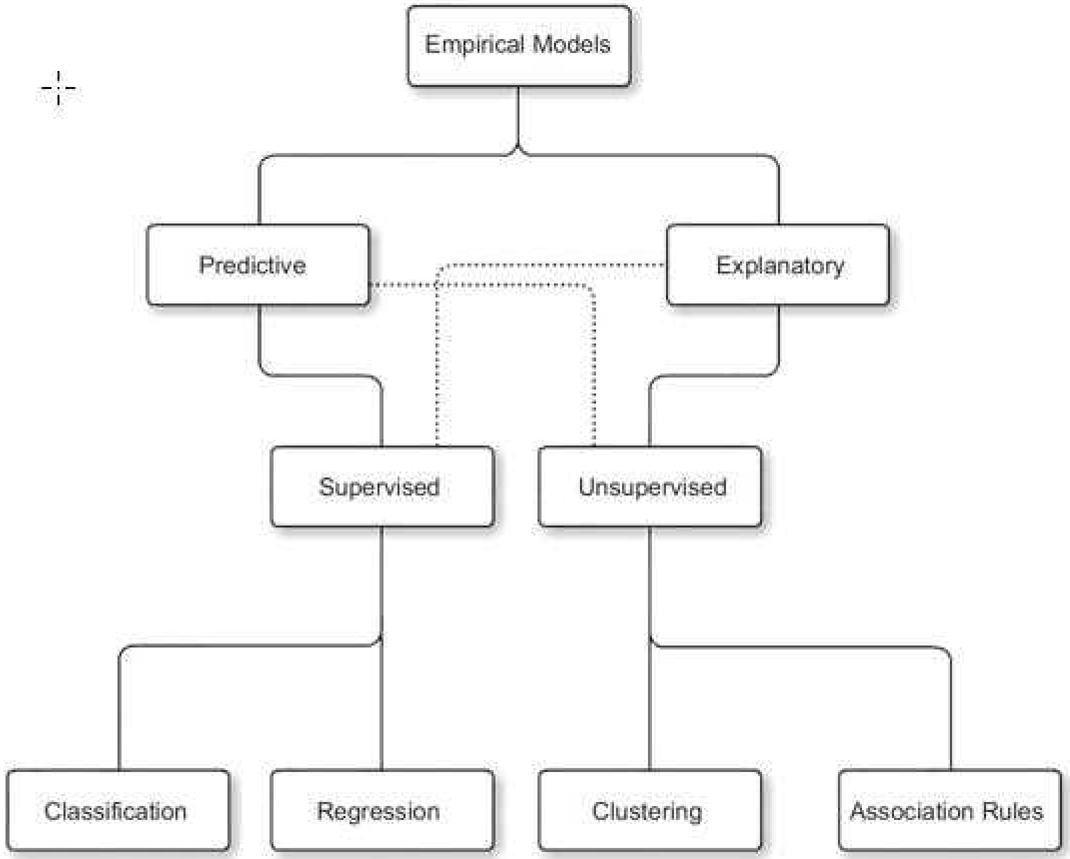
4. Classification

Models



Empirical Models
Predictive
Explanatory
Supervised
Unsupervised
Classification
Regression
Clustering
Association Rules

Classification

Target variable is categorical. Predictors could be of any data type.

Algorithms

Decision Trees

Rule induction

kNN

Naive Bayesian

Neural Networks

Support Vector Machines

Ensemble Meta Models

Decision Trees

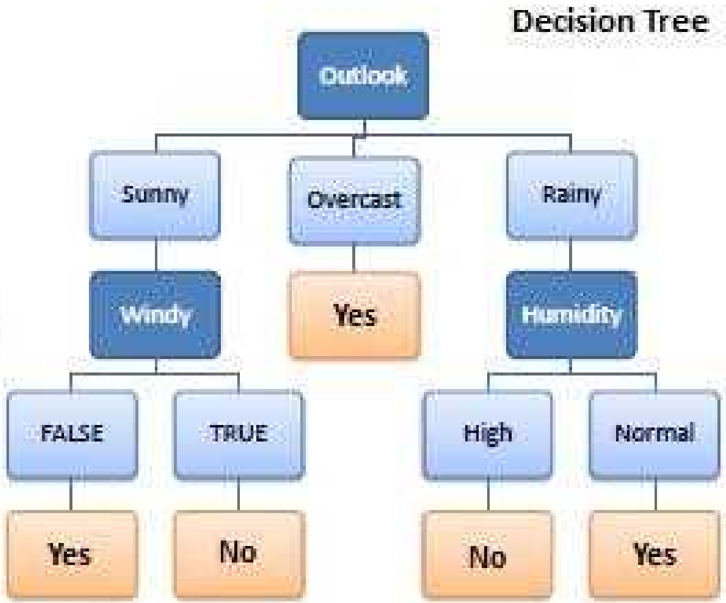
Play golf dataset

Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
overcast	hot	high	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	cool	normal	FALSE	yes
rainy	cool	normal	TRUE	no
overcast	cool	normal	TRUE	yes
sunny	mild	high	FALSE	no
sunny	cool	normal	FALSE	yes
rainy	mild	normal	FALSE	yes
sunny	mild	normal	TRUE	yes
overcast	mild	high	TRUE	yes
overcast	hot	normal	FALSE	yes
rainy	mild	high	TRUE	no

Decision Trees

Predictors | Target

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



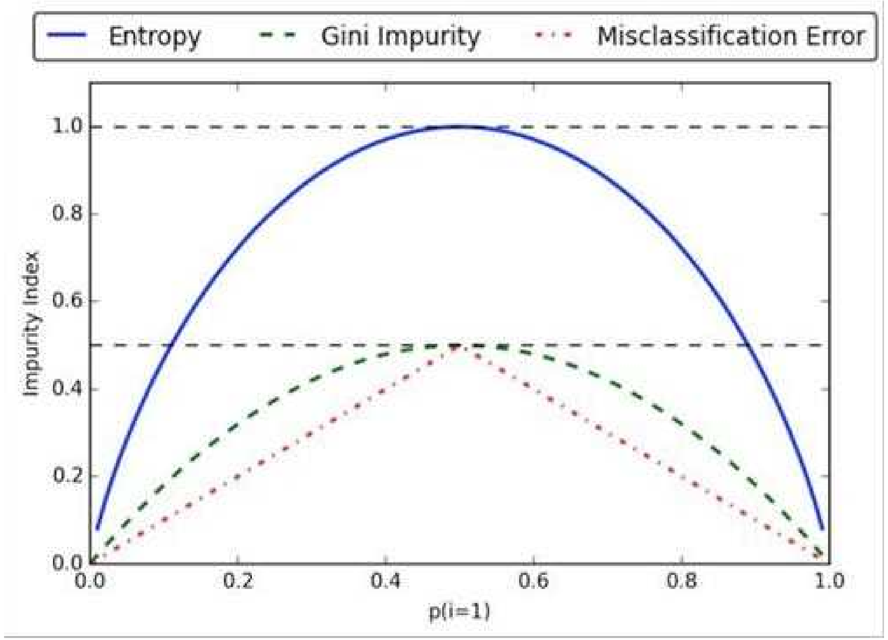
Measure of impurity

Every split ties to make child node more pure.

Gini impurity

Information Gain (Entropy)

Misclassification Error



https://www.quora.com/What-are-the-advantages-of-different-Decision-Trees-Algorithms

Table 4.2 Computing the Information Gain for All Attributes

Attribute	Information Gain
Temperature	0.029
Humidity	0.102
Wind	0.048
Outlook	0.247

Row No.	Play	Outlook
1	no	sunny
2	no	sunny
3	yes	overcast
4	yes	rain
5	yes	rain
6	no	rain
7	yes	overcast
8	no	sunny
9	yes	sunny
10	yes	rain
11	yes	sunny
12	yes	overcast
13	yes	overcast
14	no	rain

Entropy

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

Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)

= Entropy (0.36, 0.64)

= - (0.36 $\log_2$ 0.36) - (0.64 $\log_2$ 0.64)

= 0.94

$$E(T, X) = \sum_{c \in X} P(c) E(c)$$

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

E(PlayGolf, Outlook) = **P**(Sunny)***E**(3,2) + **P**(Overcast)***E**(4,0) + **P**(Rainy)***E**(2,3)

= (5/14)*0.971 + (4/14)*0.0 + (5/14)*0.971

= 0.693

IG

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			

$$Gain(T,X) = Entropy(T) - Entropy(T,X)$$

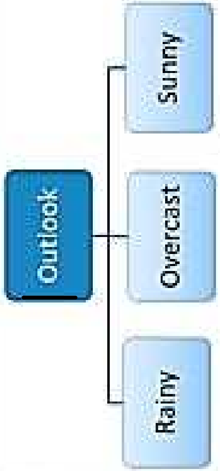
G(PlayGolf, Outlook) = **E**(PlayGolf) – **E**(PlayGolf, Outlook)

= 0.940 – 0.693 = 0.247

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

Sub table / sub tree

1



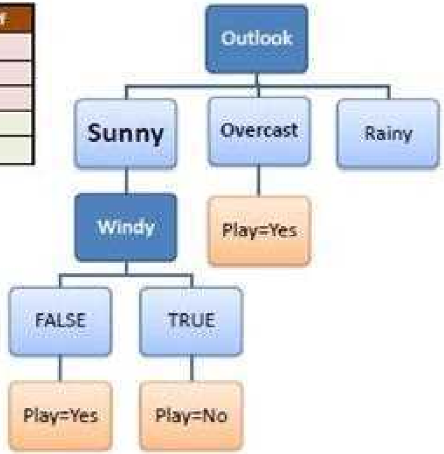
Outlook	Temp	Humidity	Windy	Play Golf
Sunny	Mild	High	FALSE	Yes
Sunny	Cool	Normal	FALSE	Yes
Sunny	Cool	Normal	TRUE	No
Sunny	Mild	Normal	FALSE	Yes
Sunny	Mild	High	TRUE	No
Overcast	Hot	High	FALSE	Yes
Overcast	Cool	Normal	TRUE	Yes
Overcast	Mild	High	TRUE	Yes
Overcast	Hot	Normal	FALSE	Yes
Rainy	Hot	High	FALSE	No
Rainy	Hot	High	TRUE	No
Rainy	Mild	High	FALSE	No
Rainy	Cool	Normal	FALSE	Yes
Rainy	Mild	Normal	TRUE	Yes

2

Temp	Humidity	Windy	Play Golf
Hot	High	FALSE	Yes
Cool	Normal	TRUE	Yes
Mild	High	TRUE	Yes
Hot	Normal	FALSE	Yes

3

Temp	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No



Decision tree – rule extraction

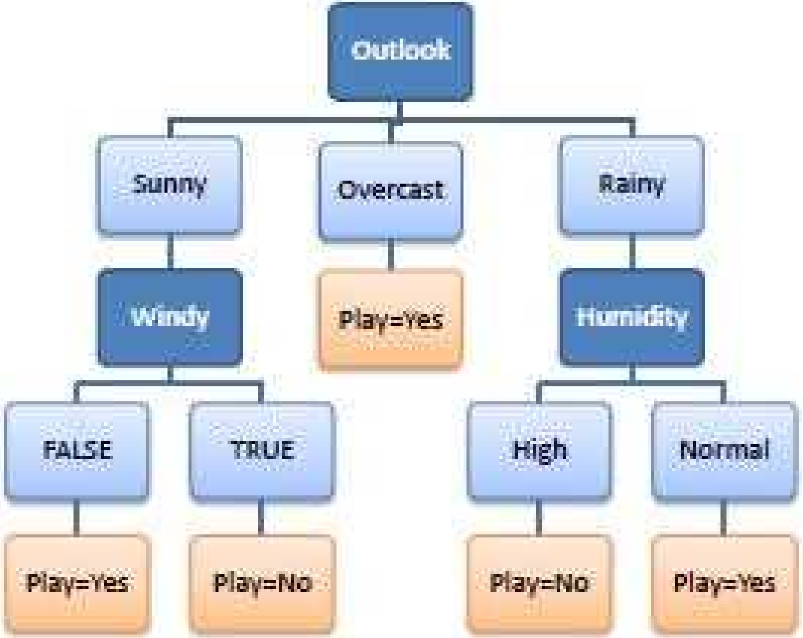
R_1: IF (Outlook=Sunny) AND (Windy=FALSE) THEN Play=Yes

R_2: IF (Outlook=Sunny) AND (Windy=TRUE) THEN Play=No

R_3: IF (Outlook=Overcast) THEN Play=Yes

R_4: IF (Outlook=Rainy) AND (Humidity=High) THEN Play=No

R_5: IF (Outlook=Rain) AND (Humidity=Normal) THEN Play=Yes



Decision Tree Algorithm

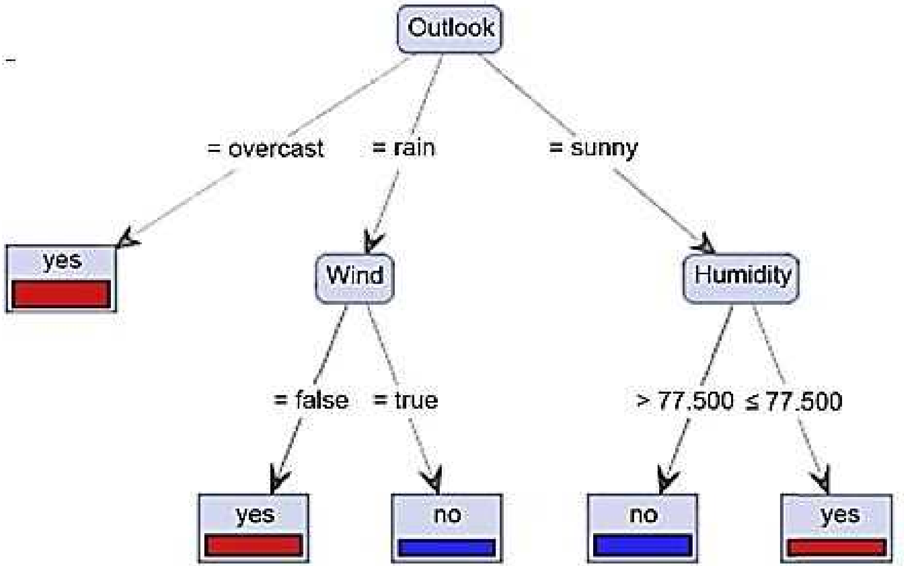
- A general algorithm for a decision tree can be described as follows:
  1. Pick the best attribute (highest IG)
  2. Make branches,
  3. for each branch **repeat** the process for the subset / sub table until one of stop conditions is satisfied.

DT: When to Stop Splitting Data?

- No attribute satisfies a minimum information gain threshold
- A maximal depth is reached
- There are less than a certain number of examples in the current subtree

Rule Induction

Tree to Rules



Rule 1: if (Outlook = overcast) then yes

Rule 2: if (Outlook = rain) and (Wind = false) then yes

Rule 3: if (Outlook = rain) and (Wind = true) then no

Rule 4: if (Outlook = sunny and (Humidity > 77.5) then no

Rule 5: if (Outlook = sunny and (Humidity ≤ 77.5) then yes

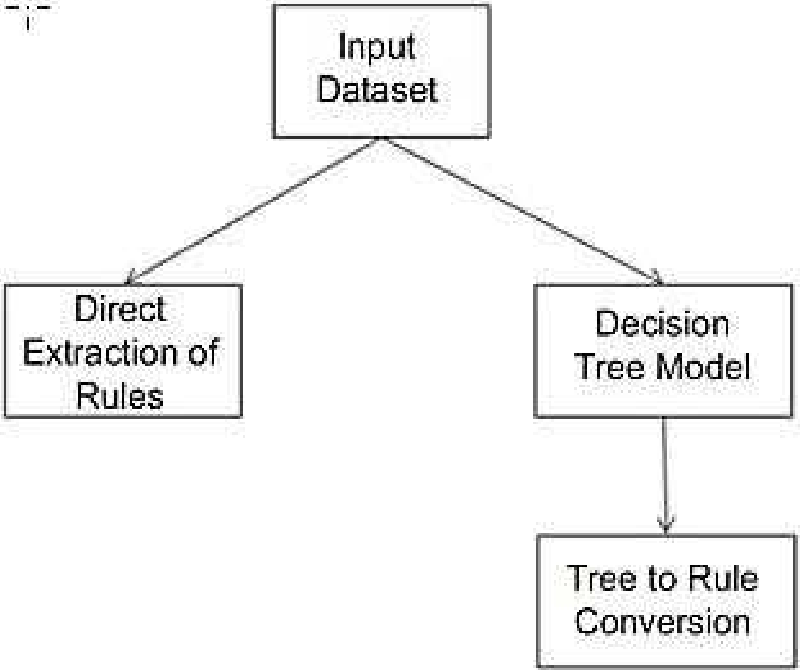
Rules

$R = \{ r_1 \cap r_2 \cap r_3 \cap .. r_k \}$

Where k is the number of disjuncts in a rule set. Individual disjuncts can be represented as

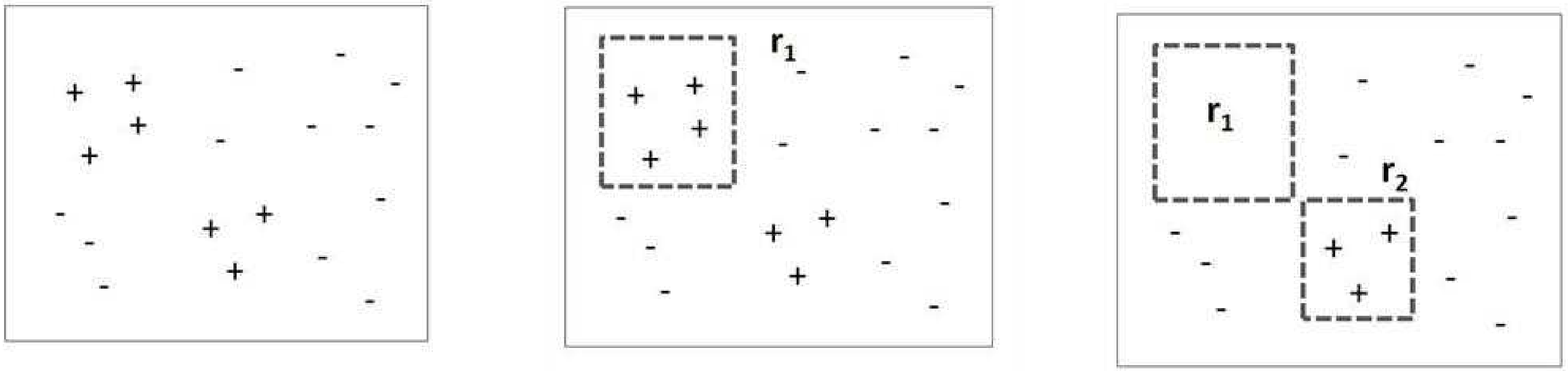
r_i = (antecedent or condition) then (consequent)

Approaches



Input Dataset
Direct Extraction of Rules
Decision Tree Model
Tree to Rule Conversion

Sequential covering



$$\text{Rule accuracy } A(r_i) = \frac{\text{Correct records covered by rule}}{\text{All records covered by the rule}}$$

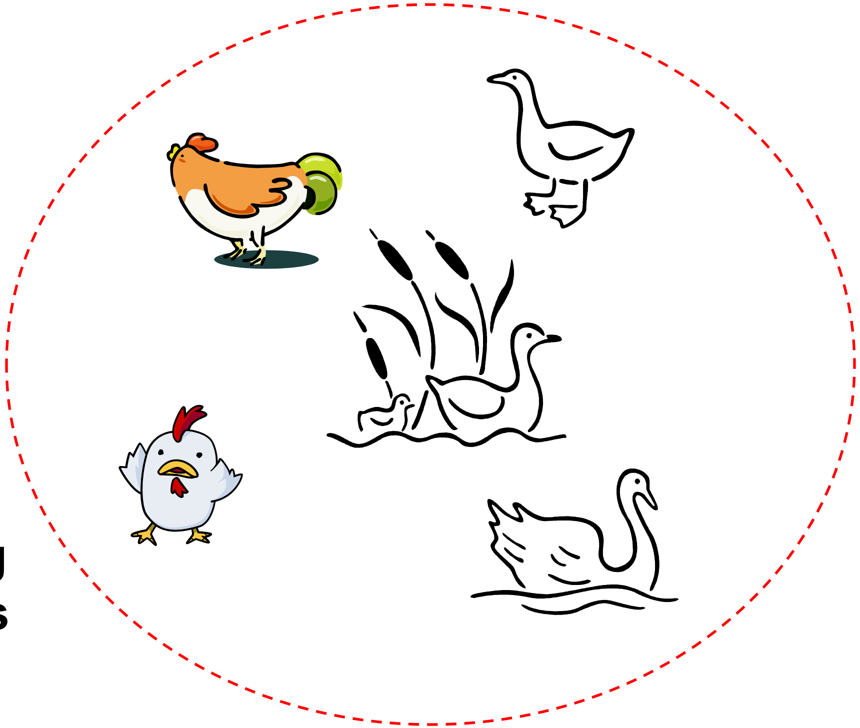
Rule-based Classifier (Example)

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

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds
R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes
R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals
R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles
R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

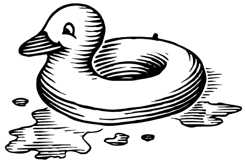
K Nearest Neighbors

KNN

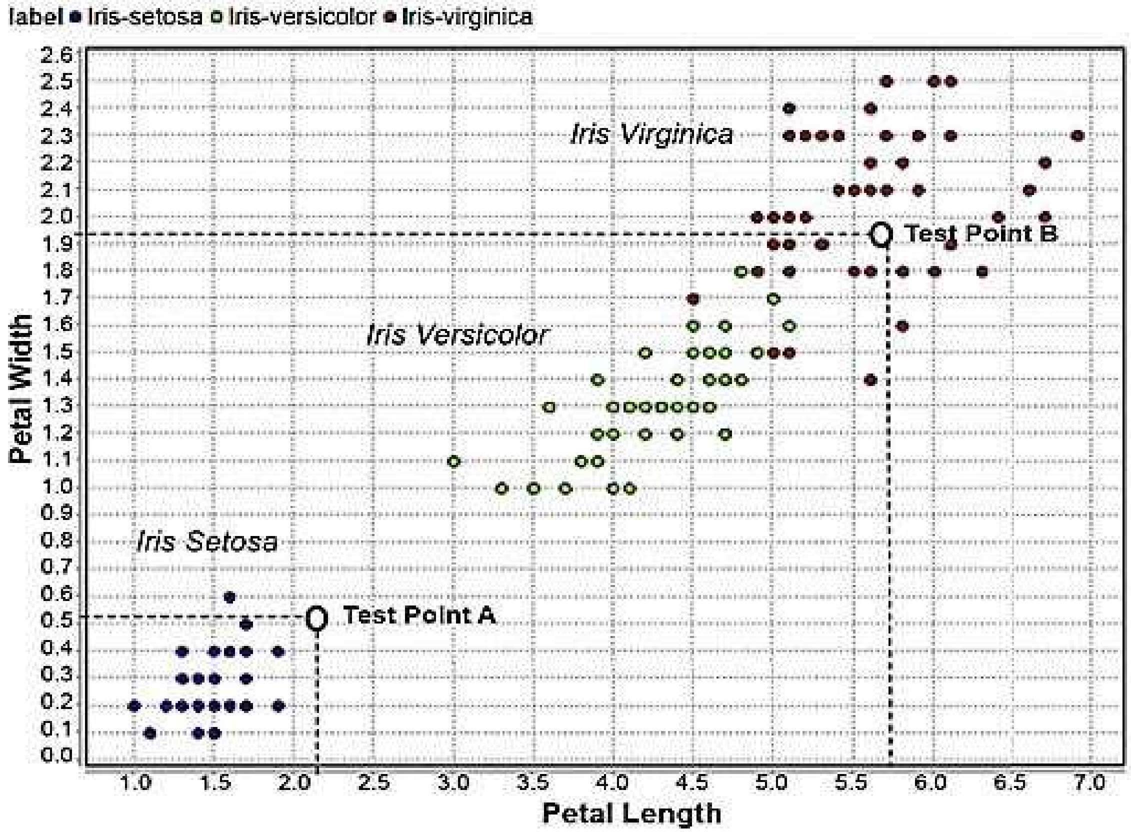


Training Records

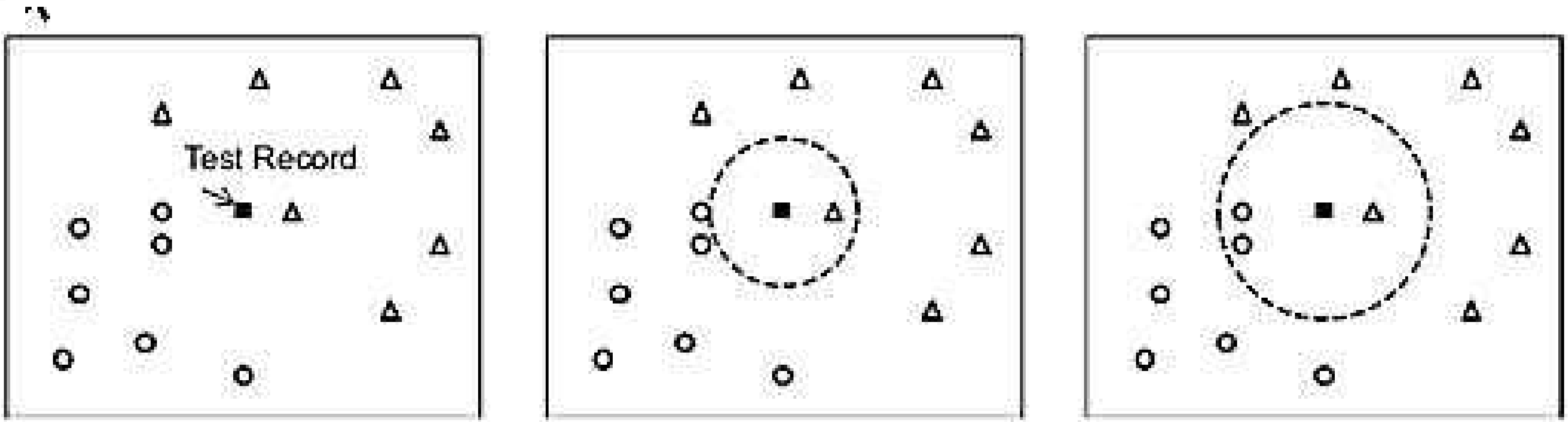
Test Record



Guess the species for A and B



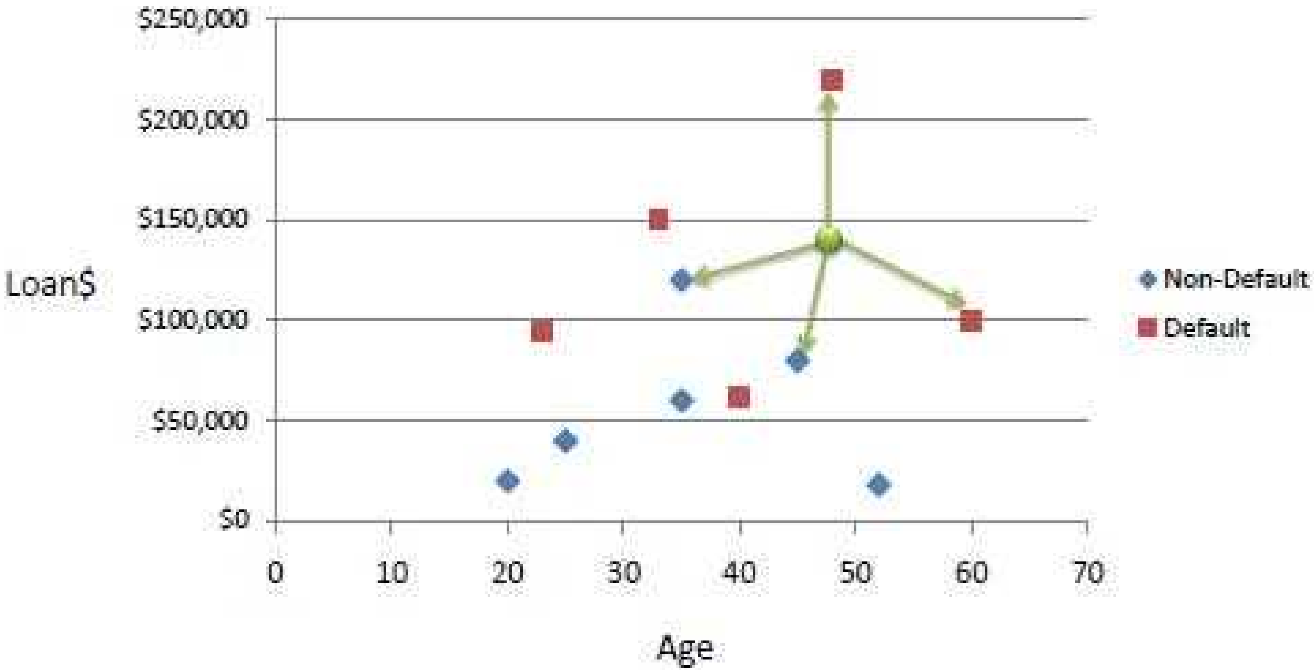
KNN



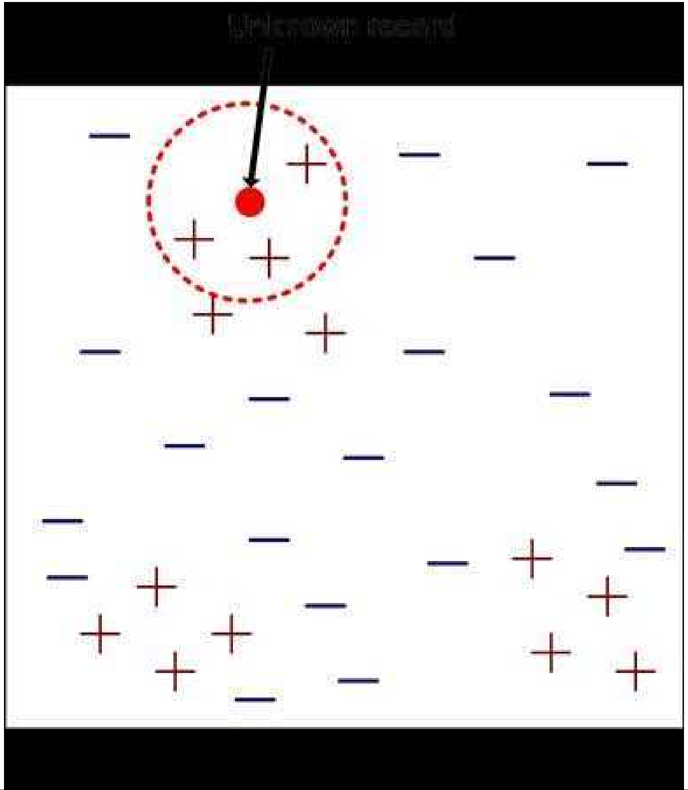
K=1 Predicted Class is triangle

K=3 Predicted Class is circle

kNN, k=3


$250,000
$200,000
$150,000
$100,000
$50,000
$0
Loan$
0
10
20
30
40
50
60
70
Age
Non-Default
Default

Nearest-Neighbor Classifiers



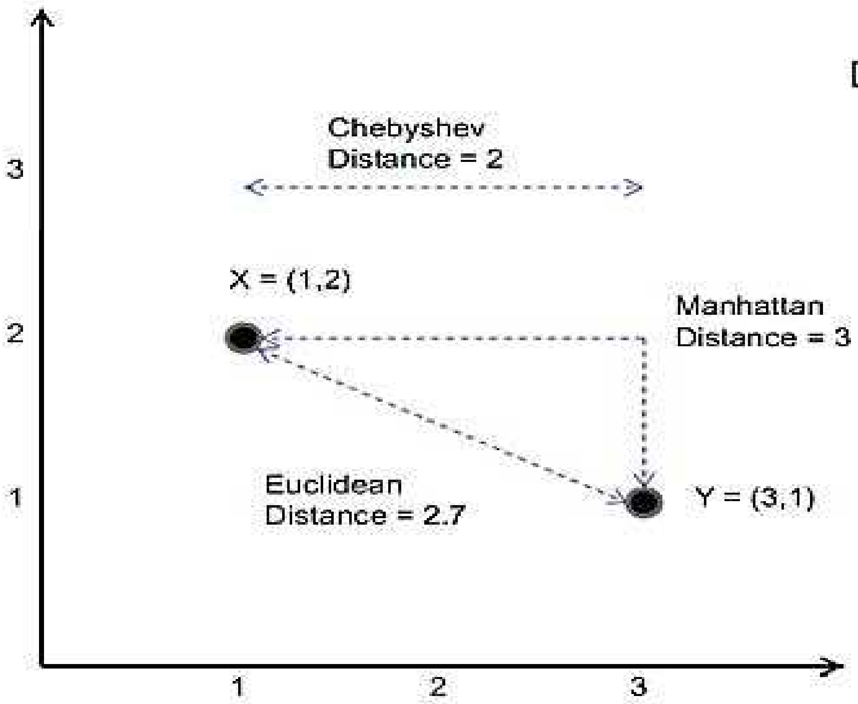
Requires the following:

- A set of labeled records
- Proximity metric to compute distance/similarity between a pair of records e.g., Euclidean distance
- The value of *k*, the number of nearest neighbors to retrieve
- A method for using class labels of K nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)
- **Take the majority vote of class labels among the k-nearest neighbors**

Measure of Proximity

Distance

$$\text{Distance d} = \sqrt{(x1 - y1)^2 + (x2 - y2)^2}$$



$$\text{Distance d} = \sqrt{(x1 - y1)^2 + (x2 - y2)^2 + \cdots + (xn - yn)^2}$$

Measure of Proximity

Correlation similarity

$$\text{Correlation } (X, Y) = \frac{s_{xy}}{s_x * s_y}$$

Simple matching coefficient

$$\text{Simple matching coefficient } (SMC) = \frac{\textit{matching occurences}}{\textit{total occurences}}$$

Jaccard Similarity

$$\text{Jaccard coefficient} = \frac{\textit{common occurences}}{\textit{total occurences}}$$

Cosine similarity

$$\text{Cosine similarity } (|X, Y|) = \frac{x \cdot y}{\| x \| \, \| y \|}$$

Hamming distance

Hamming Distance

$$D_H = \sum_{i=1}^{k} |x_i - y_i|$$

$$x = y \Rightarrow D = 0$$

$$x \neq y \Rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

KNN algorithm steps

1. Choose the number of nearest neighbors (K) and the distance metric to use.
2. Train the model by storing the data in memory. Since KNN is a lazy learner, it does not build a model during the training phase. Instead, it stores the training data in memory and uses it to make predictions during the testing phase.
3. Test the model by making predictions on new data.
   - To make a prediction for a new data point:
     - The KNN algorithm calculates the distance between the new data point and all the points in the training dataset.
     - Then selects the K nearest neighbors based on the chosen distance metric and uses them to make a prediction.
     - For **classification** tasks, the prediction is the **most common class** among the K nearest neighbors. For **regression** tasks, the prediction is the **average** of the values of the K nearest neighbors.
4. Evaluate the model by measuring its performance on the test data. This

NAÏVE BAYESIAN

Bayes' theorem

Probability of the outcome

Class conditional probability

$$P(Y|X) = \frac{P(Y) * P(X|Y)}{P(X)}$$

Posterior probability

Probability of the conditions

Bayes Classifier

- A probabilistic framework for solving classification problems
- Conditional Probability:

$$P(Y \mid X) = \frac{P(X,Y)}{P(X)}$$

$$P(X \mid Y) = \frac{P(X,Y)}{P(Y)}$$

- Bayes theorem:

$$P(Y \mid X) = \frac{P(X \mid Y)P(Y)}{P(X)}$$

Data set

Table 4.4 Golf Data Set with Modified Temperature and Humidity Attributes

No.	Temperature X_1	Humidity X_2	Outlook X_3	Wind X_4	Play (Class Label) Y
1	high	med	sunny	false	no
2	high	high	sunny	true	no
3	low	low	rain	true	no
4	med	high	sunny	false	no
5	low	med	rain	true	no
6	high	med	overcast	false	yes
7	low	high	rain	false	yes
8	low	med	rain	false	yes
9	low	low	overcast	true	yes
10	low	low	sunny	false	yes
11	med	med	rain	false	yes
12	med	low	sunny	true	yes
13	med	high	overcast	true	yes
14	high	low	overcast	false	yes

Frequency Table

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3

Likelihood Table

		Play Golf	
		Yes	No
Outlook	Sunny	3/9	2/5
	Overcast	4/9	0/5
	Rainy	2/9	3/5

		Play Golf	
		Yes	No
Humidity	High	3/9	4/5
	Normal	6/9	1/5

		Play Golf	
		Yes	No
Temp.	Hot	2/9	2/5
	Mild	4/9	2/5
	Cool	3/9	1/5

		Play Golf	
		Yes	No
Windy	False	6/9	2/5
	True	3/9	3/5

Frequency Table		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3

$$P(x \mid c) = P(Sunny \mid Yes) = 3/9 = 0.33$$

Likelihood Table		Play Golf		
		Yes	No	
Outlook	Sunny	3/9	2/5	5/14
	Overcast	4/9	0/5	4/14
	Rainy	2/9	3/5	5/14
		9/14	5/14	

$$P(x) = P(Sunny)$$
$$= 5/14 = 0.36$$

$$P(c) = P(Yes) = 9/14 = 0.64$$

Posterior Probability:

$$P(c \mid x) = P(Yes \mid Sunny) = 0.33 \times 0.64 \div 0.36 = 0.60$$

Frequency Table		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
		9	5	14

$$P(x \mid c) = P(Sunny \mid No) = 2/5 = 0.4$$

$$P(x) = P(Sunny)$$
$$= 5/14 = 0.36$$

$$P(c) = P(No) = 5/14 = 0.36$$

Posterior Probability: $P(c \mid x) = P(No \mid Sunny) = 0.40 \times 0.36 \div 0.36 = 0.40$

Class conditional probability

Class Conditional Probability of Temperature		
Temperature (X_1)	**$P(X_1 \mid Y = no)$**	**$P(X_1 \mid Y = yes)$**
high	2/5	2/9
med	1/5	3/9
low	2/5	4/9

Conditional Probability of Humidity, Outlook, and Wind		
Humidity (X_2)	**$P(X_1 \mid Y = no)$**	**$P(X_1 \mid Y = yes)$**
high	2/5	2/9
low	1/5	4/9
med	2/5	3/9
Outlook (X_3)	**$P(X_1 \mid Y = no)$**	**$P(X_1 \mid Y = yes)$**
overcast	0/5	4/9
Rain	2/5	3/9
sunny	3/5	2/9
Wind (X_4)	**$P(X_1 \mid Y = no)$**	**$P(X_1 \mid Y = yes)$**
false	2/5	6/9
true	3/5	3/9

Test record

Table 4.7 Test Record

No.	Temperature X_1	Humidity X_2	Outlook X_3	Wind X_4	Play (Class Label) Y
Unlabeled Test	high	low	sunny	false	?

Calculation of posterior probability P(Y/X)

$$P(Y = yes|X) = \frac{P(Y) * \prod_{i=1}^{n} P(Xi|Y)}{P(X)}$$

$= P(Y = yes) * \{P(Temp = high|Y = yes) * P(Humidity = low|Y = yes) * P(Outlook = sunny|\ Y = yes) * P(Wind = false|Y = yes)\}/P(X)$

$= 9/14 * \{2/9 * 4/9 * 2/9 * 6/9\}/P(X)$

$= 0.0094/P(X)$

$P(Y = no|X) = 5/14 * \{2/5 * 4/5 * 3/5 * 2/5\}$

$= 0.0274/P(X)$

We normalize both the estimates by dividing both by (0.0094 + 0.027) to get

$$\text{Likelihood of (Play = yes)} = \frac{0.0094}{0.0274 + 0.0094} = 26\%$$

$$\text{Likelihood of (Play = no)} = \frac{0.0274}{0.0274 + 0.0094} = 74\%$$

Issues

- Incomplete training set -> Use laplace correction
- Continuous numeric attributes -> Use Probability density function

Numerical Predictors

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Mean

$$\sigma = \left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2\right]^{0.5}$$

Standard deviation

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

Normal distribution

		Humidity	*Mean*	*StDev*
Play Golf	yes	86 96 80 65 70 80 70 90 75	79.1	10.2
	no	85 90 70 95 91	86.2	9.7

$$P(\text{humidity} = 74 \mid \text{play} = \text{yes}) = \frac{1}{\sqrt{2\pi}(10.2)} e^{-\frac{(74-79.1)^2}{2(10.2)^2}} = 0.0344$$

$$P(\text{humidity} = 74 \mid \text{play} = \text{no}) = \frac{1}{\sqrt{2\pi}(9.7)} e^{-\frac{(74-86.2)^2}{2(9.7)^2}} = 0.0187$$

More example NB

Outlook	Temp	Humidity	Windy	Play
Rainy	Cool	High	True	?

$$P(Yes \mid X) = P(Rainy \mid Yes) \times P(Cool \mid Yes) \times P(High \mid Yes) \times P(True \mid Yes) \times P(Yes)$$

$$P(Yes \mid X) = 2/9 \times 3/9 \times 3/9 \times 3/9 \times 9/14 = 0.00529 \longrightarrow 0.2 = \frac{0.00529}{0.02057 + 0.00529}$$

$$P(No \mid X) = P(Rainy \mid No) \times P(Cool \mid No) \times P(High \mid No) \times P(True \mid No) \times P(No)$$

$$P(No \mid X) = 3/5 \times 1/5 \times 4/5 \times 3/5 \times 5/14 = 0.02057 \longrightarrow 0.8 = \frac{0.02057}{0.02057 + 0.00529}$$

Example of Naïve Bayes Classifier

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

A: attributes

M: mammals

N: non-mammals

$$P(A|M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(A|N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(A|M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

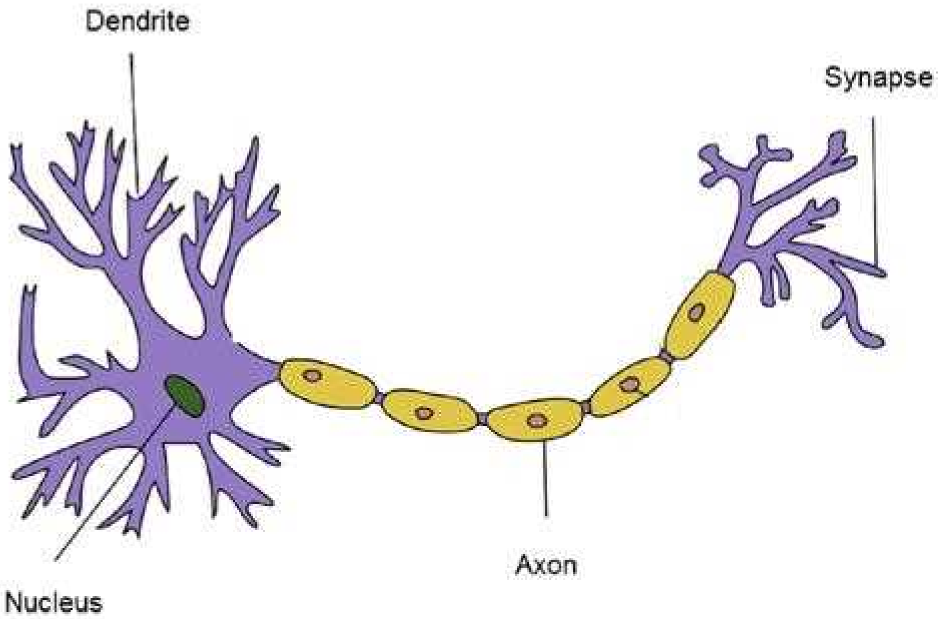
$$P(A|N)P(N) = 0.004 \times \frac{13}{20} = 0.0027$$

P(A|M)P(M) > P(A|N)P(N)

=> Mammals

NEURAL NETWORKS

Neurons – biological neuron



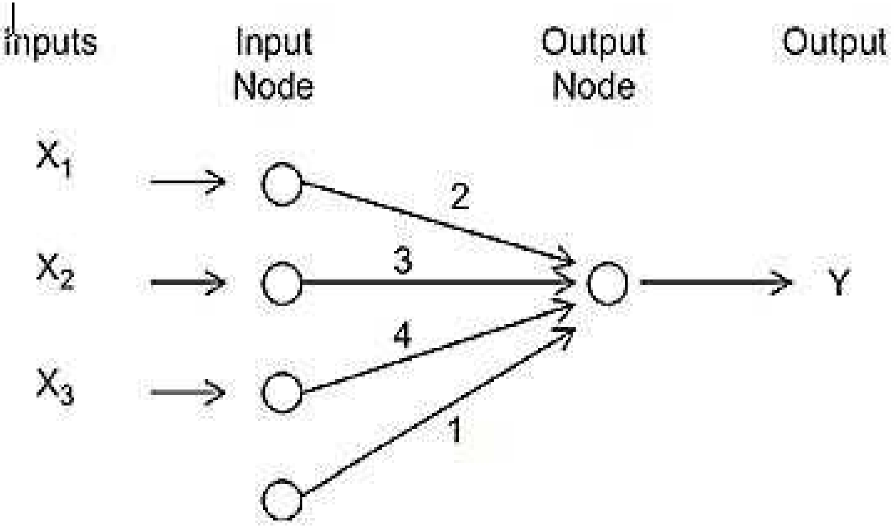
Artificial Neural Networks (ANN)

- **Basic Idea:** A complex non-linear function can be learned as a composition of simple processing units

- ANN is a collection of simple processing units (nodes) that are connected by directed links (edges)
  - Every node receives signals from incoming edges, performs computations, and transmits signals to outgoing edges
  - Analogous to *human brain* where nodes are neurons and signals are electrical impulses
  - Weight of an edge determines the strength of connection between the nodes

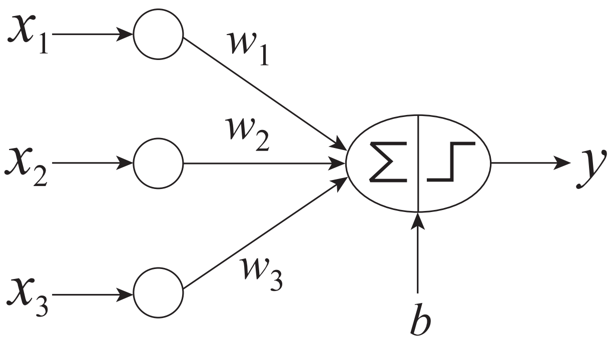
- Simplest ANN: **Perceptron** (single neuron)

Model



Y = 1 + 2X1 + 3X2 + 4X3

Basic Architecture of Perceptron



$$y = \begin{cases} 1, & \text{if } \mathbf{w}^T\mathbf{x} + b > 0. \\ -1, & \text{otherwise.} \end{cases}$$

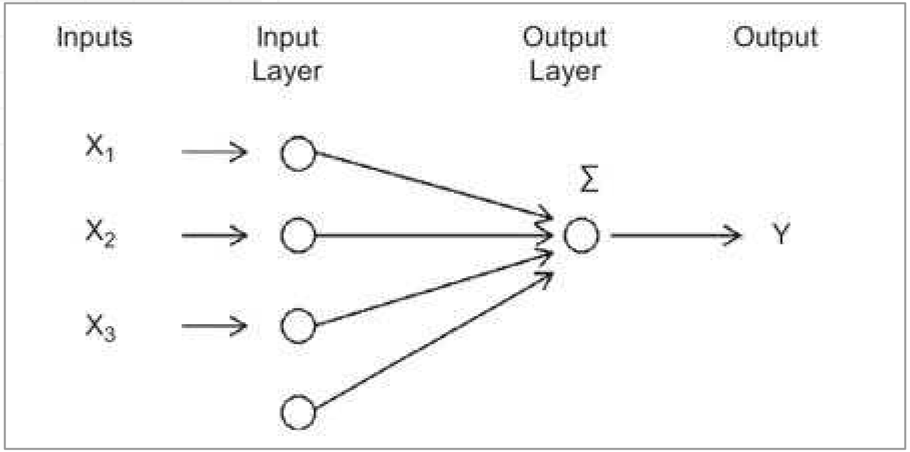
$$\tilde{\mathbf{w}} = (\mathbf{w}^T \;\; b)^T \qquad \tilde{\mathbf{x}} = (\mathbf{x}^T \;\; 1)^T$$

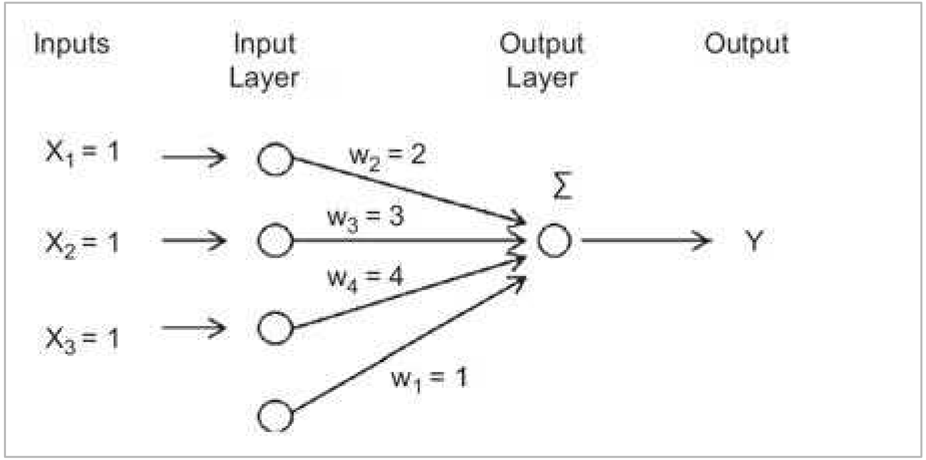
$$\hat{y} = sign(\tilde{\mathbf{w}}^T \tilde{\mathbf{x}})$$

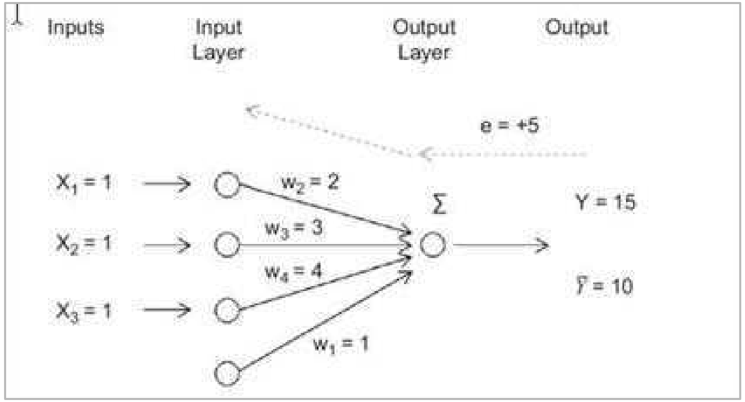
Activation Function

- Learns linear decision boundaries
- Related to logistic regression (activation function is sign instead of sigmoid)

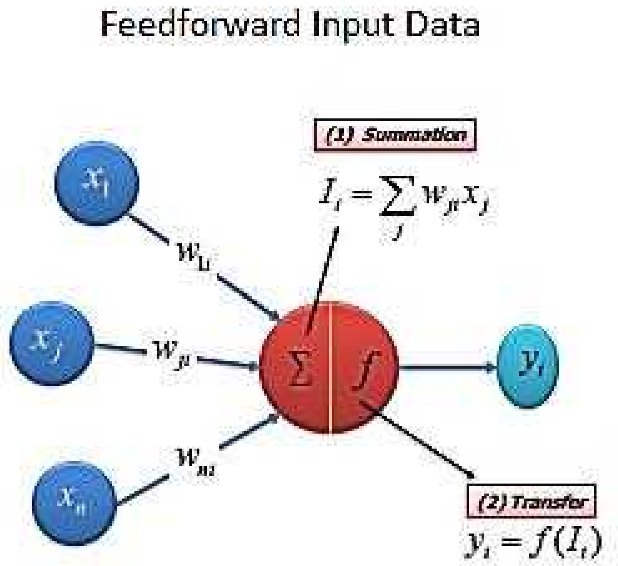
Compute output

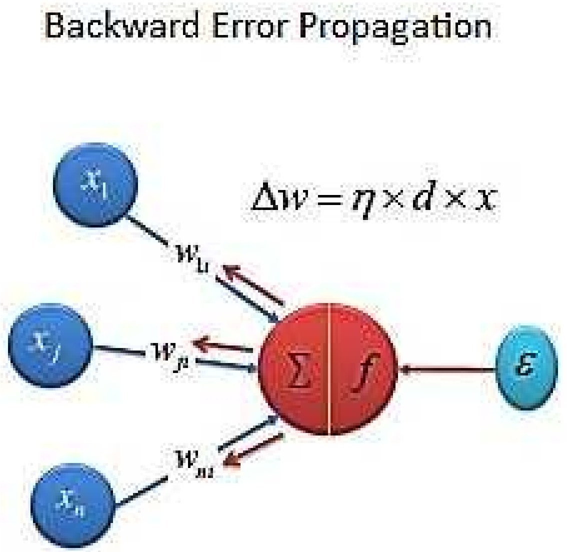






NN training



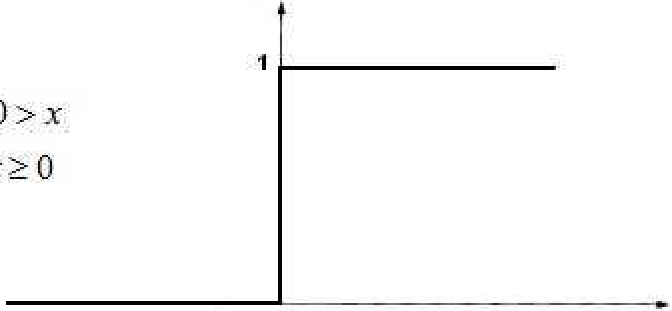


$$W_{ij} = W_{ij} + \text{delta}$$

Activation functions

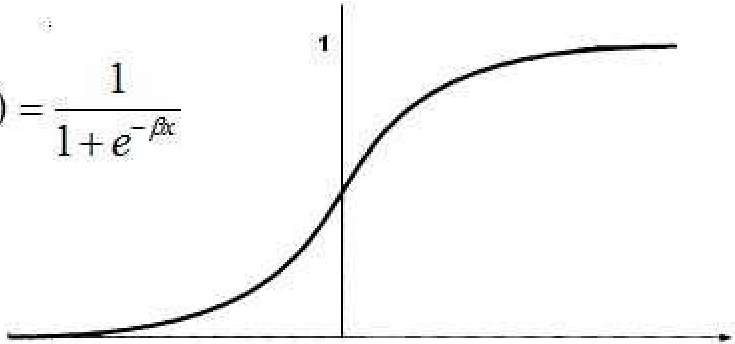
Unit step (threshold)

$$f(x)=\begin{cases}0 \text{ if } 0>x\\ 1 \text{ if } x\geq 0\end{cases}$$



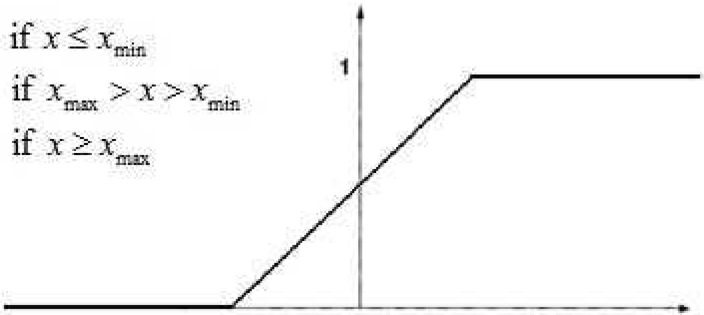
Sigmoid

$$f(x)=\frac{1}{1+e^{-\beta x}}$$



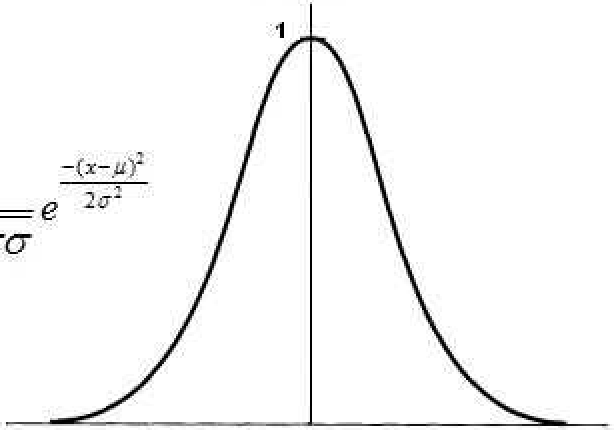
Piecewise Linear

$$f(x)=\begin{cases}0 & \text{if } x\leq x_{\min}\\ mx+b & \text{if } x_{\max}>x>x_{\min}\\ 1 & \text{if } x\geq x_{\max}\end{cases}$$

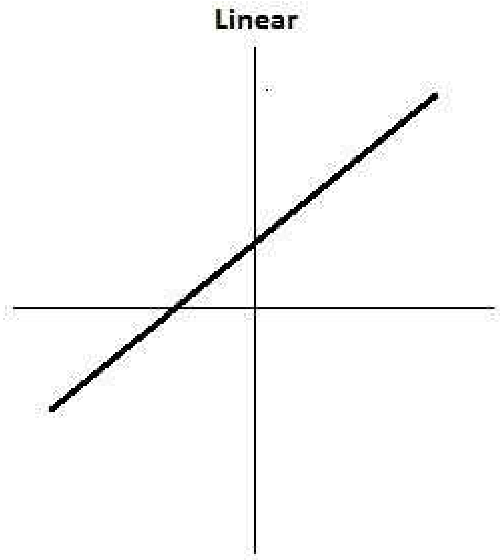


Gaussian

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



Linear activation function



NN training algorithm

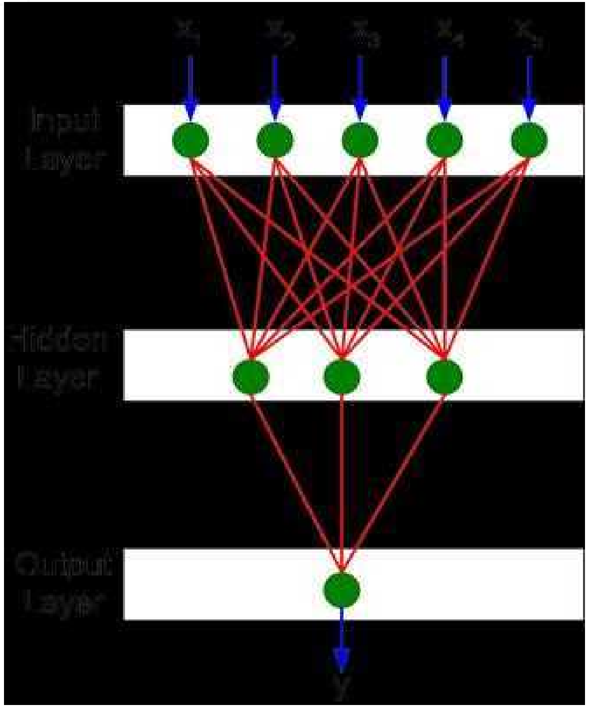
- Initialize the weights (w_0, w_1, …, w_d) randomly
- Repeat
  - For each training example (x_i, y_i)
    - Compute $\widehat{y}_i$
    - Update the weights:

$$w_j^{(k+1)} = w_j^{(k)} + \lambda\big(y_i - \hat{y}_i^{(k)}\big)x_{ij}$$

- Until stopping condition is met

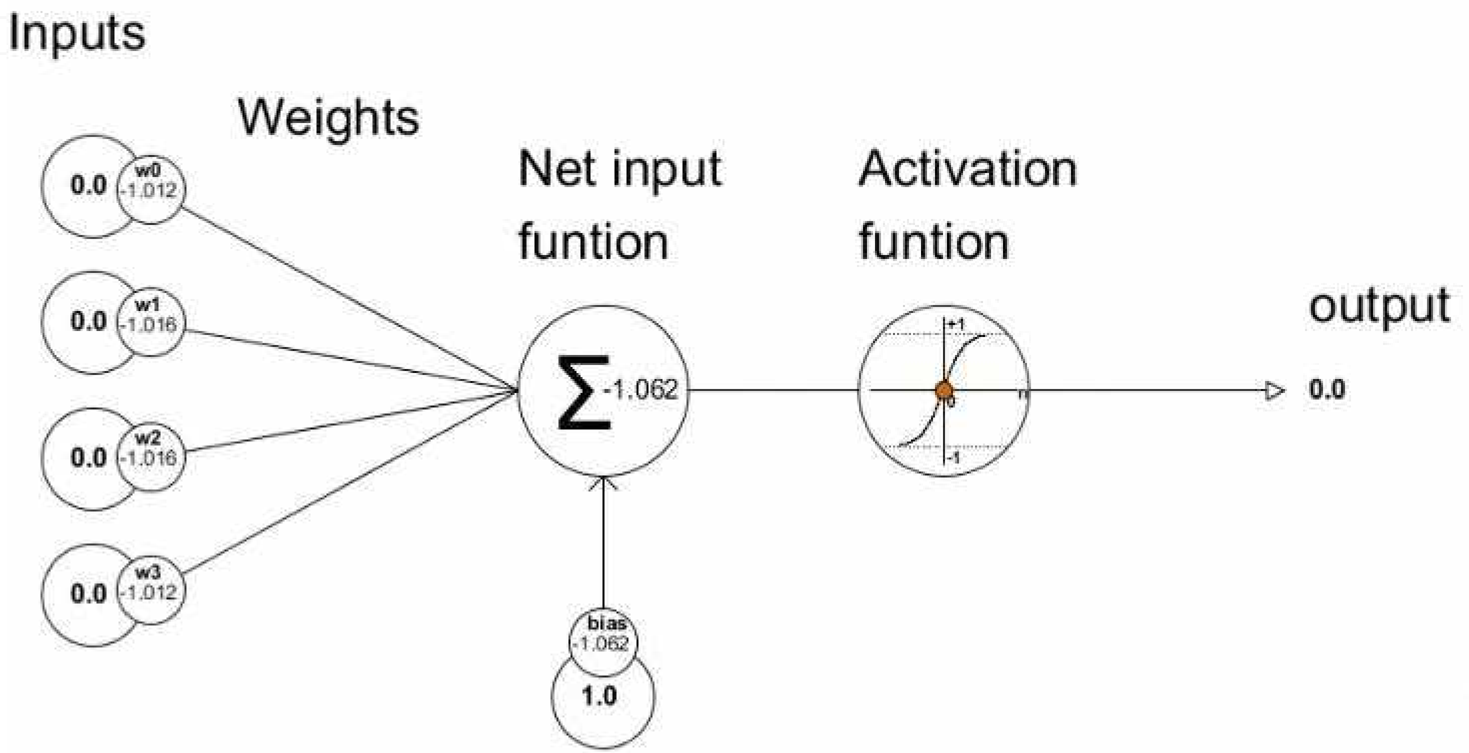
- k: iteration number; λ: learning rate

Multi-layer Neural Network



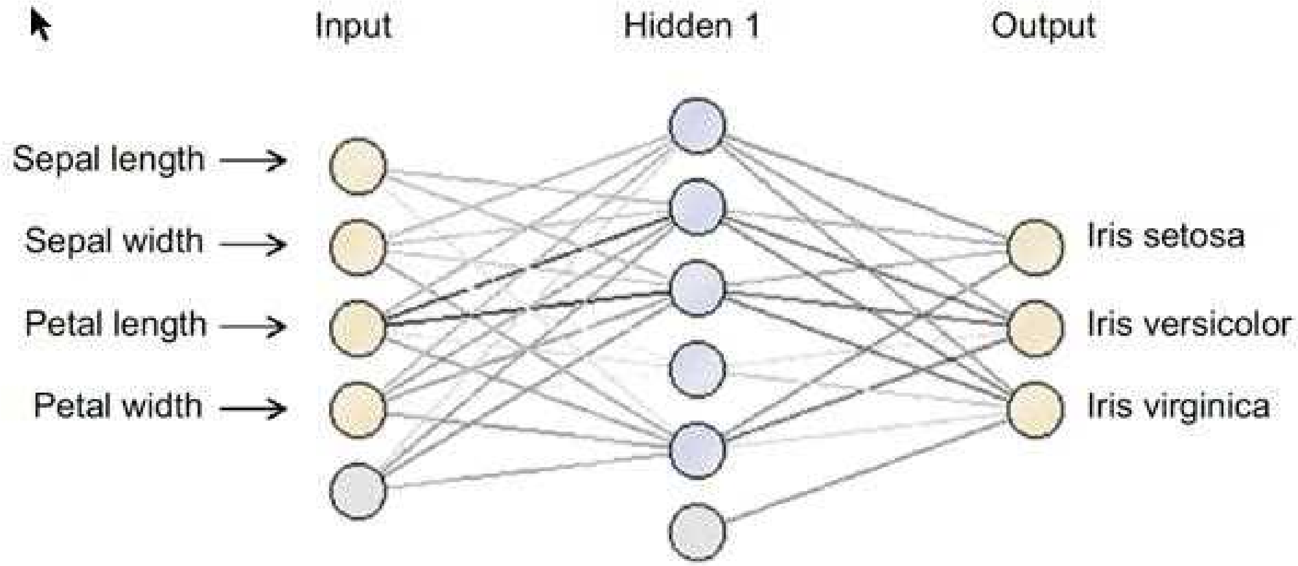
- More than one *hidden layer* of computing nodes
- Every node in a hidden layer operates on activations from preceding layer and transmits activations forward to nodes of next layer
- Also referred to as “feedforward neural networks”

NN training



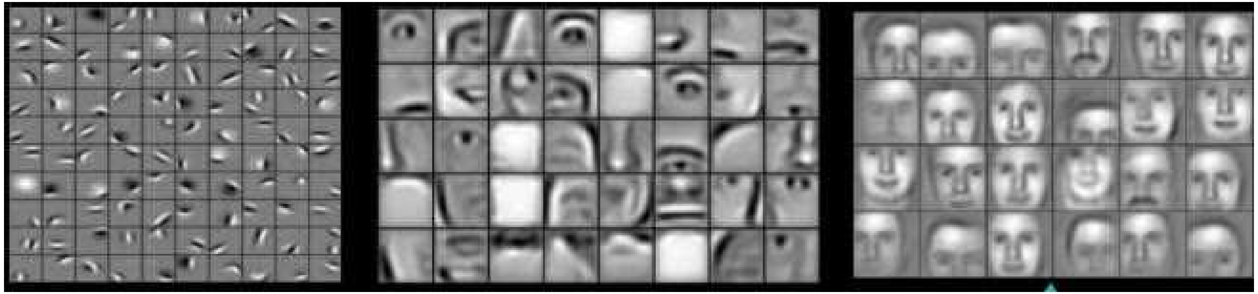
Inputs
Weights
Net input
funtion
Activation
funtion
output
0.0
w0
-1.012
0.0
w1
-1.016
0.0
w2
-1.016
0.0
w3
-1.012
Σ -1.062
bias
-1.062
1.0
+1
0
-1
0.0

Multilayer NN for Iris dataset



Why Multiple Hidden Layers?

- Activations at hidden layers can be viewed as features extracted as functions of inputs
- Every hidden layer represents a level of abstraction
  - *Complex features are compositions of simpler features*



- Number of layers is known as **depth** of ANN
  - *Deeper networks express complex hierarchy of features*

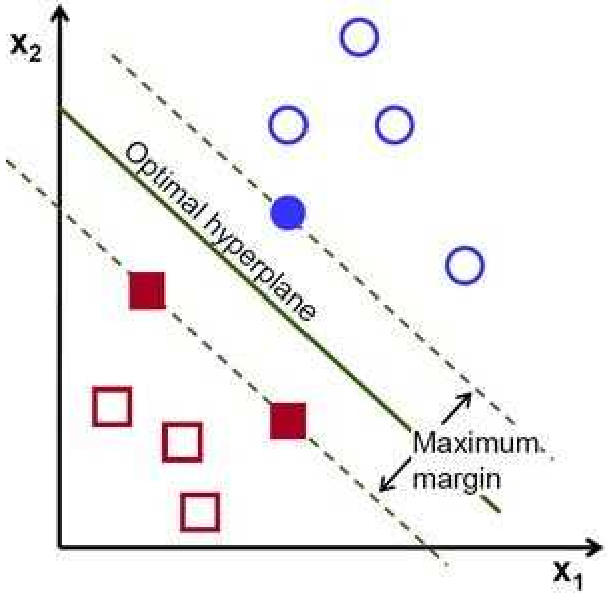
Design Issues in ANN

- Number of nodes in input layer
  - One input node per **binary/continuous** attribute
  - k or $\log_2$ k nodes for each **categorical** attribute with k values
- Number of nodes in output layer
  - One output for binary class problem
  - k or $\log_2$ k nodes for k-class problem
- Number of hidden layers and nodes per layer
- Initial weights and biases
- Learning rate, max. number of epochs, mini-batch size for mini-batch SGD, …

Deep Learning Trends

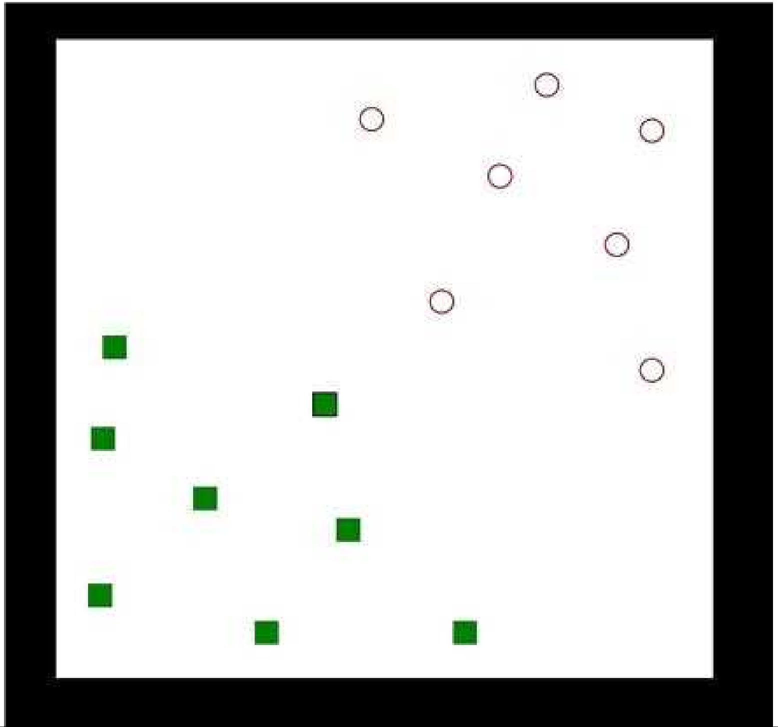
- Training **deep** neural networks (more than 5-10 layers) could only be possible in recent times with:
  - Faster computing resources (GPU)
  - Larger labeled training sets
- Algorithmic Improvements in Deep Learning
  - Responsive activation functions (e.g., RELU)
  - Regularization (e.g., Dropout)
  - Supervised pre-training
  - Unsupervised pre-training (auto-encoders)
- Specialized ANN Architectures:
  - Convolutional Neural Networks (for image data)
  - Recurrent Neural Networks (for sequence data)
- Generative Models: Generative Adversarial Networks GANs

SUPPORT VECTOR MACHINES



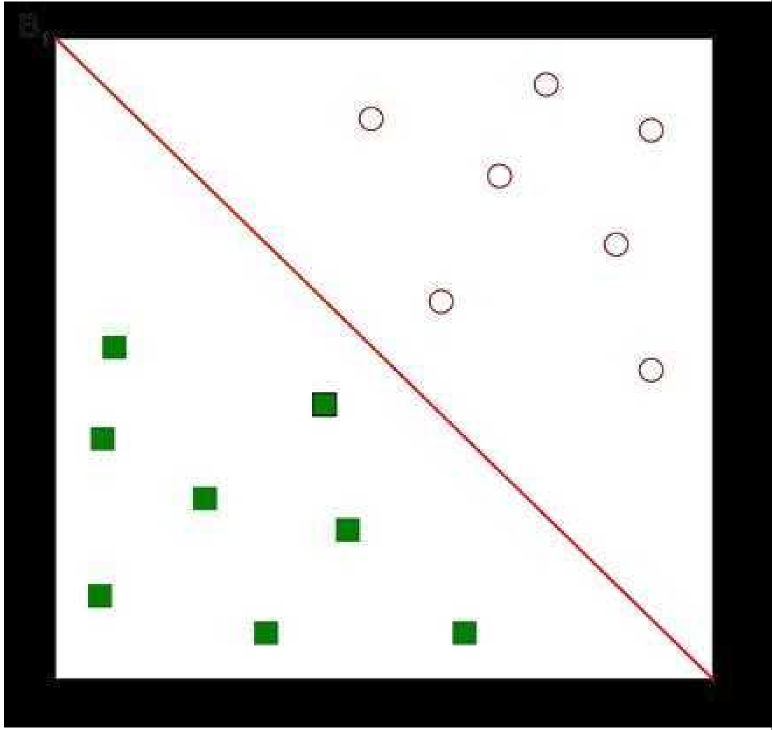
Support Vector Machines

- Find a linear hyperplane (decision boundary) that will separate the data



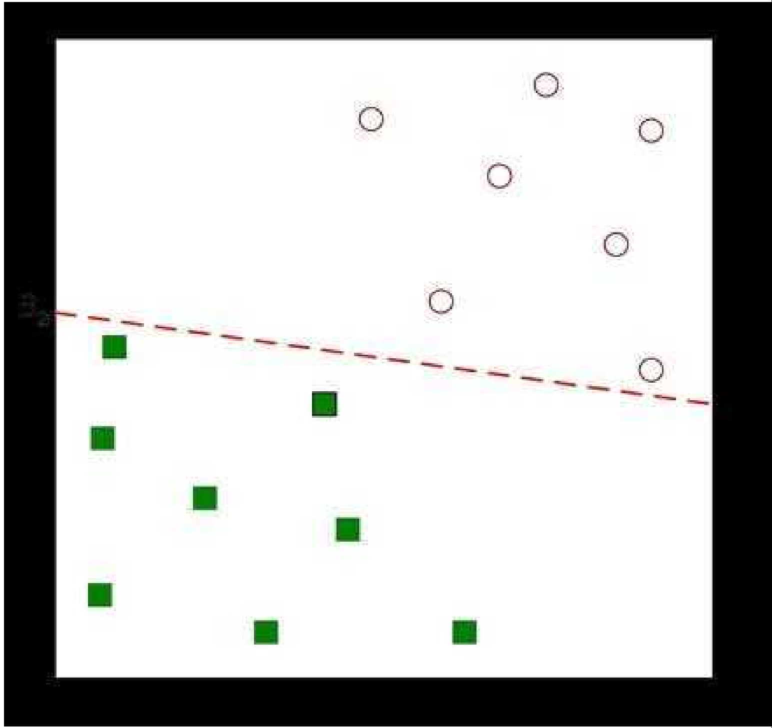
Support Vector Machines

- One Possible Solution



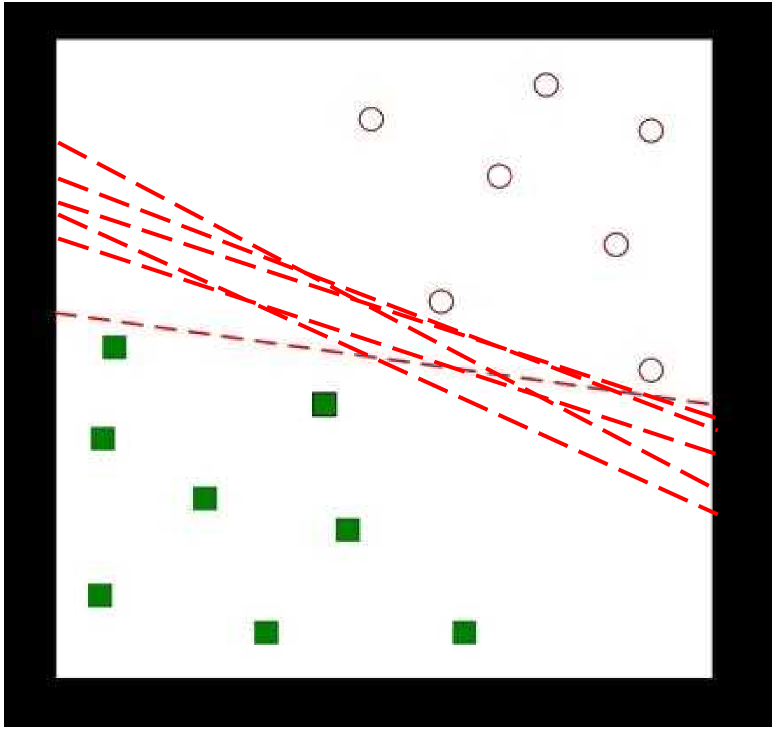
Support Vector Machines

- Another possible solution



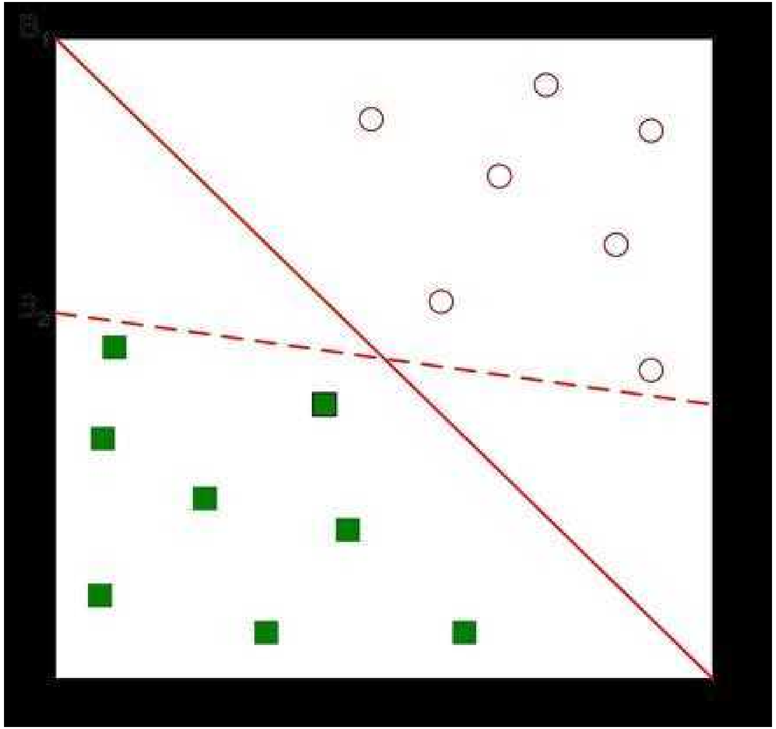
Support Vector Machines

- Other possible solutions



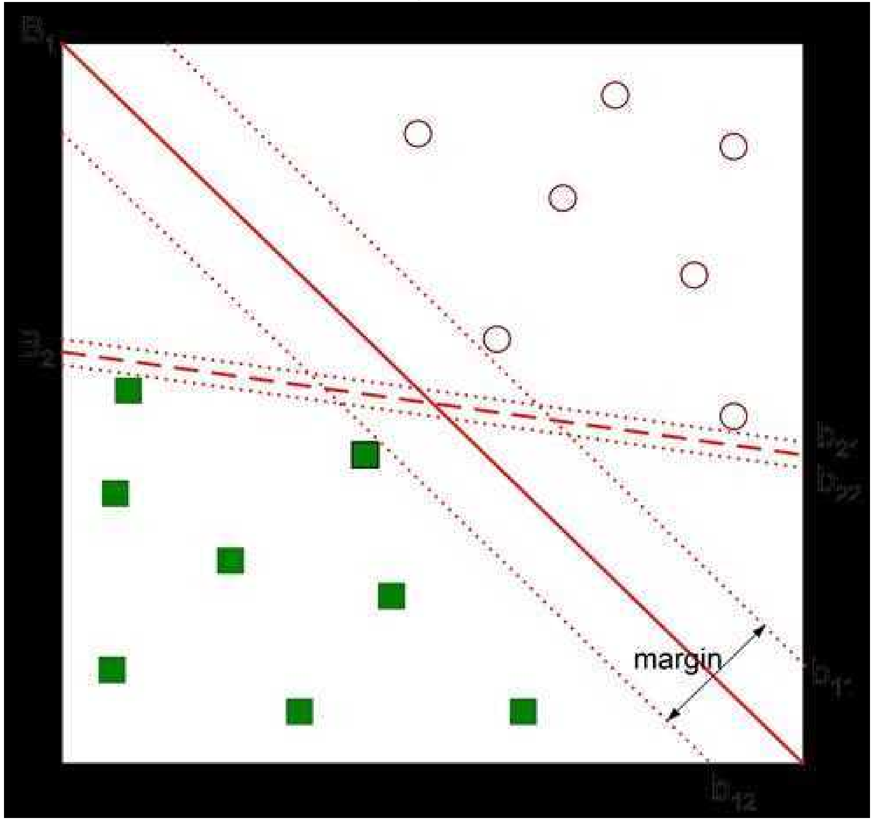
Support Vector Machines

- Which one is better? B1 or B2?
- How do you define better?



Support Vector Machines

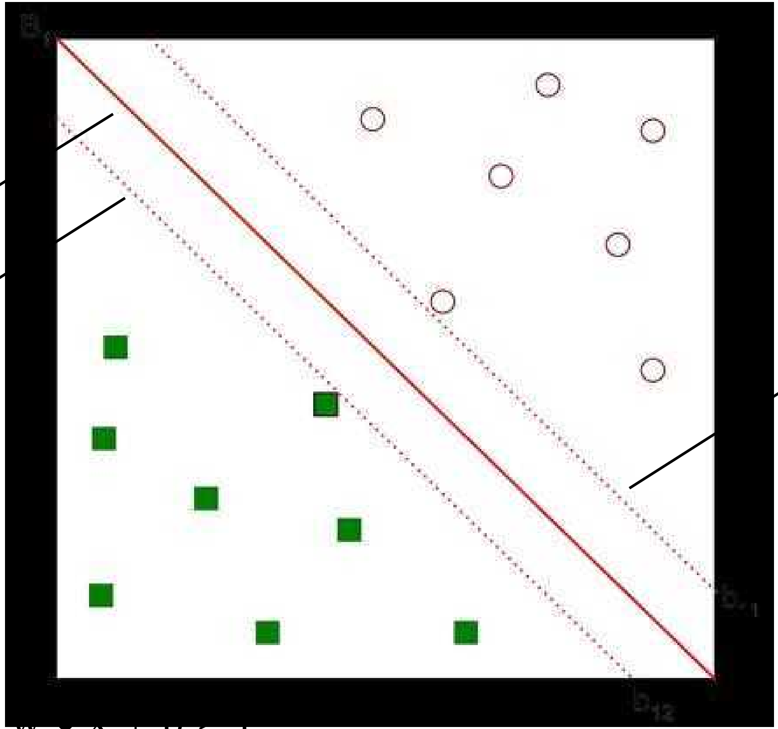
- Find hyperplane maximizes the margin => B1 is better than B2



Support Vector Machines

$$\vec{w}\bullet\vec{x}+b=0$$

$$\vec{w}\bullet\vec{x}+b=-1$$



$$\vec{w}\bullet\vec{x}+b=+1$$

$$f(\vec{x})=\begin{cases}1 & \text{if } \vec{w}\bullet\vec{x}+b\ge 1\\ -1 & \text{if } \vec{w}\bullet\vec{x}+b\le -1\end{cases}$$

$$\text{Margin}=\frac{2}{\|\vec{w}\|}$$

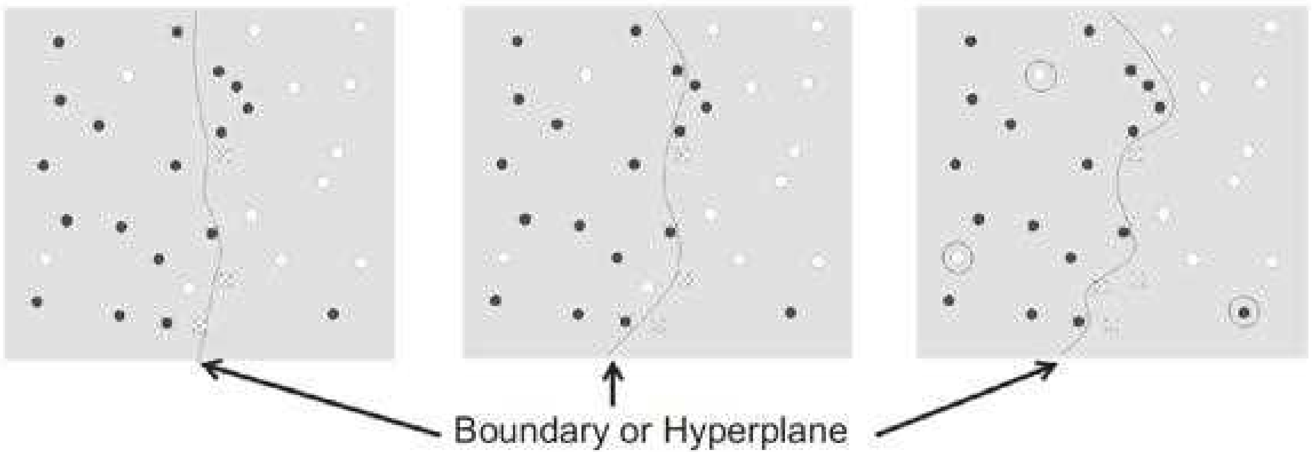
Linear SVM

- Linear model:

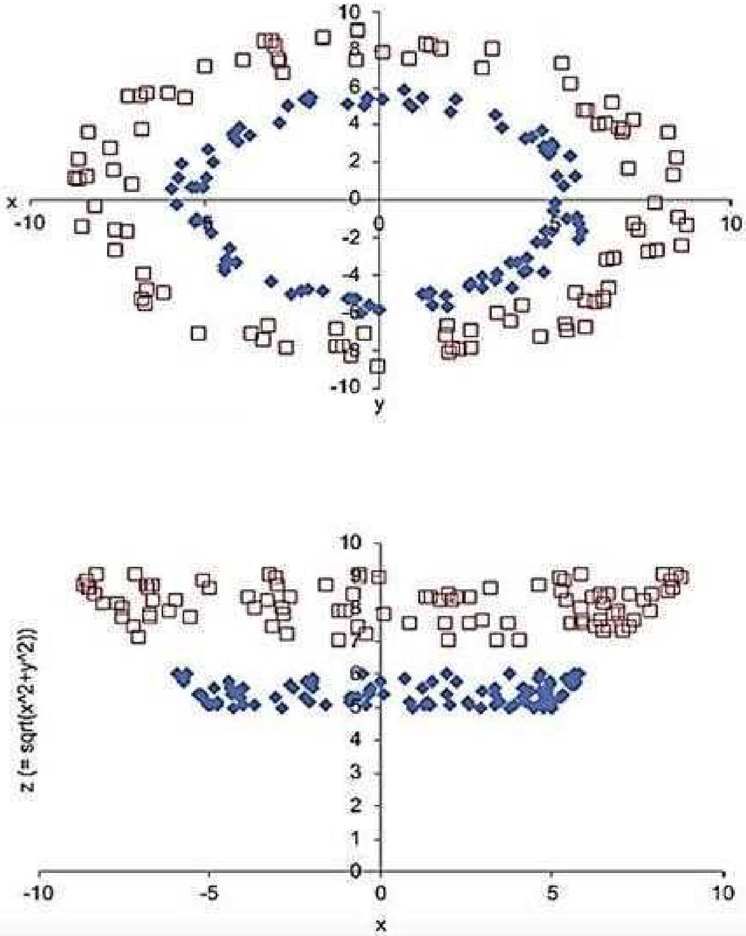
$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

- Learning the model is equivalent to determining the values of $\vec{w}$ and b
  - Find from training data

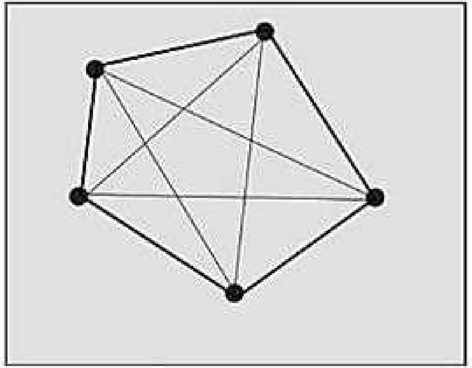
Boundary

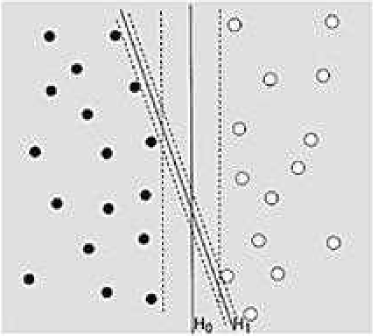


Transforming linearly non-separable data



Optimal hyperplane





SVM kernels

Linear SVM $\quad x_i \cdot x_j$

Non-linear SVM $\quad \phi(x_i) \cdot \phi(x_j)$

Kernel function $\quad k(x_i \cdot x_j)$

Ensemble Learners

Ensemble model

Wisdom of the Crowd

Meta learners = sum of several base models

Reduces the model generalization error

Ensemble models