 * -0.0065 * 1$
$= 0.194$



Backpropagation Problem...

Step 2 (Backpropagate) :-
(Update Weight and Biase)

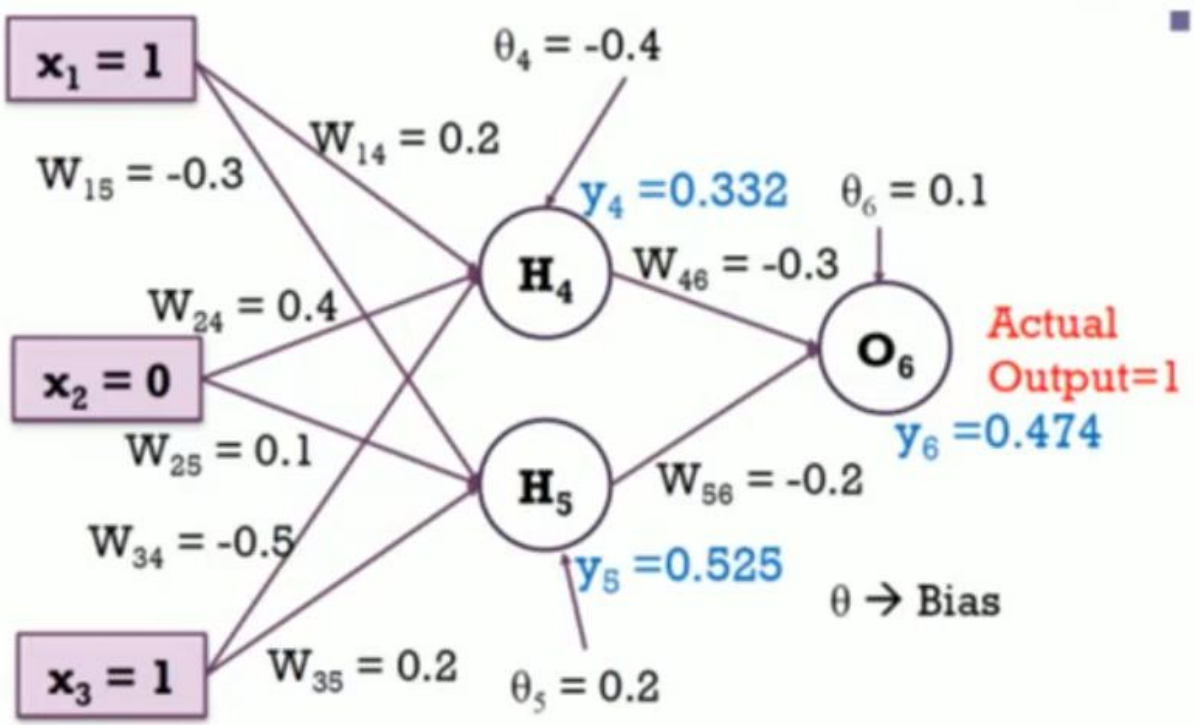
Updated Bias (θ_j): $\theta_j = \theta_j + \Delta\theta_j$
Where $\Delta\theta_j = \eta * Err_j$

New θ of Unit: 4, 5 & 6

Update $\theta_6 = \theta_6 + \eta * Err_6$
$= 0.1 + 0.9 * 0.1311$
$= 0.218$

Update $\theta_5 = \theta_5 + \eta * Err_5$
$= 0.2 + 0.9 * -0.0065$
$= 0.194$

Update $\theta_4 = \theta_4 + \eta * Err_4$
$= -0.4 + 0.9 * -0.0087$
$= -0.408$



Backpropagation Problem...

Parameter (Weight)	OLD	NEW
W14	0.2	0.192
W15	-0.3	-0.306
W24	0.4	0.4
W25	0.1	0.1
W34	-0.5	-0.508
W35	0.2	0.194
W46	-0.3	-0.261
W56	-0.2	-0.138

Parameter (Bias Value)	OLD	NEW
$\theta 4$	-0.4	-0.408
$\theta 5$	0.2	0.194
$\theta 6$	0.1	0.218

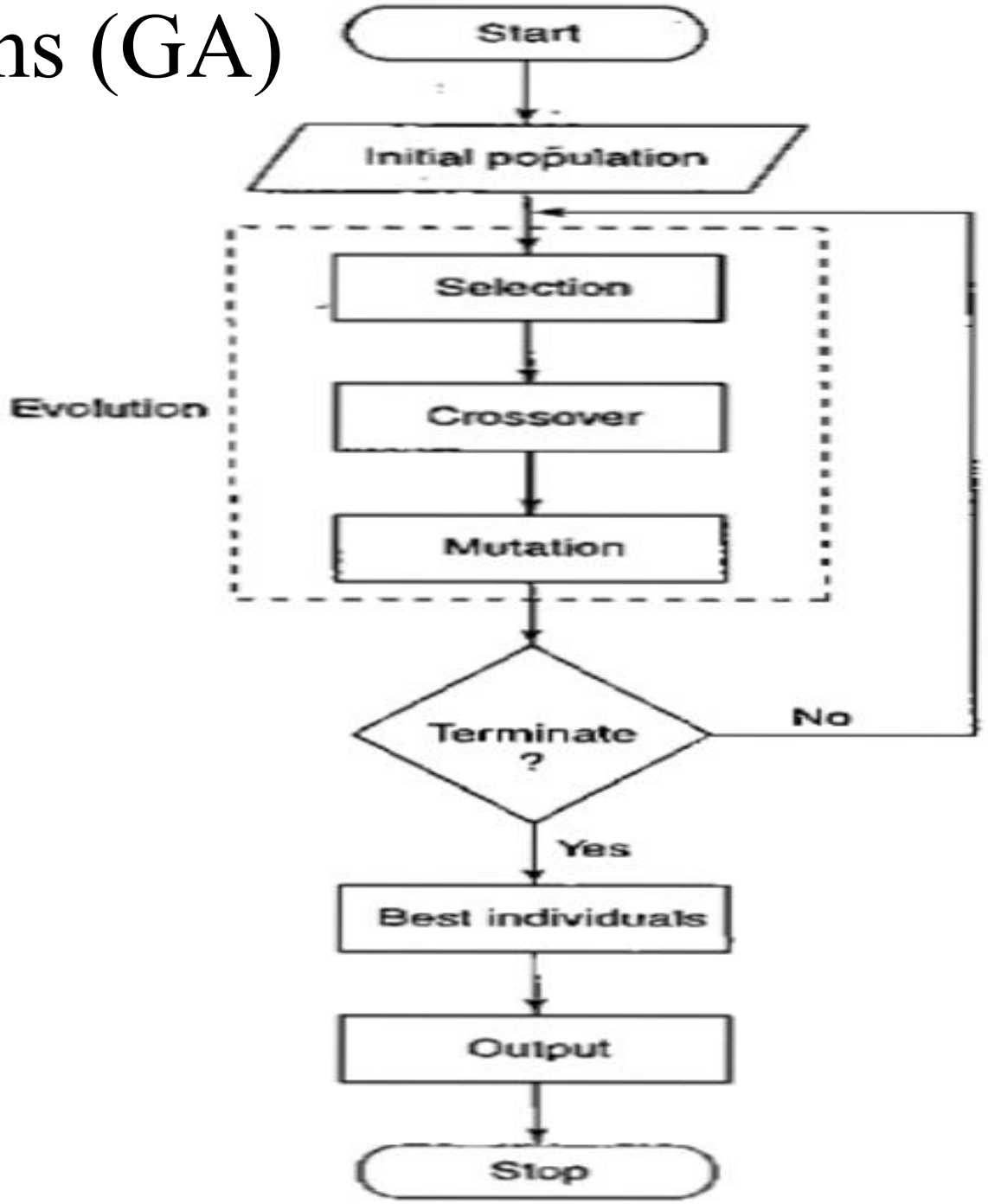
Iterate the same process of finding error and updating parameters till the error value is not acceptable

Other Classification Method

- Genetic Algorithm
- Rough Set Approach
- Fuzzy Set Approach

Genetic Algorithms (GA)

- Genetic Algorithm is based on an analogy to biological evolution
- This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation.
- Five phases are considered in a genetic algorithm.
  - Initial population
  - Fitness function
  - Selection
  - Crossover
  - Mutation



Genetic Algorithms (GA) in Classification

- **Initial Population:** An initial **population** is created consisting of randomly generated rules
  - Each rule is represented by a string of bits
  - E.g., if A_1 and $\neg A_2$ then C_2 can be encoded as 100
  - If an attribute has $k > 2$ values, k bits can be used
- **Fitness Function:** Based on the notion of survival of the **fittest**, a new population is formed to consist of the fittest rules and their offspring. The *fitness of a rule* is represented by its classification accuracy on a set of training examples
- **Selection, Mutation & Crossover:** Offspring are generated by *crossover* and *mutation* and then is *selected*/tested on the basis of fitness function.
- **Termination Condition:** The process continues until a population P evolves *when each rule in P satisfies a prespecified threshold*

Slow but easily parallelizable

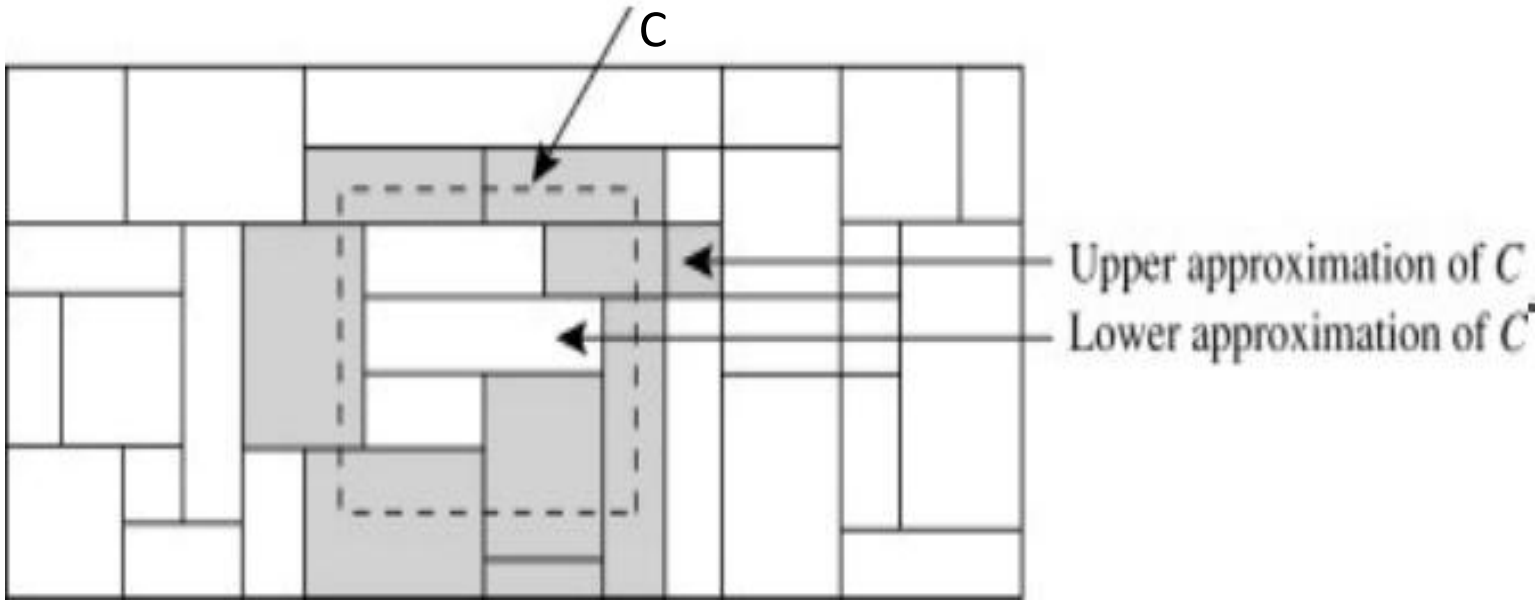
Rough Set Approach

- Rough set approach is used to discover structural relationship within imprecise and noisy data.
- **Note** − This approach can only be applied on discrete-valued attributes. Therefore, continuous-valued attributes must be discretized before its use.
- The Rough Set Theory is based on the establishment of equivalence classes within the given training data. The tuples that forms the equivalence class are indiscernible. It means the samples are identical with respect to the attributes describing the data.
- There are some classes in the given real world data, which cannot be distinguished in terms of available attributes. We can use the rough sets to roughly define such classes.

Rough Set Approach...

For a given class C, the rough set definition is approximated by two sets as:

- Lower Approximation of C
- Upper Approximation of C



- Lower Approximation of C − The lower approximation of C consists of all the data tuples, that based on the knowledge of the attribute, are certain to belong to class C.

- Upper Approximation of C − The upper approximation of C consists of all the tuples, that based on the knowledge of attributes, cannot be described as not belonging to C.

Fuzzy Set Approach

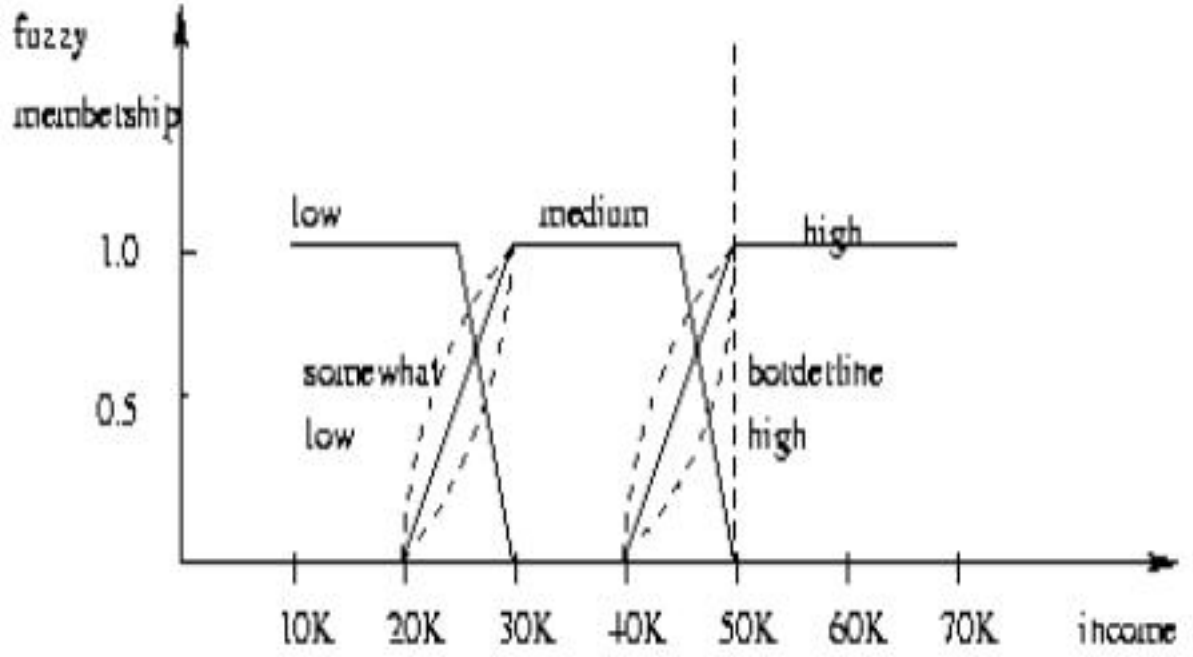
- Fuzzy Set Theory (Possibility Theory) was proposed by Lotfi Zadeh in 1965 as an alternative for two-value logic and probability theory. This theory allows to work at a high level of abstraction and provides the means for dealing with imprecise measurement of data.

- The fuzzy set theory also allows to deal with vague or inexact facts.

  For example, being a member of a set of high incomes is in exact (e.g. if $50,000 is high then what about $49,000 and $48,000). Unlike the traditional CRISP set where the element either belong to S or its complement but in fuzzy set theory the element can belong to more than one fuzzy set.

Fuzzy Set Approach...

- Fuzzy logic uses truth values between 0.0 and 1.0 to represent the degree of membership and can be represented through fuzzy membership graph



- Attribute values are converted to fuzzy values. Let income, *x*, is assigned a fuzzy membership value to each of the discrete categories {low, medium, high}

  e.g. $49K belongs to "medium income" with fuzzy value 0.15 but belongs to "high income" with fuzzy value 0.96

- Fuzzy membership values do not have to sum to 1.
- Each applicable rule contributes a vote for membership in the categories. Typically, the truth values for each predicted category are summed, and these sums are combined

Performance Measure of Classifier

Metrics for Evaluating Classifier Performance

- Performance measure of classifier presnts how good or how"accurate" the classifier is at predicting the class label of tuples.
- In classification problem, performance evaluation can be done through the use of some metrics. The use of of metrics depends on the type of classifier problem. The overall metrics used for performance measure:
  - Accuracy and Error rate/Misclassification rate } [Used when class tuples/classes are evenly distributed]
  - Sensitivity and Specificity
  - Precision and Recall
  - F/F1 Score and F_B Score

  } [Imbalance Class/ One class may be *rare*] e.g fraud, or HIV-positive
- A model is over optimistic if the accuracy evaluation happens on training data. => It is better to measure the classification accuracy on test set which were not used in the traing phase.

Metrics for Evaluating Classifier Performance...

There are some "building blocks" used in computing metrics of performance measure.

[Assumption binary class with class value positive or negative]

- True Positive (TP): Number of positive tuples that are correctly classified.
- True Negative(TN): Number of negative tuples that are correctly classified.
- False Positive(FP): Number of negative tuples that are incorrectly classified.
- False Negative(FN): Number of positive tuples that are incorrectly classified.

A confusion matrix is a tool that summerizes the behavior of the classifier by presenting TP & TN [No. of correct classification] and FP & FN [No. of incorrect classification]

ACTUAL CLASS \ PREDICTED CLASS	Positive	Negative	TOTAL
Positive	**TP**	**FN**	**P**
Negative	**FP**	**TN**	**N**
TOTAL	**P'**	**N'**	

CM_{ij}:- number of tuples of class i that are labeled by the classifier as class j

Metrics for Evaluating Classifier Performance...

RID	Age	Income	Student	Credit_Rating	Actual Class: Buys Computer	Predicted Class: Buys Computer
1	Youth	High	No	Fair	No	Yes
2	Youth	High	No	Excellent	No	Yes
3	middle_aged	High	No	Fair	Yes	No
4	senior	Medium	No	Fair	Yes	Yes
5	senior	Low	Yes	Fair	Yes	No
6	senior	Low	Yes	Excellent	No	Yes
7	middle_aged	Low	Yes	Excellent	Yes	Yes
8	Youth	Medium	No	Fair	No	Yes
9	Youth	Low	Yes	Fair	Yes	No
10	senior	Medium	Yes	Fair	Yes	No
11	Youth	Medium	Yes	Excellent	Yes	Yes
12	middle_aged	Medium	No	Excellent	Yes	No
13	middle_aged	High	Yes	Fair	Yes	No
14	senior	Medium	No	Excellent	No	Yes

Metrics for Evaluating Classifier Performance

Confusion matrix of the given dataset is:

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	**3**	**6**	9
buy_computer = no	**5**	**0**	5
Total	8	6	14

A classifier is called as **ideal classifier** if in its confusion matrix the diagonal entries contain most/all tuples (number) and rest entries contain less/zero tuples (number).

Metrics for Evaluating Classifier Performance

For Balanced Class Problem:

1. **Accuracy** or **recognition rate:** Accuracy simply measures how often the classifier correctly classify. It is the ratio or percentage of test set tuples that are correctly classified by the classifier.

[Example: Out of all the patients who visited the doctor, how many were correctly diagnosed as Covid positive and Covid negative.]

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

*** Missclassification rate/ Error Rate: 1-accuracy(M)**

$$\textbf{Error Rate} = 1 - \frac{TP+TN}{TP+TN+FP+FN} = \frac{FP+FN}{TP+TN+FP+FN}$$

Note: If the error rate is computed on training data then this is known as resubstitution error

As per the example [confusion matrix] : $\text{Acuracy} = \frac{3+0}{3+5+6+0} = \frac{3}{14} = 21\%$

$$\text{Error Rate} = \frac{5+6}{3+5+5+0} = \frac{11}{14} = 78\%$$

Metrics for Evaluating Classifier Performance...

For Imbalanced Class Problem: [Fraud, cancer]

In class imbalance problem, the goal is not the what is the accuracy of teh model. Rather, it is required to know how well the classifier classifies the positive tuple (cancer/covid= "no": FN cost is more) and how well the classifier classifies the negative tuple (Spam = "yes": FP cost is less).

2. **a. Sensitivity [True Positive rate]:** The propotion of positve tuples that are correctly identified. Or, Out of all the actual real positive cases, how many were identified as positive.

$$Sensitivity = \frac{TP}{P} = \frac{TP}{TP + FN}$$

[Out of all the actual real positive cases, how many were identified as positive]

b. Specificity [True Negative rate]: The propotion of negative tuples that are correctly identified.

$$Specificity = \frac{TN}{N} = \frac{TN}{FP + TN}$$

[Out of all the actual negative cases, how many were identified as negative.]

Metrics for Evaluating Classifier Performance...

Accuracy cbe a function of sensitivity and specificity:

$$Accuracy = \text{sensitivity}\frac{P}{T+P} + specificity\frac{N}{T\ +\ N}$$

[Out of all the actual real positive cases, how many were identified as positive]

Example: Confusion matrix for medical data (Cancer)

Actual class	Predicted		
	yes	no	Total
yes	**90**	**210**	300
no	**140**	**9560**	9700
Total	230	9770	10,000

$$Sensitivity = \frac{TP}{TP\ +\ FN} = \frac{90}{300} = 30\%$$

$$Specificity = \frac{TN}{TN\ +\ FP} = \frac{9560}{9700} = 98.56\%$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{90+9560}{10000} = 96.50\%$$

Althogh the classifier has a high accuracy, it's ability to correctly label the positive (rare) class is poor [its low sensitivity]. It has high specificity, means accurately recognize negative tuples.

Metrics for Evaluating Classifier Performance...

For Imbalanced Class Problem: .....

3. **a. Precision[Meausure of exactness]:** The percentage of tuples that are classified as positive are actually positive.

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

[Out of all that were marked as covid positive, how many are actually truly positive.]

Prcision=1: TP/(TP+FP) =1 ⇒ FP =0: None of the negative tuples are wrongly identified

b. Recall [Measure of Completeness]: The propotion of positive tuples that are correctly identified. **Sensitivity is same as recall.**

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

Recall = 1: TP/(TP+FN) =1 ⇒ FN =0: None of the positive tuples are wrongly identified

[Out of all the actual real positive cases, how many were identified as positive]

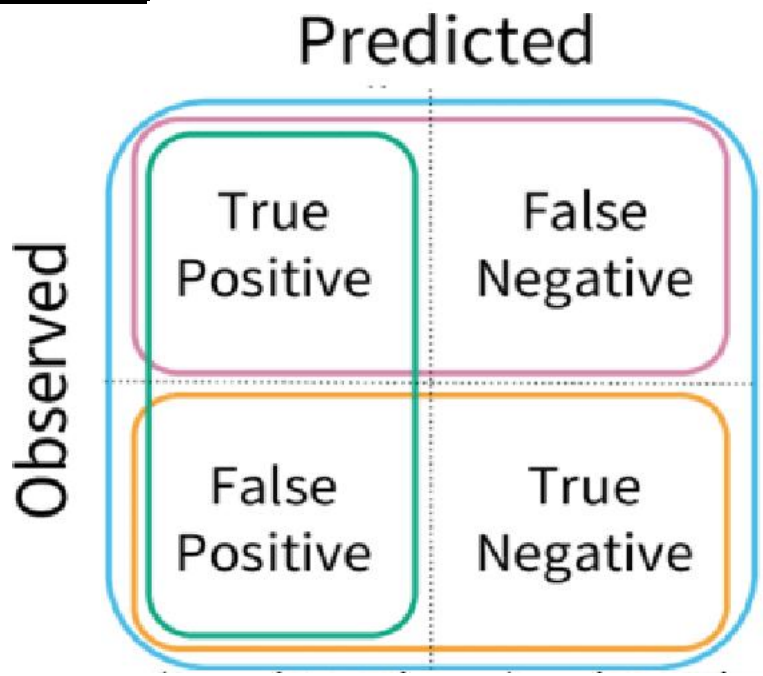
Metrics for Evaluating Classifier Performance...

Example: Confusion matrix for medical data (Cancer)

Actual class	Predicted		
	yes	no	Total
yes	**90**	**210**	300
no	**140**	**9560**	9700
Total	230	9770	10,000

$$\text{Prcision} = \frac{TP}{TP + FP} = \frac{90}{230} = 39.13\%$$

$$Recall = \frac{TP}{TP + FN} = \frac{90}{300} = 30\%$$



Accuracy = $\frac{TP + TN}{TP + TN + FP + FN}$

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

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

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

Metrics for Evaluating Classifier Performance...

For Imbalanced Class Problem: .....

An alternative way to useprecision and recall is to combine them into a single measure

4. **a. F1-Score or F-Score:** Sometimes it is required to give weightage to FP and sometimes to FN. F1 score is a **weighted average of Precision and Recall**, which means there is equal importance given to FP and FN. This is a very useful metric compared to "Accuracy"..

$$F1Score = \frac{2 * precision * Recall}{Prcision + Recall}$$

[Out of all that were marked as covid positive, how many are actually truly positive.]

b. F_β-Score: It is a weighted measure of precision and recall. It assigns β time as much weight to recall as precision.

$$F_\beta Score = \frac{(1+\beta)^2 * precision * Recall}{\beta^2 * Prcision + Recall}$$