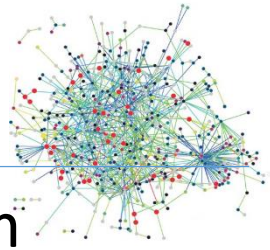
Chapter 3 - Classification (12 hours)

Er. Jeewan Rai

Basics and Algorithms
Decision Tree Classifier [human oriented]
Rule Based Classifier
Nearest Neighbor Classifier
Bayesian Classifier
Artificial Neural Network Classifier
Issues: Overfitting, Validation, Model Comparison

Terminology



- **Learning** – In this process the data are analyzed by classification algorithm
- **Classification**, which is the task of assigning objects to one of the several predefined categories, is a universal problem that encompasses many diverse applications.
- **Classification** is a data mining technique that assigns categories to a collection of data in order to aide in more accurate predictions and analysis.

Examples:

- Detecting spam email messages based upon the message header and content
- Categorizing cells as bad or good based upon the results of MRI scans
- Classifying galaxies based upon their shapes
- Classification high-risk or low-risk patients based on conditions.

- **Training set** used to build the model
- **Test set** used to determine the accuracy of the model

Classification Process

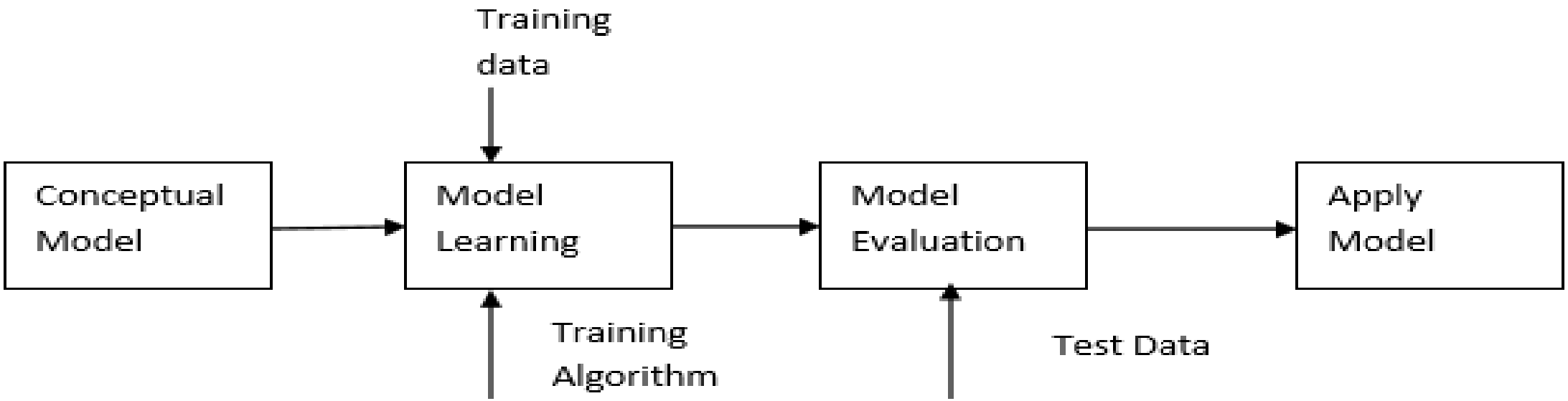
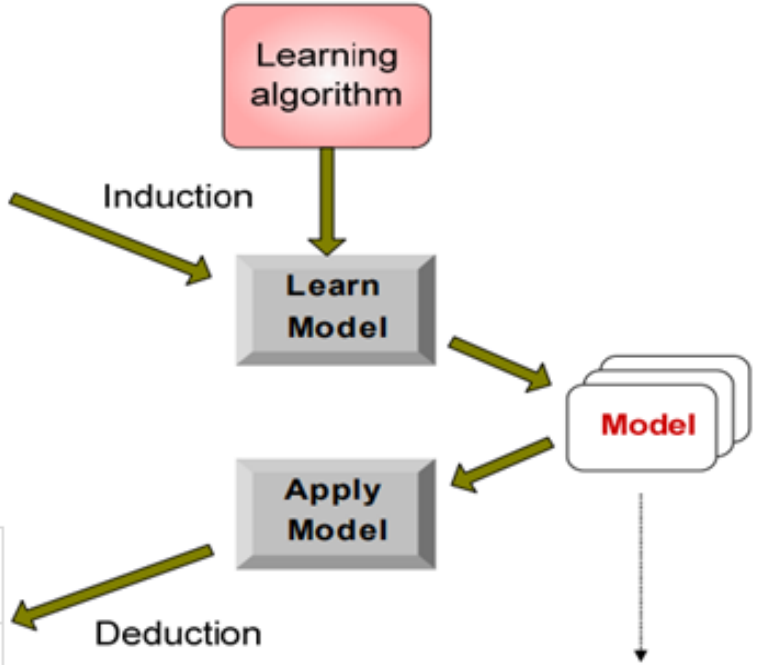


Fig. Stages in Classification

HOW CLASSIFICATION WORKS ?

Training Data / Training Set

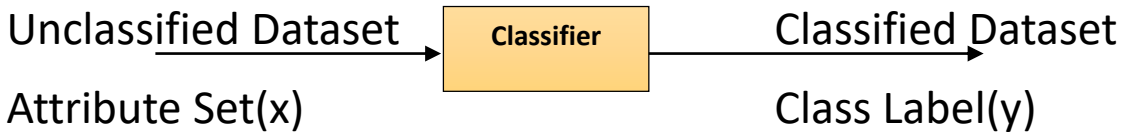
Name	Age	Gender	Annual Income	Credit Card Offer
John Doe	25	M	$39,500	No
Jane Doe	56	F	$125,000	Yes



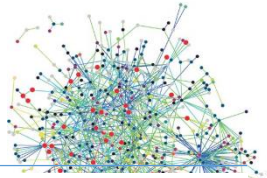
Predictor Data / Test Set

Name	Age	Gender	Annual Income	Credit Card Offer
Jack Frost	42	M	$88,000	
Mary Murray	16	F	$0	

IF (Age>18 OR Age <75) AND Annual Income >40,000 THEN Credit Card Offer=yes

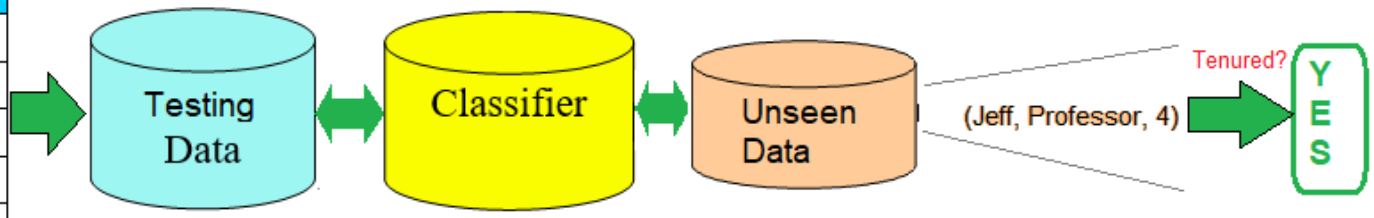


Classification—A Two-Step Process

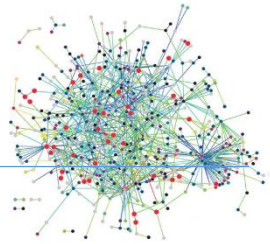


- **Model construction**: describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute. The set of tuples used for model construction is training set
  - The model is represented as classification rules, decision trees, or mathematical formulae
- **Model usage:** for classifying future or unknown objects
  - 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 samples that are correctly classified by the model
    - Test set is independent of training set, otherwise over-fitting will occur
  - If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known

NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no



Uses of Classification Model



- **Descriptive Modeling:** It serves as an explanatory tool to distinguish between objects of different classes.
- **Predictive Modeling:** used to predict the class label of unknown records.

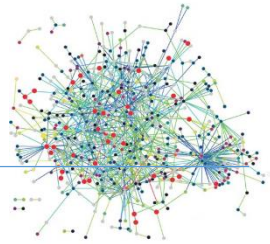
Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Class Label
Human	Warm-blooded	Hair	Yes	No	No	Yes	Mammal
Python	Cold-blooded	Scales	No	No	No	No	Reptile
Whale	Warm-blooded	Scales	Yes	Yes	No	No	Mammal

Descriptive Modeling

Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Class Label
Frog	Cold-blooded	None	No	Semi	No	Yes	? *(amphibian)*

Predictive Modeling:

Types of Classifier:

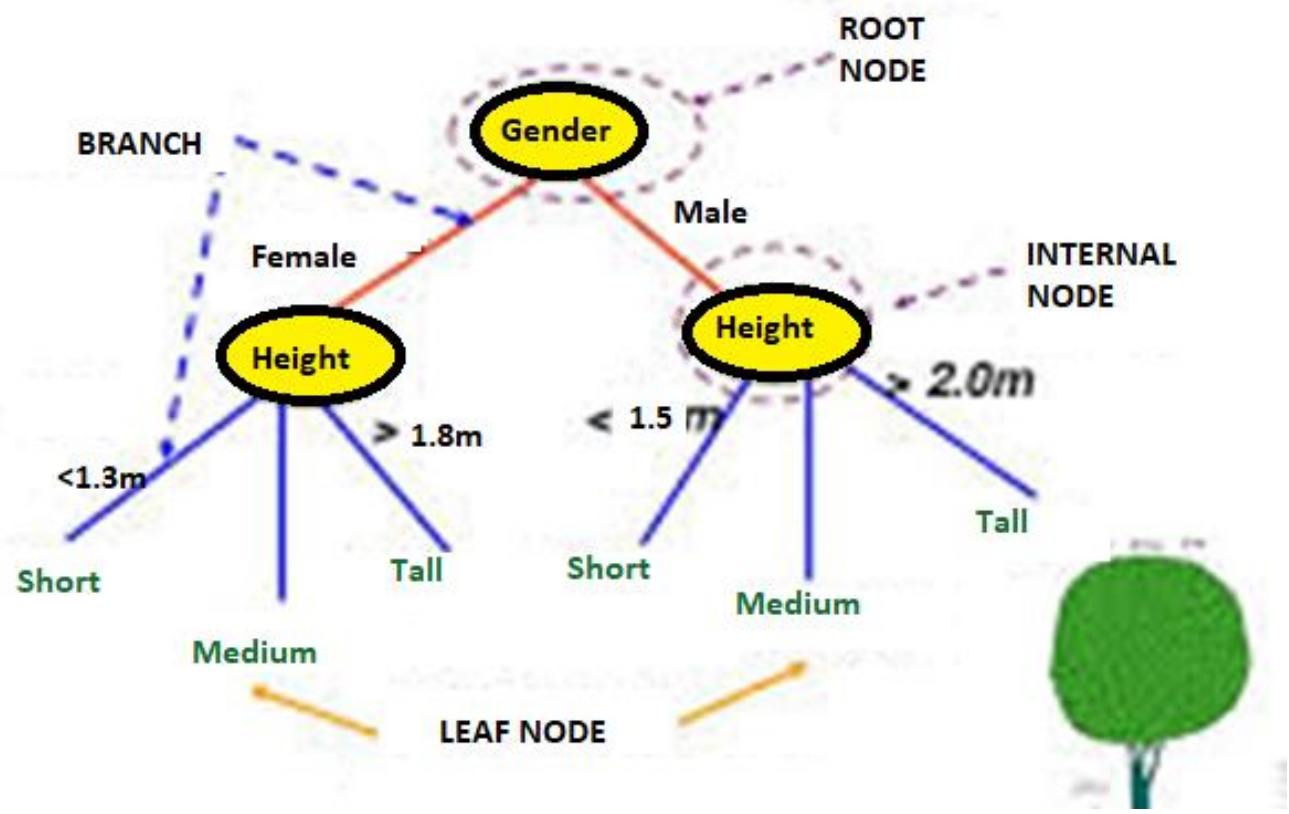


- Decision Tree classifier
- Rule Based Classifier
- Nearest Neighbor Classifier
- Bayesian Classifier
- Artificial Neural Network (ANN) Classifier

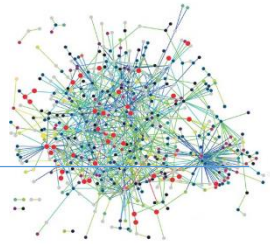
Decision Tree Classifier – *human oriented*

A **structure** that includes where each

- **Internal node** (non-leaf node) denotes a test on an attribute,
- each **Branch** represents an outcome of the test, and
- each **Leaf node** (or terminal node) holds a class label

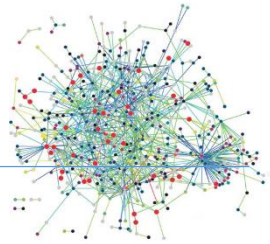


Algorithm for Decision Tree Induction



- **Basic algorithm**
  - Tree is constructed in a top-down recursive divide-and-conquer manner
  - At start, all the training examples are at the root
  - Attributes are categorical (if continuous-valued, they are discredited in advance)
  - Examples are partitioned recursively based on selected attributes
  - Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)
- **Conditions for stopping partitioning**
  - 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

Phases of Tree Generation

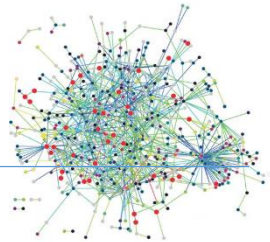


- **Tree Construction**
  - At start all the training examples are at the root
  - Partition examples recursively based on selected attributes
- **Tree Pruning**
  - Identify and remove branches that reflect noise or outliers
  - Once the tree is build
  - Use of decision tree: Classifying an unknown sample

Tree Pruning Approaches

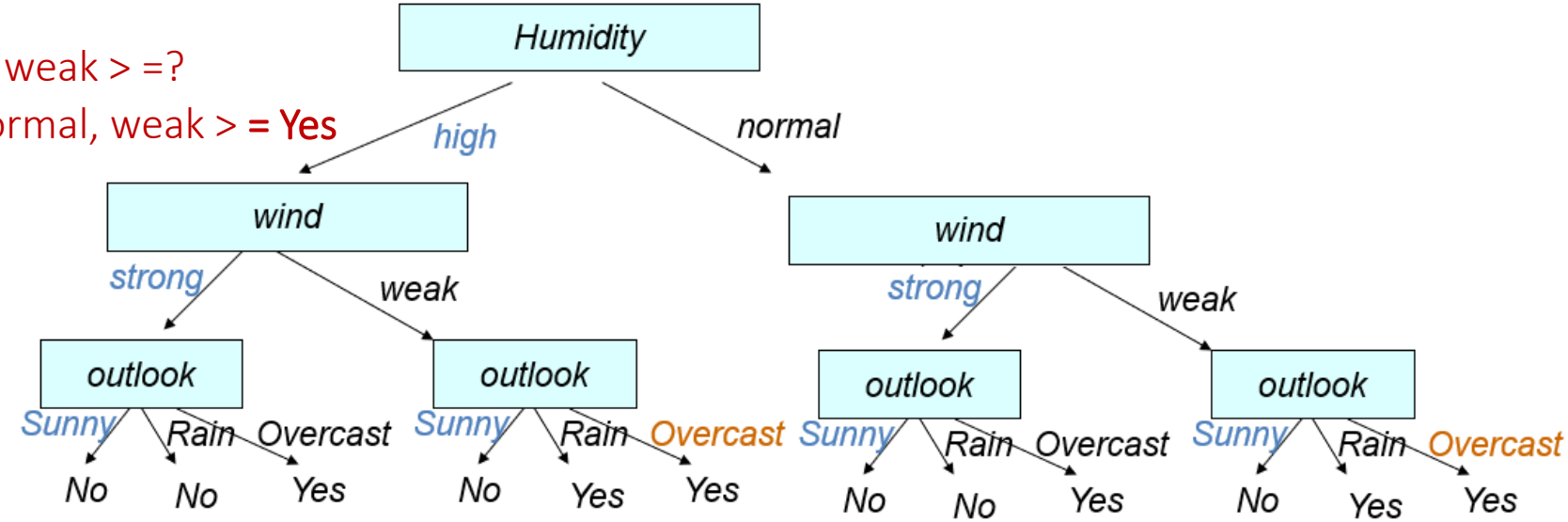
- **Pre-pruning** – The tree is pruned by halting its construction early.
- **Post-pruning** - This approach removes a sub-tree from a fully grown tree.

Decision Trees and Logic - for Play Tennis

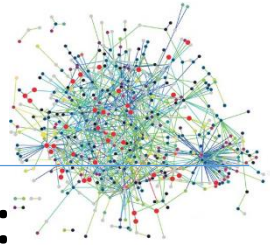


Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	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

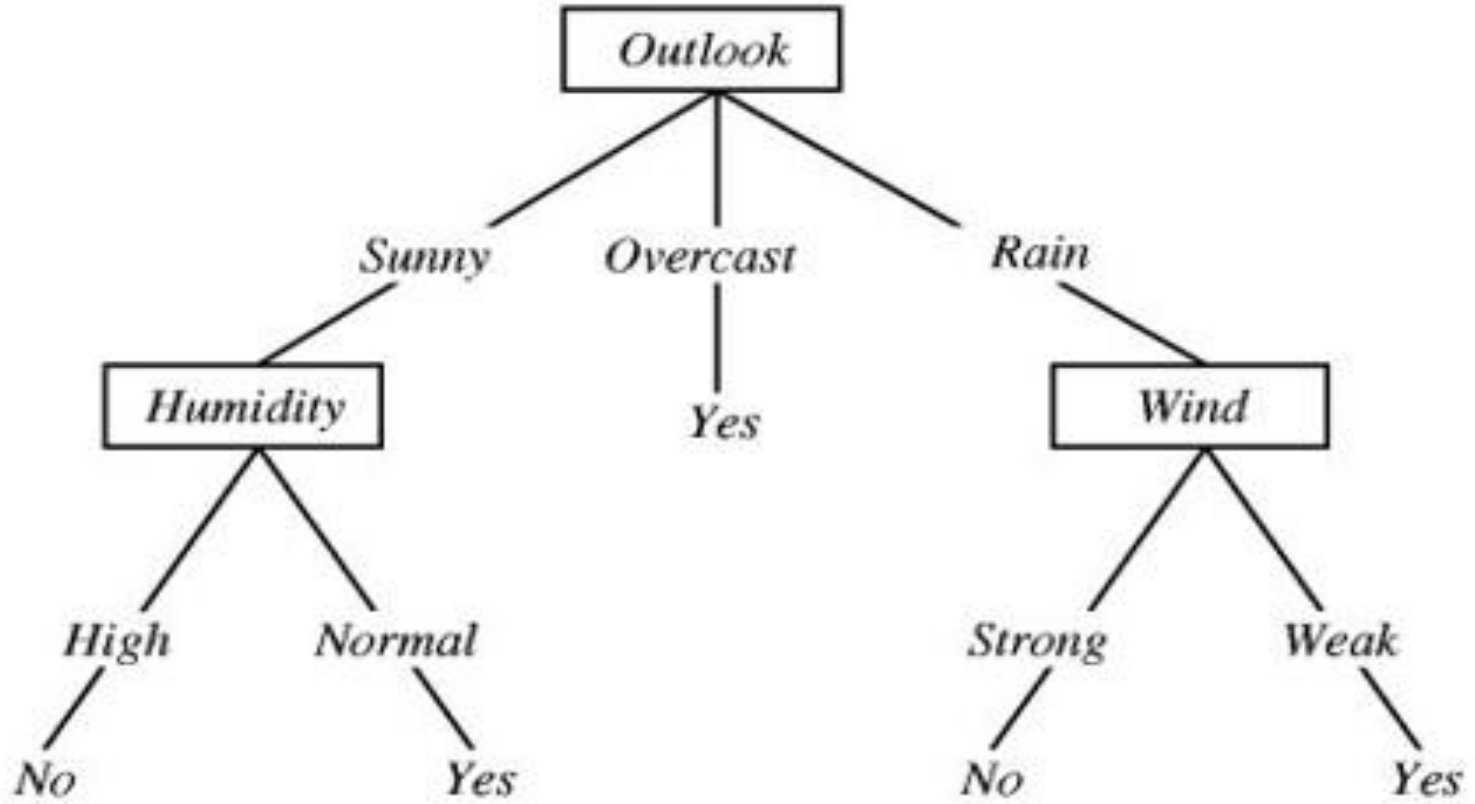
- **Classifying:** < sunny, hot, normal, weak > =?
- **Classification for:** < sunny, hot, normal, weak > **= Yes**



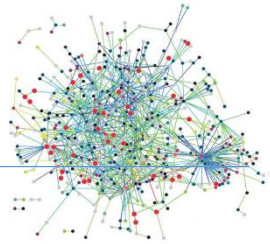
A Big Problem…



- Tree from the same training data that has a different attribute order:
- **Which attribute should we choose for each branch?**

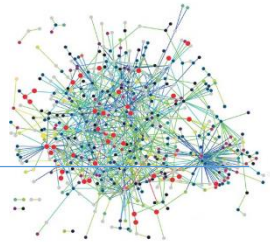


Decision Tree Algorithm



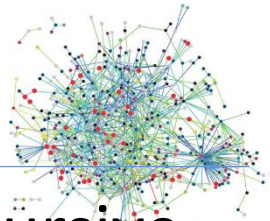
1. Hunt's Algorithm (Based on recursive fashion)
2. ID3, J48, C4.5 (Based on Entropy Calculation)
3. SLIQ, SPRINT, CART (Based on Gini-Index)

Decision Tree Algorithm – Hunt's



- Based on **recursive fashion** by partitioning the training data
- Each combination of attribute sets has a **unique class label**
- **Steps** that are done until the tree is fully grown.
  - Examine the record data and find the best attribute for the first node.
  - Split the record data based on this attribute
  - Recurse on each corresponding child node choosing other attributes

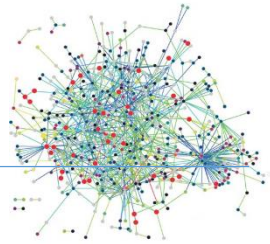
Decision Tree Algorithm – Hunt's



Let $\mathbf{D_t}$ be the set of training data that reach a node '**t**'. The general recursive procedure is defined as:

- If $\mathbf{D_t}$ contains records that belong the **single class** $\mathbf{y_t}$, then **t** is a leaf node labelled as $\mathbf{y_t}$.
- If $\mathbf{D_t}$ is an **empty set**, then **t** is a leaf node labelled by the default class, $\mathbf{y_d}$
- If $\mathbf{D_t}$ contains records that belong to **more than one class**, use an attribute test to split the data into smaller subsets.
- It recursively applies the procedure to each subset **until all the records in the subset belong to the same class.**

Tree Induction

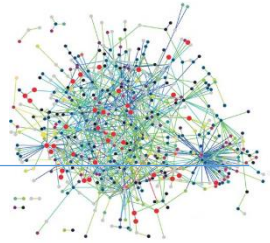


- learning of decision trees from class-labeled training tuples.

Design Issues of Decision Tree Induction:

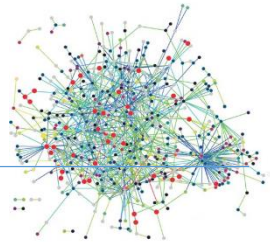
- **How to split the record?**
- **How to specify the attribute test condition?**
  - Depends on attribute types and number of ways to split the record i.e. 2-ways (Binary) split or multiway split.
  - Depends upon attribute types. (Nominal, Ordinal, Continuous)

Tree Induction: …Design Issues



- **When to stop splitting?**
  - When all records are belonging to the same class or all records have similar attributes.
  - At what point do you decide to stop the tree-growing process? This is another crucial area that must not be overlooked. One might assume that all we need to is to allow the recursive steps play out all the way through to the end.
- **How to determine the best split?**
  - Nodes with homogenous class distribution are preferred.
  - Measure the node impurity.
    - Gini-Index
    - Entropy
    - Misclassification Error

Gini-Index



- The Gini Index measures the impurity of data set (D) as: -

$$\text{Gini(D)} = 1 - \sum_1^n p_i{}^2$$

Where, n = Number of classes, pi = Probability of i^{th} class.

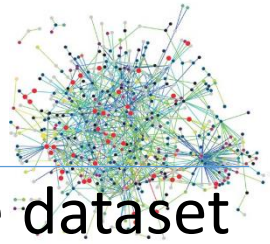
- It considers binary split for each attribute.
- When D is partition into D_1 and D_2 then

GSplit(D) or Gini(D) = D_1/D Gini(D_1) + D_2/D Gini(D_2)

ΔG = GStart - GSplit(D)

- The attribute that maximize the reduction (ΔG) in impurity is selected as splitting attribute. i.e. minimum GSplit

Entropy



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

$$H(S) = \text{-p1 log p1 - p2 log p2 ... - -pn log pn } = -\sum_{x \in X} p(x) \log_2 p(x)$$

Where,

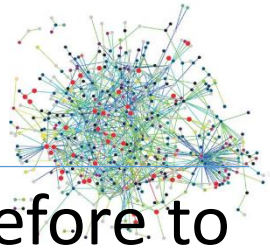
- S The current dataset for which entropy is being calculated (changes every iteration of the ID3 algorithm)

- X - Set of classes in

- P(x) - The probability of each set S

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

Information Gain



- Information gain is the measure of the difference in entropy from before to after the set S is split on an attribute A. In other words, how much uncertainty in dataset (S) was reduced after splitting dataset S on attribute A.

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

Where,

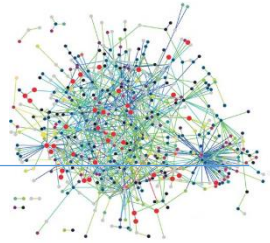
H(S) - Entropy of dataset S

T - The subsets created from splitting dataset S by attribute A.

P(t) - The probability of class t

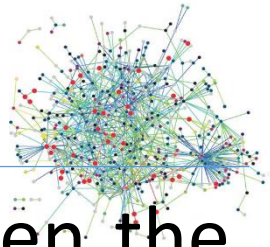
H(t) - Entropy of subset t

Decision Tree Algorithm: ID3



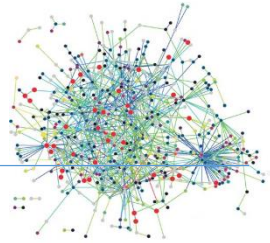
- The ID3 algorithm begins with the original dataset as the root node.
- On each iteration of the algorithm, it iterates through every unused attribute of the dataset and calculates the entropy (or information gain) of that attribute.
- It then selects the attribute which has the smallest entropy (or largest information gain) value.
- The dataset is then split by the selected attribute to produce subsets of the data.
- The algorithm continues to recur on each subset, considering only attributes never selected before.

ID3 Algorithm



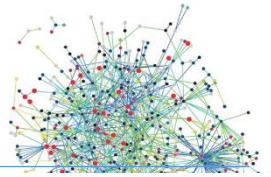
- Every element in the subset belongs to the same class, then the node is turned into a leaf and labelled with the class of the examples
- If the examples do not belong to the same class,
  - Calculate entropy and hence information gain to select the best node smallest entropy (or largest information gain) value to split data.
  - Partition the data into subset.
- Recursively repeat until all data are correctly classified

Advantages of Decision Tree Classifier



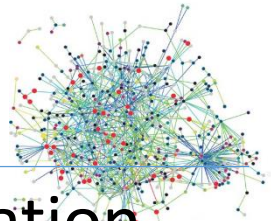
- Inexpensive to construct
- Extremely fast at classifying unknown records
- Easy to interpret for small-sized trees
- Accuracy is comparable to other classification techniques for many simple data sets

Assignment: Decision tree using Hunt's and ID3



Day	Outlook	Temperature	Humidity	Wind	PlayTennis
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

(2) Rule Based Classifier



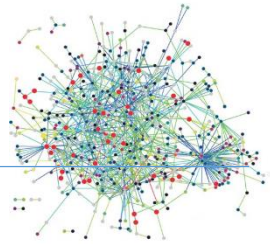
- Rule-based classifier makes use of a set of IF-THEN rules for classification. We can express a rule in the following from −

  IF *condition* THEN *conclusion*

Points to remember –

- The **IF** part of the rule is called **rule antecedent** or **precondition**.
- The **THEN** part of the rule is called **rule consequent**.
- The **antecedent** part the condition consist of one or more attribute tests and these tests are logically ANDed.
- The **consequent** part consists of class prediction.

How does Rule-Based Classifier work?



- **Case-I: If only one rule is satisfied**
- **Case-II: If more than one rules are satisfied**
- **Case-III: If no rule is satisfied**

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

Rule base are:

R1: (Give Birth = No) ^ (Can fly = Yes) => Birds

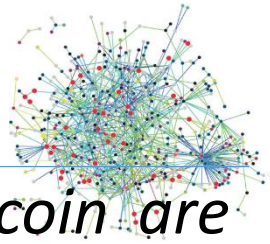
R2: (Give Birth = No) ^ (Live in Water = Yes) => Fishes

R3: (Give Birth = Yes) ^ (Blood Type = Warm) => Mammals

R4: (Give Birth = No) ^ (Can fly = No) => Reptiles

R5: (Live in Water = Sometimes) => Amphibians

Characteristics of Rule-Based Classifier



1. Mutually exclusive Rules : *Turning left and turning right, Tossing a coin are Mutually Exclusive (you can't do both at the same time)*

- Classifier contains mutually exclusive rules if all the rules are independent of each other.
- Every record is covered by at most one rule.
- Rules are no longer mutually exclusive if a record may trigger by more than one rule. To make mutually exclusive we apply rule ordering (ranked or ordered according to their priority or quality not by same class appearing together).

2. Exhaustive Rules

- Classifier has exhaustive coverage if it accounts for every possible combination of attribute values (every possible rule).
- Each record is covered by at least one rule.
- Rules are no longer exhaustive if a record may not trigger any rules. To make rules exhaustive use default class.

Approaches of Building Classification Rules

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

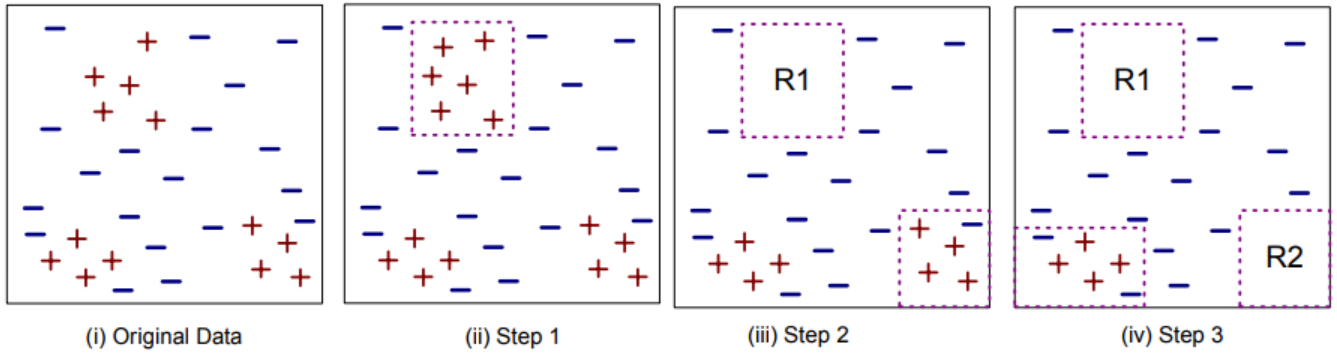


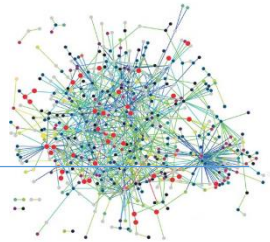
Fig. Sequential Covering

2) **Indirect Method**: Extract rules from other classification models *(e.g. learn decision trees then convert to rules, learn neural networks then extract the rules, etc.).*

Rules:
R1: (Refund = Yes) => Loan
R2: (Refund = No) ^ (Marital Status = Married) => Loan

Rule simplification
Complex rules can be simplified. In above example R2 can be simplified as:
r2: (Marital Status = Married) => Loan

Approaches of Building Classification Rules: Aspects of Sequential Covering



i. Rule Growing

– Start from an empty conjunct: {}

– Add conjuncts that minimizes the entropy measure: {A}, {A,B}, …

– Determine the rule consequent by taking majority class of instances covered by the rule

- **R0:** {} => class (initial rule)
- **R1:** {A} => class (rule after adding conjunct)

Gain (R0, R1) = t [log (p1/(p1+n1)) – log (p0/(p0 + n0))]

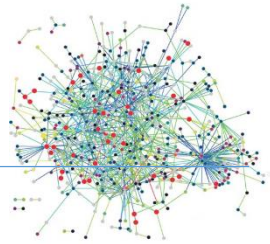
Where, t: number of positive instances covered by both R0 and R1

p0: number of positive instances covered by R0

n0: number of negative instances covered by R0

p1: number of positive instances covered by R1

n1: number of negative instances covered by R1



ii. Instance Elimination

- We need to eliminate instances otherwise; the next rule is identical to previous rule.
- We remove positive instances to ensure that the next rule is different.
- We remove negative instances to prevent underestimating accuracy of rule Compare rules R2 and R3 in the diagram

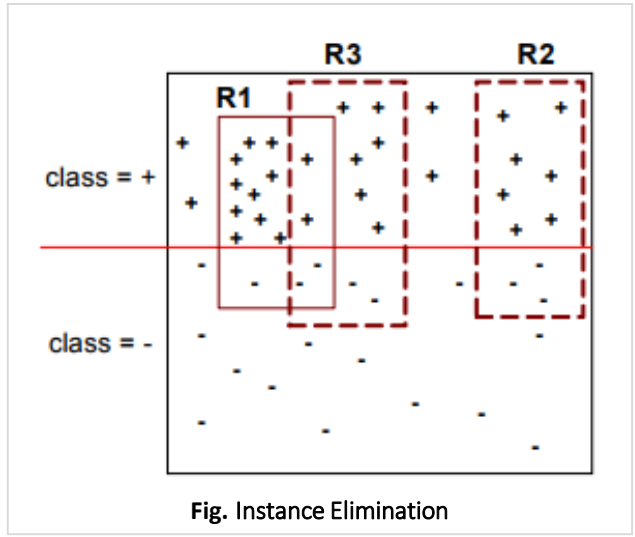
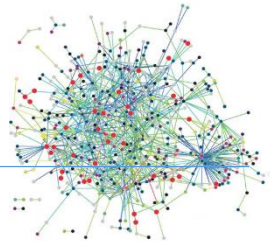


Fig. Instance Elimination

Approaches of Building Classification Rules: Aspects of Sequential Covering



iii. Rule Evaluation

- Metrics:
  - Accuracy $= \frac{n_c}{n}$
  - Laplace $= \frac{n_c + 1}{n + k}$
  - M-estimate $= \frac{n_c + kp}{n + k}$

n : Number of instances

n_c: Number of instances covered by rule

k: Number of classes

p: Prior probability

iv. Stopping Criterion

– Compute the gain, If gain is not significant, discard the new rule.

v. Rule Pruning: Similar to post-pruning of decision trees.

– Reduced Error Pruning:

- Remove one of the conjuncts in the rule
- Compare error rate on validation set before
- If error improves, prune the conjunct

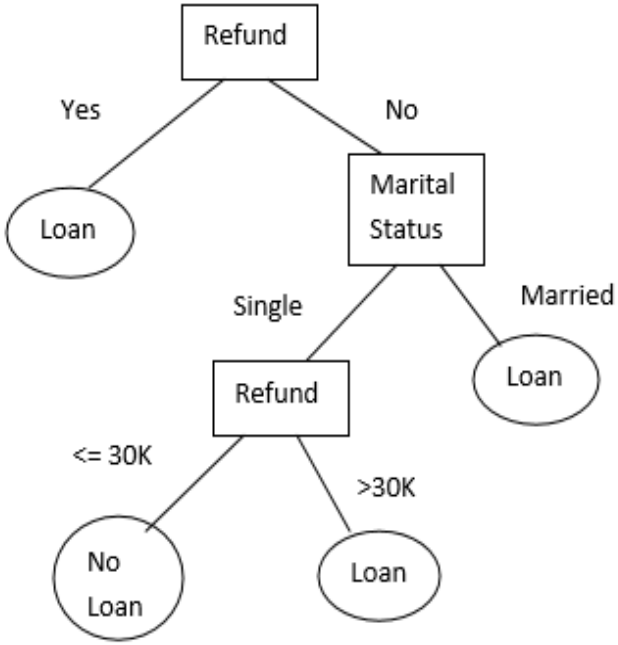
Approaches of Building Classification Rules: Indirect Method

- Extract rules from other classification models *(e.g. learn decision trees then convert to rules, learn neural networks then extract the rules, etc.).*

Rules:

R1: (Refund = Yes) => Loan

R2: (Refund = No) ^ (Marital Status = Married) => Loan



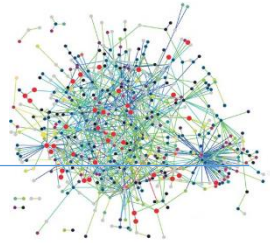
Eg; Rule Extraction from Decision Tree

Rule simplification

Complex rules can be simplified. In above example R2 can be simplified as:

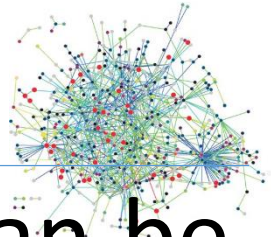
r2: (Marital Status = Married) => Loan

Advantages of Rule-Based Classifiers



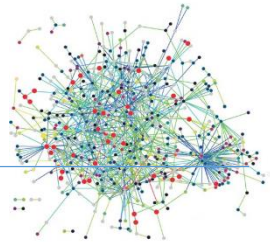
- As highly expressive as decision trees
- Easy to interpret and Easy to generate
- Can classify new instances rapidly
- Performance comparable to decision trees

(3) Nearest Neighbor Classifier



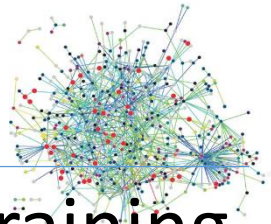
- One of the simplest decision procedures that can be used for classification technique is the nearest neighbor (NN) rule.
- It classifies a sample based on the category of its nearest neighbor

Characteristics of NN



- **Nearest neighbor classifier requires:**
  - Set of stored records
  - Distance matric to compute distance between records. For distance calculation any standard approach can be used sch as Euclidean distance.
  - The value of **'K'**, the number of nearest neighbor to retrieve.
- **To classify the unknown records**
  - Compute distance to other training records.
  - Identify the **k-nearest** neighbor.
  - Use class label nearest neighbors to determine the class label of unknown record. In case of conflict, use majority vote for classification.

NN Classification phases



- **Training** phase: a model is constructed from the training instances.

✓Classification algorithm finds relationships between predictors and targets

✓Relationships are summarized in a model

- **Testing** phase: test the model on a test sample whose class labels are known but not used for training the model
- **Usage** phase: use the model for classification on new data whose class labels are unknown

query ⇒ x = (Maths = 6, CS = 8), (K=3)

	maths	Cs	Result
1)	4	3	Fail
2)	6	7	Pass
3)	7	8	Pass
4)	5	5	Fail
5)	8	8	Pass

Euclidean distance :-

$$d = \sqrt{|x_{01} - x_{A1}|^2 + |x_{02} - x_{A2}|^2}$$

(I) $\sqrt{(6-4)^2+(8-3)^2} = \sqrt{29} = 5.38$

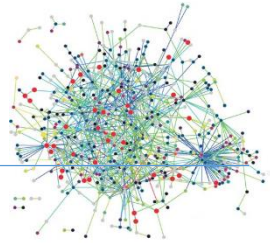
(II) $\sqrt{(6-6)^2+(8-7)^2} = 1$

(III) $\sqrt{(6-7)^2+(8-8)^2} = 1$

(IV) $\sqrt{(6-5)^2+(8-5)^2} = \sqrt{10} = 3.16$

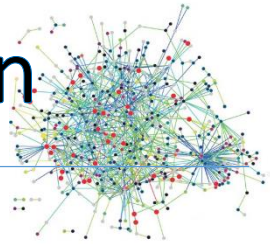
(V) $\sqrt{(6-8)^2+(8-8)^2} = 2$

NN Algorithms



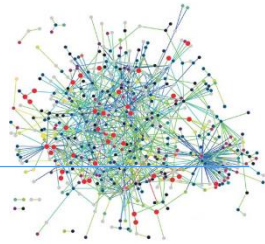
- Initialization, define k
- Compute distance (test instance, each training instance)
- Sort the distance
- Take k nearest neighbors
- Apply simple majority
- **Class**

Issues of classification using k-nearest neighbor classification



- Choosing the value of K
- Scaling Issue
- Distance computing for non-numeric data.
- Missing values

Adv. and DisAdv. Of NN Classifier



- **Advantages**
  - It is conceptually simple.
  - It does not require learning (term: memory-base).
  - It can be used even with few examples.
  - Even for moderate k: wonderful performer.
  - It works very well in low dimensions for complex decision surfaces.
- **Disadvantages**:
  - Poor accuracy when data have noise and irrelevant attributes.
  - Slow when classifying test tuples.
  - Classifying unknown records are relatively expensive

(4) Bayesian Classifier

- Predicts class membership probabilities such as the probability that a given tuple belongs to a particular class.
- **Baye's Law**

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

Where;

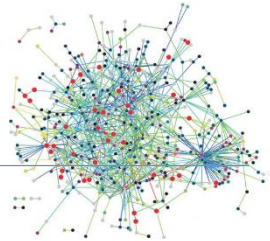
A and **B** are events and **P(B) != 0**

P(A|B) is a conditional probability: the likelihood of event **A** occurring given that **B** is true.

P(B|A) is also a conditional probability: the likelihood of event **B** occurring given that A is true.

P(A) and **P(B)** are the probabilities of observing A and **B** independently of each other; this is known as the marginal probability.

Types of Bayesian Classifier



1) Bayesian Belief Networks (Graphical Method)

- Specifies joint conditional probability distributions.
- It allows class conditional independencies to be defined between subsets of variables.
- It provides a graphical model of causal relationship on which learning can be performed.
- It represents a set of random variables and their conditional dependencies via a directed acyclic graph

2) Naïve Bayesian Classifier

$$P(X/Ci) = \prod_{k=1}^{n} P(\frac{Xk}{ci})$$
$$= P(x1/Ci) * P(x2/Ci) * \ldots\ldots\ldots\ldots * P(xn/Ci)$$

- Based on the so-called Bayesian theorem and is particularly suited when the dimensionality of the inputs is high.
- It simplifies the computational complexity.
- Naïve Bayesian Classifier assumes that the effect of an attribute value on a given class is independent of the value of other attributes i.e. class conditional independence.

Naive Bayes

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

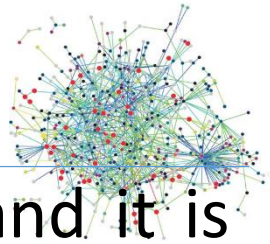
- $$P(Yes \mid Sunny) = \frac{P(Sunny \mid Yes) * P(Yes)}{P(Sunny)}$$

Here,

P (Sunny |Yes) = 3/9 = 0.33, P(Sunny) = 5/14 = 0.36, P(Yes)= 9/14 = 0.64

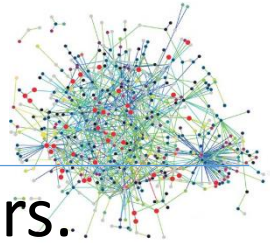
Now, P (Yes | Sunny) = 0.33 * 0.64 / 0.36 = 0.60, which has higher probability.

Applications of Naive Bayes Algorithms



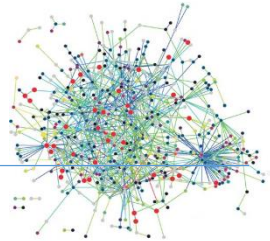
- **Real time Prediction:** Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.
- **Multi class Prediction:** This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.
- **Text classification/ Spam Filtering/ Sentiment Analysis:** Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)
- **Recommendation System:** Naive Bayes Classifier and Collaborative Filtering together builds a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not.

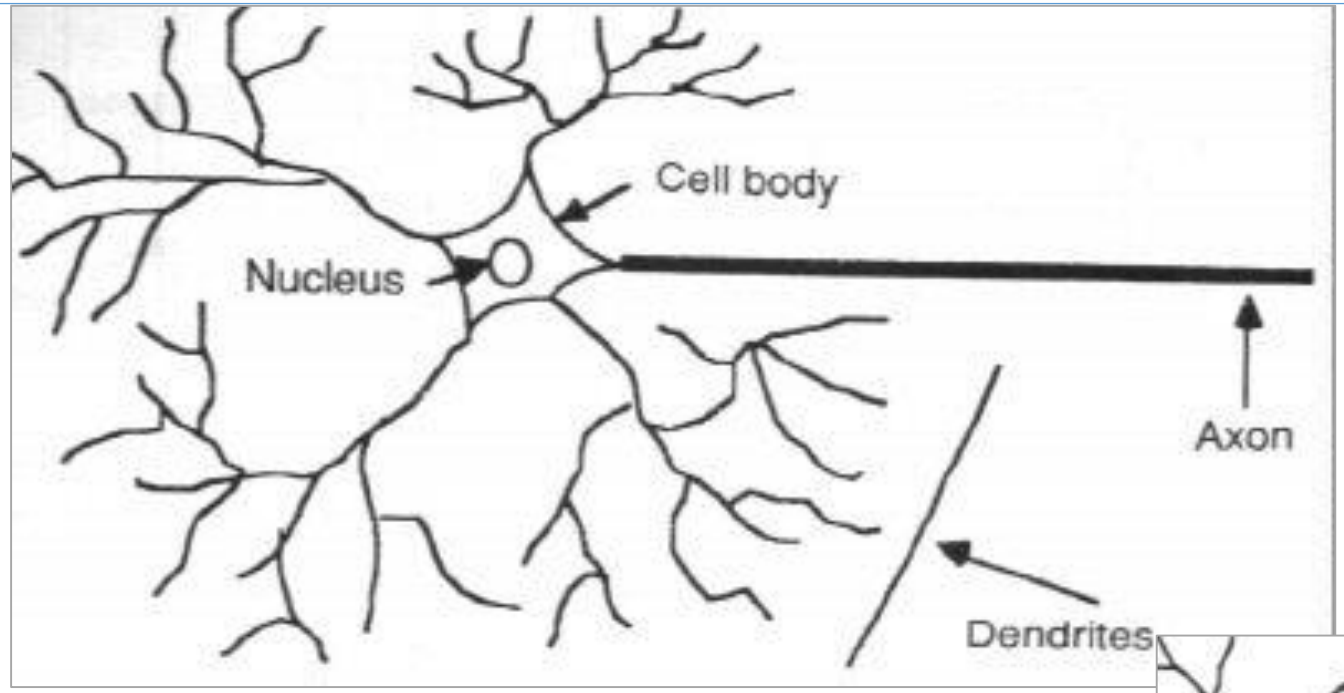
(5) Artificial Neural Networks (ANN) Classifier

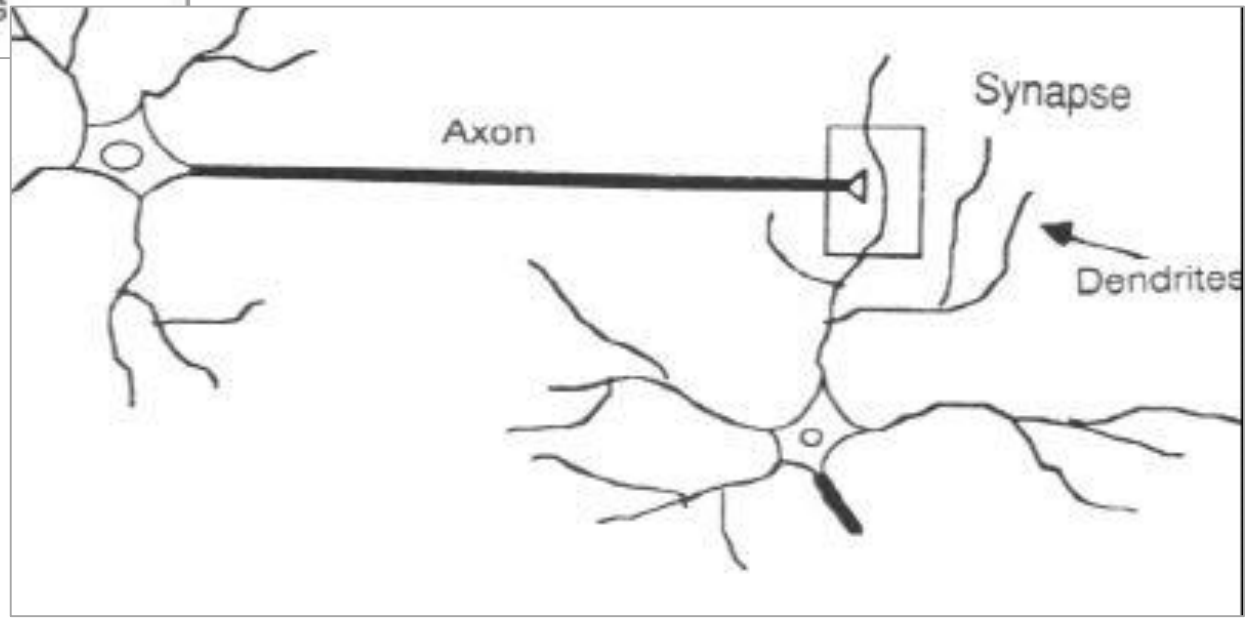


- Based on Biological System, a new method of programming computers.
- Programs that employ neural nets are also capable of learning on their own and adapting to changing conditions.
- ANN is an information processing paradigm that is inspired by the biological nervous systems, such as the human brain's information processing mechanism.
- The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. NNs, like people, learn by example.
- An NN is configured for a specific application, such as pattern recognition or data classification, through a learning process.
- Learning in biological systems involves adjustments to the synaptic (between nerve cells) connections that exist between the neurons. This is true of NNs as well.

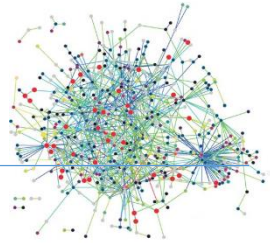
How the Human Brain learns?

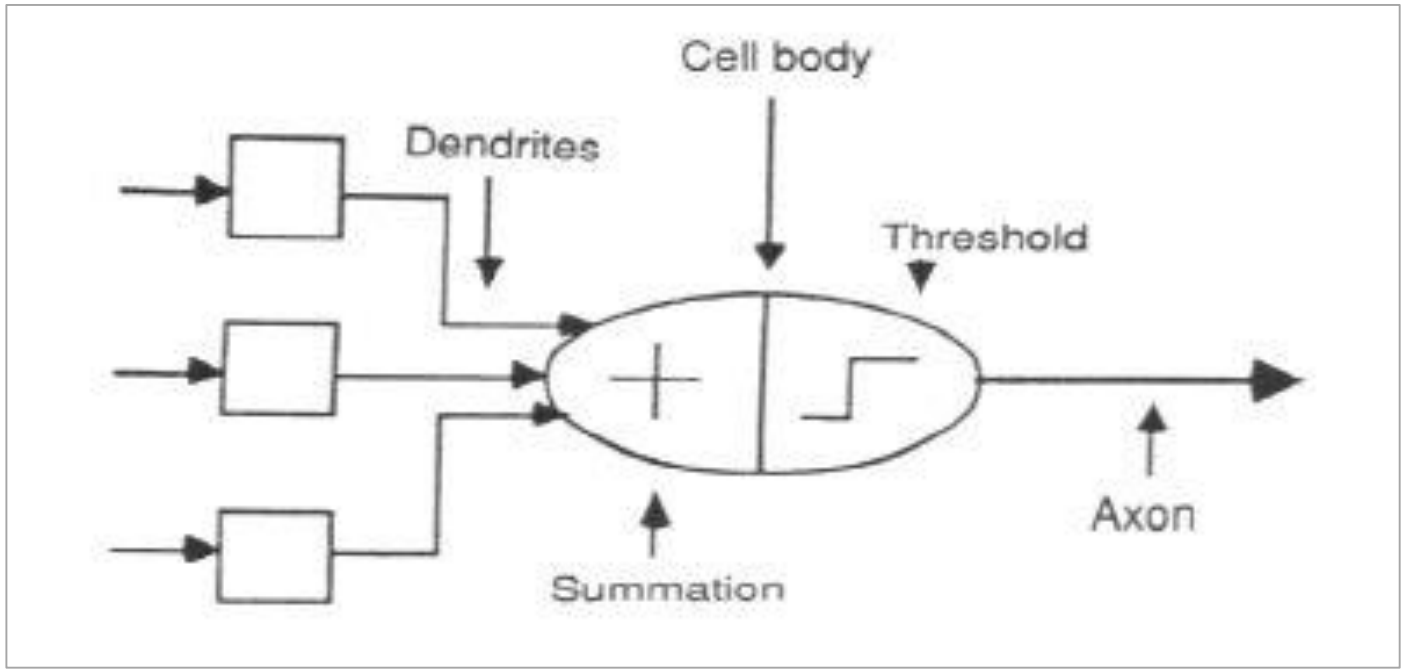


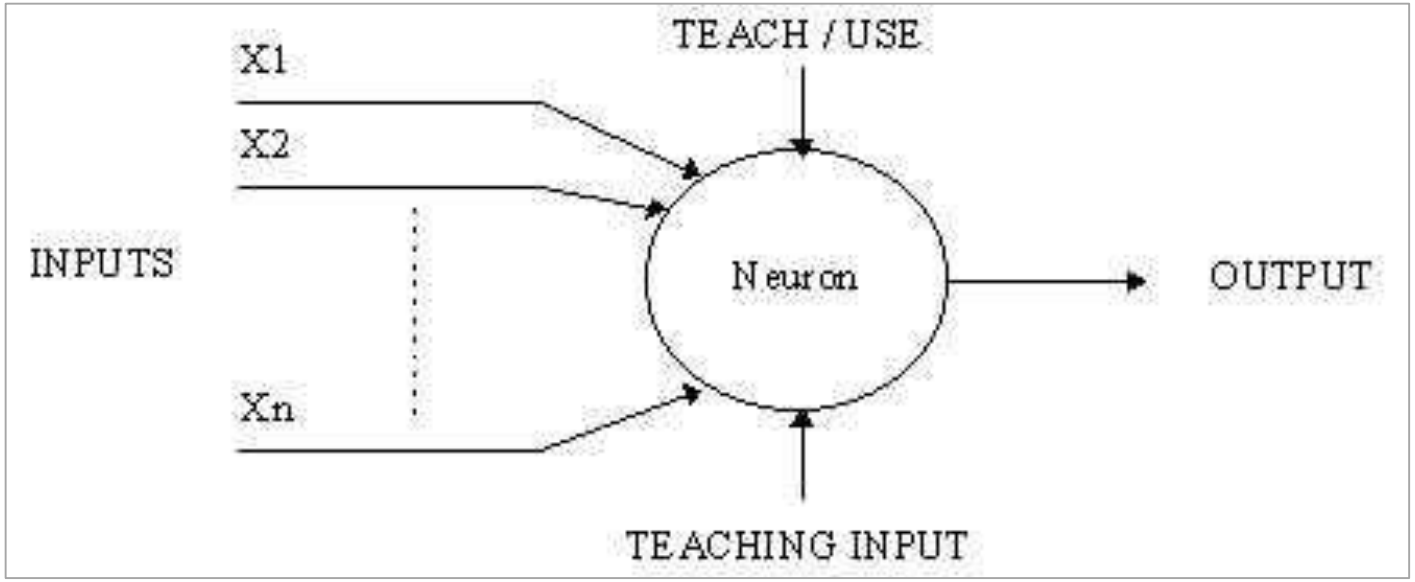




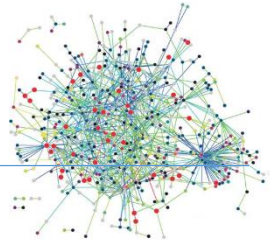
A Neuron Model

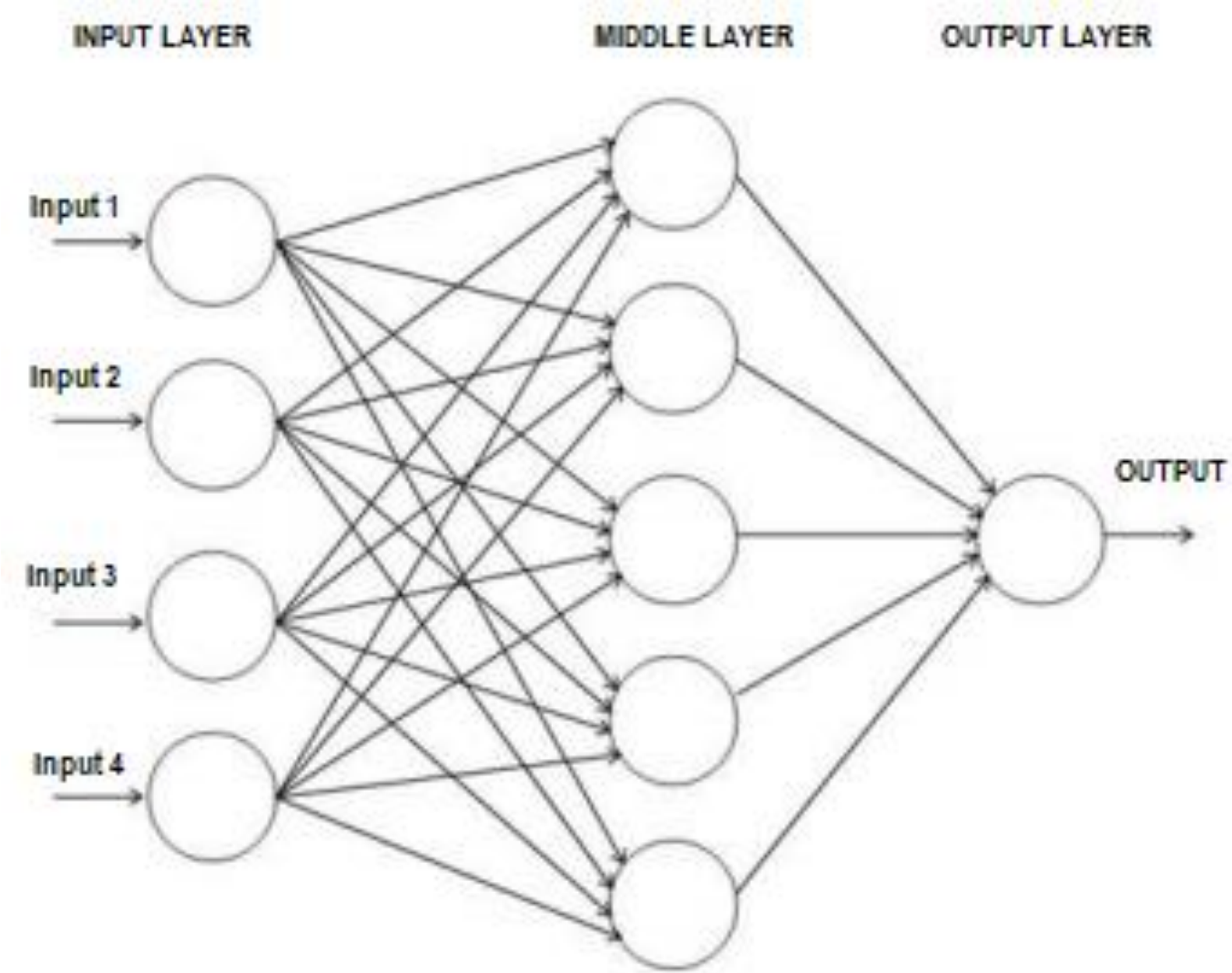




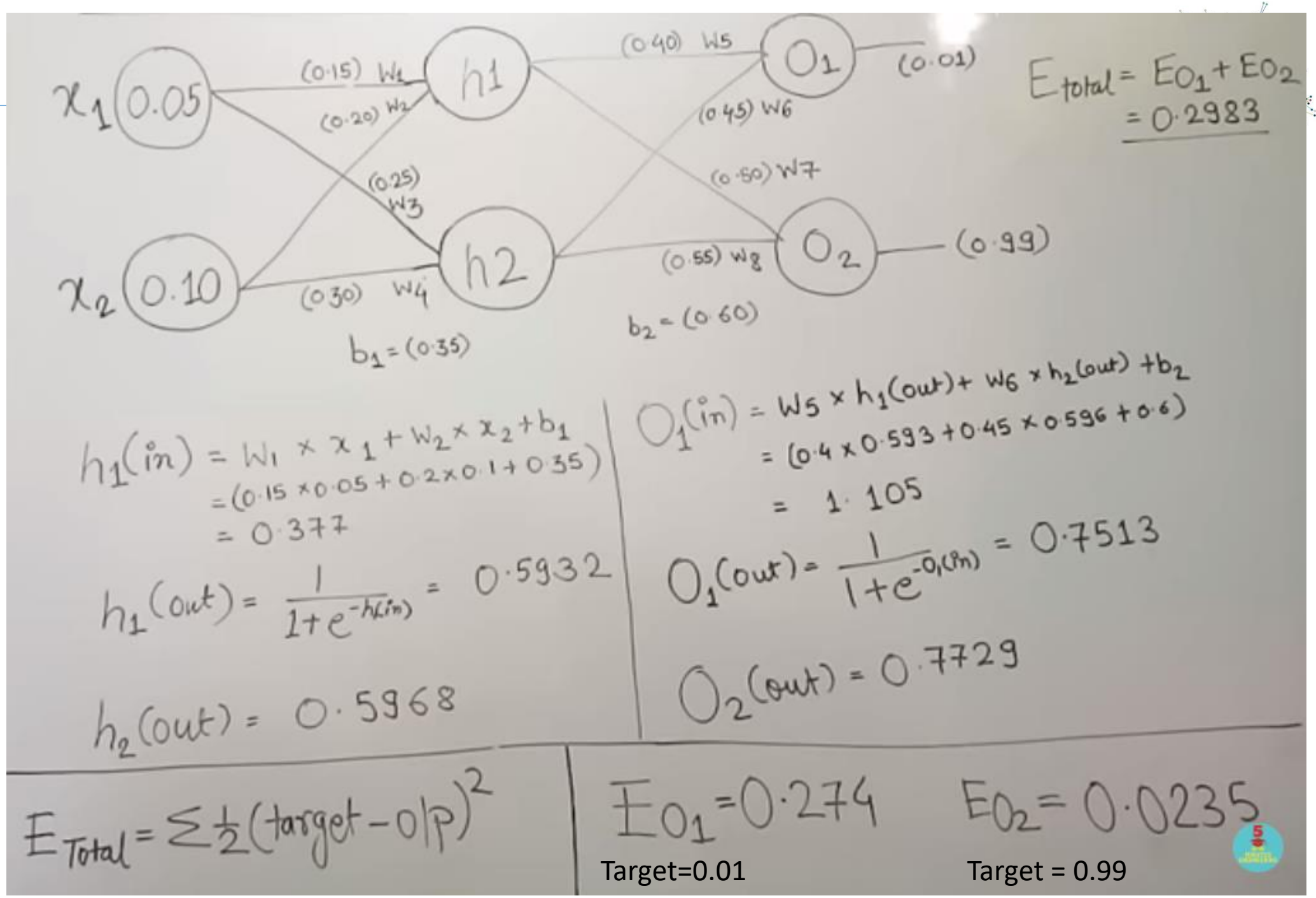


Network Layers

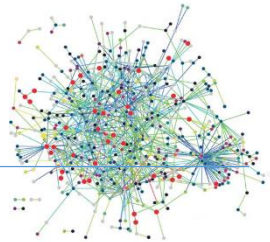




BACKPROPAGATION

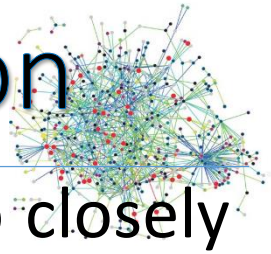


Back propagation algorithm

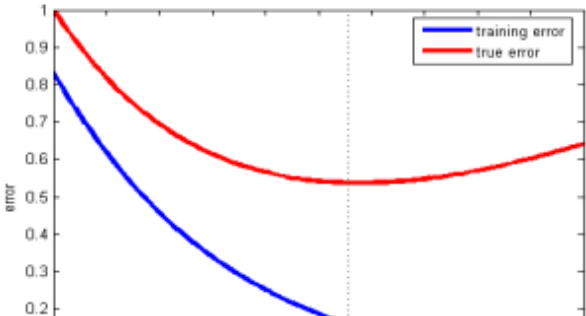


- Backpropagation is a common method for training a neural network.
- **Step 1: Initialization**: Set all the weights and thresholds levels of the network to random numbers uniformly distributed inside a small range.
- **Step 2: Activation**: Activate the back propagation neural network by applying i/ps and desired o/ps.
  - Calculate the actual o/ps of the neurons in the hidden layers.
  - Calculate the actual o/ps of the neurons in the o/p layers.
- **Step 3: Weight training**:
  - Updates weights in the back-propagation network by propagating backwards the errors associated with the o/p neurons.
  - Calculate error gradient of o/p layer and hence of neurons in the hidden layer.
- **Step 4: Iteration**: Increase iteration by repeating steps 2 and 3 until selected error criteria is satisfied.

Issues: Overfitting, Validation, Model Comparison



- **Overfitting** is a modeling error which occurs when a function is too closely fit to a limited set of data points.
- **Overfitting** the model generally takes the form of making an overly complex model to explain idiosyncrasies in the data under study.
- Overfitting occurs when a statistical model describes random error or noise instead of the underlying relationship.
- Overfitting generally occurs when a model is excessively complex, such as having too many parameters relative to the number of observations.
- A model which has been overfit will generally have poor predictive performance.
- Overfitting depends not only on the number of parameters and data but also the conformability of the model structure.
- In order to avoid overfitting, it is necessary to use add (e.g. cross-validation, pruning (Pre or Post), model compa



Reason of Overfitting

- Noise in training data: e.g. Fig(a) decision boundary is distorted by noise point
- Incomplete training data: e.g. Fig(b) difficult to predict correctly the class labels of the region
- Error in assumed theory.

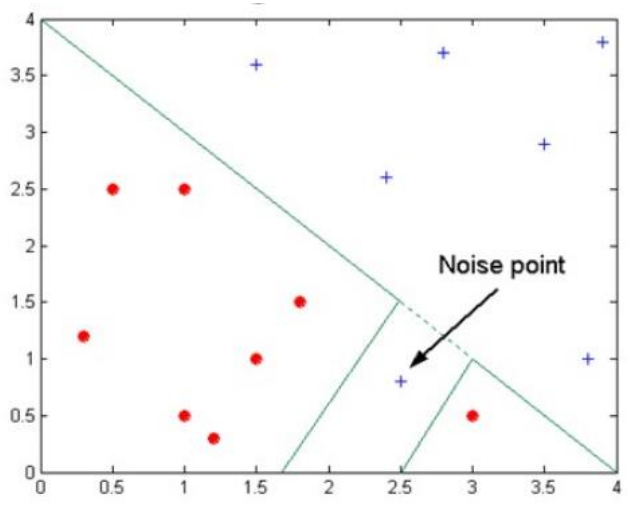


Fig. Overfitting due to the Noise

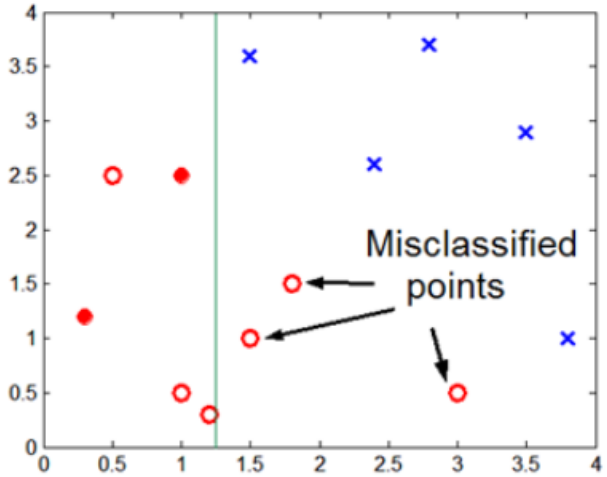
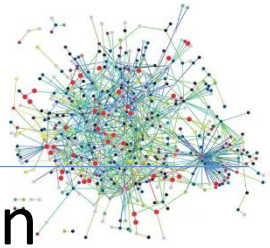


Fig.(b) Overfitting due to the incomplete data

Validation

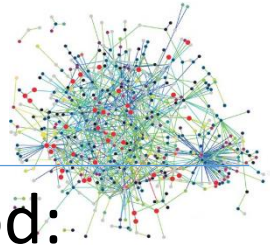


- Validation techniques are motivated by two fundamental problems in pattern recognition: model selection and performance estimation
- **Validation Approaches:**
  - One approach is to use the entire training data to select our classifier and estimate the error rate, but the final model will normally overfit the training data.
  - A much better approach is to split the training data into disjoint subsets cross validation (The Holdout Method)
- **Cross Validation (The holdout method)**

Data set divided into two groups.

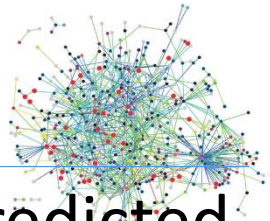
- ✓**Training set**: used to train the classifier and
- ✓**Test set**: used to estimate the error rate of the trained classifier
- ✓**Total number of examples** = Training Set +Test Set

Model Evaluations



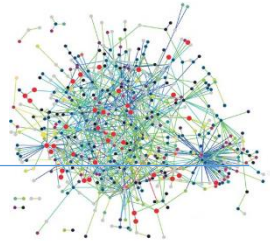
- Models can be evaluated based on the output using different method:
  - Confusion Matrix
  - ROC Analysis
  - Others such as: Gain and Lift Charts, K-S Charts

Confusion Matrix (Contingency Table)



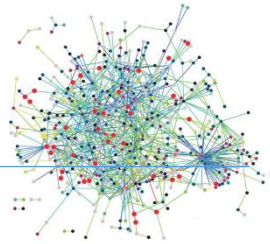
- A confusion matrix contains information about actual and predicted classifications done by classifier.
- Performance of such system is commonly evaluated using data in the matrix.
- It is also known as a contingency table or an error matrix, is a specific table layout that allows visualization of the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

		Predicted	
		Positive	**Negative**
Actual Class	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)



- **Error rate =** (FP+FN)/(TP+TN+FN+FP)
- **Accuracy** (total no of predictions that were correct) = (TP + TN) / Total data count (TP+TN+FP+FN)
- **Precision** or Positive Predictive Value (proportion of positive cases that were correctly identified) = TP / (TP + FP) or TN/ (TN + FN)
- **Sensitivity** or Recall or True Positive Rate (TPR) (the proportion of actual positive cases which are correctly identified) = TP/(TP+FN)
- **Specificity** or True Negative Rate (TNR)= TN / (TN+FP)
- Fall-Out or False Positive Rate (FPR)= FP / (FP +FN)
- Miss Rate or False Negative Rate (FNR)= FN / (FP +FN)
- Negative Predicate Value (NPV) = TN/(TN+FN)
- False Discovery Rate (FDR) = FP/(FP+TP)

Example: Confusion Matrix



	a = yes	B = no
a = yes	6	4
b = no	2	8

Measure		**calculated value**
Error rate	ERR	6 / 20 = 0.3
Accuracy	ACC	14 / 20 = 0.7
Sensitivity True positive rate Recall	SN TPR REC	6 / 10 = 0.6
Specificity True negative rate	SP TNR	8 / 10 = 0.8
Precision Positive predictive value	PREC PPV	6 / 8 =0.75
False positive rate	FPR	2 / 10 = 0.2

ROC Analysis: graphical plot

- Receiver Operating Characteristic (ROC), or ROC curve, is a **graphical plot** that illustrates the performance of a binary classifier system as its discrimination threshold is varied.
- The curve is created by plotting the **true positive rate against the false positive rate** at various threshold settings.