

Machine Learning (19CSE305)

Syllabus

Unit 1

Foundations of supervised learning - Decision trees and inductive bias, Regression Vs Classification, Supervised: Linear Regression, Logistic Regression, Generalisation, Training, Validation and Testing, Problem of Overfitting, Bias vs Variance, Performance metrics, Decision Tree, Random Forest, Perceptron, Beyond binary classification

Unit 2

Advanced supervised learning - Naive Bayes, Bayesian Belief Network, K-Nearest Neighbour, Support vector machines, Markov model, Hidden Markov Model, Parameter Estimation : MLE and Bayesian Estimate, Expectation Maximisation, Neural Networks

Unit 3

Unsupervised Learning : Curse of Dimensionality, Dimensionality Reduction Techniques, Principal component analysis, Linear Discriminant Analysis Clustering: K-means, Hierarchical, Spectral, subspace clustering, association rule mining. Case Study: Recommendation systems

Text Books

Tom Mitchell. Machine Learning. McGraw Hill; 2017

Reference(s)

Christopher M Bishop. Pattern Recognition and Machine Learning. Springer 2010

Richard O. Duda, Peter E. Hart, David G. Stork. Pattern Classification. Wiley, Second Edition; 2007

Kevin P. Murphy. Machine Learning, a probabilistic perspective. The MIT Press Cambridge, Massachusetts, 2012.

Introduction to Machine Learning



Machine Learning

- **Herbert Alexander Simon:**
“Learning is any process by which a system improves performance from experience.”
- “Machine Learning is concerned with computer programs that automatically improve their performance through experience.”



Herbert Simon
Turing Award 1975
Nobel Prize in Economics 1978

Why now?

- Flood of available data (especially with the advent of the Internet)
- Increasing computational power
- Growing progress in available algorithms and theory developed by researchers
- Increasing support from industries

ML Applications



The concept of learning in a ML system

- Learning = Improving with experience at some task
 - Improve over task T ,
 - With respect to performance measure, P
 - Based on experience, E .

Motivating Example Learning to Filter Spam

Example: Spam Filtering

Spam - is all email the user does not want to receive and has not asked to receive

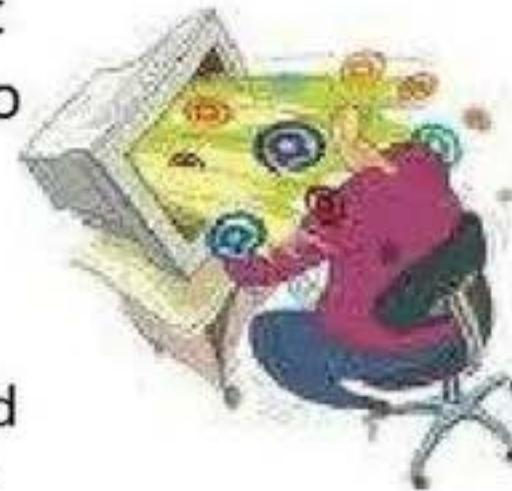
T: Identify Spam Emails

P:

% of spam emails that were filtered

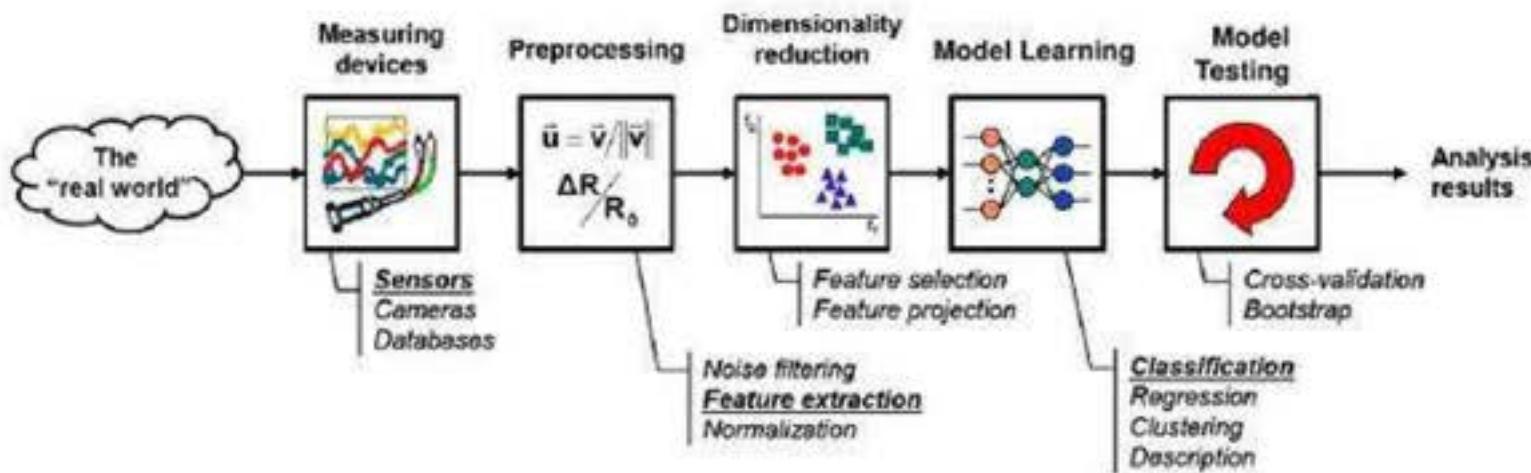
% of ham/ (non-spam) emails that were incorrectly filtered-out

E: a database of emails that were labelled by users

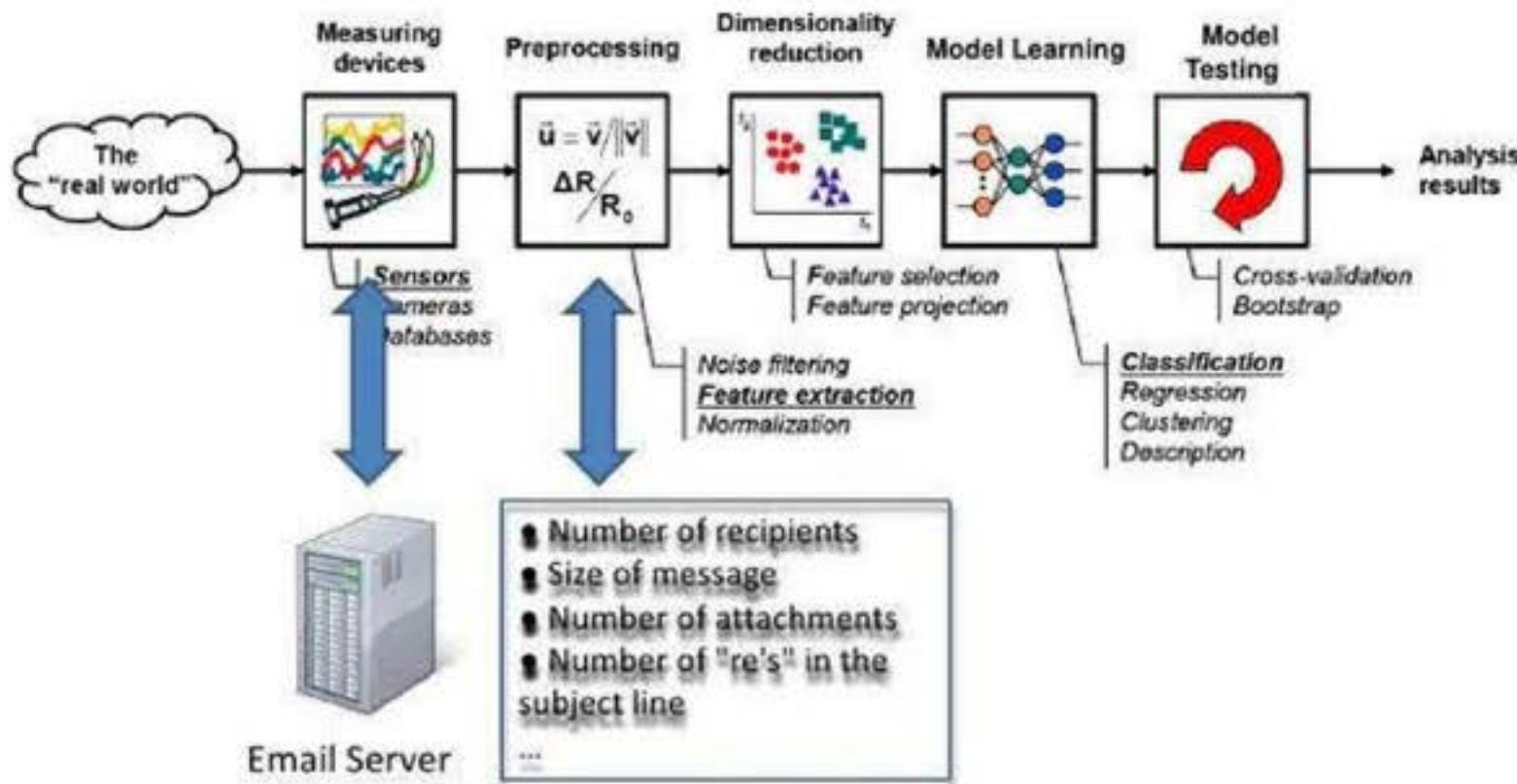




The Learning Process



The Learning Process in our Example



Data Set

Input Attributes Target Attribute

Number of new Recipients	Email Length (K)	Country (IP)	Customer Type	Email Type
0	2	Germany	Gold	Ham
1	4	Germany	Silver	Ham
5	2	Nigeria	Bronze	Spam
2	4	Russia	Bronze	Spam
3	4	Germany	Bronze	Ham
0	1	USA	Silver	Ham
4	2	USA	Silver	Spam

Instances

Numeric Nominal Ordinal

The diagram illustrates a data set with 7 instances. The first column is labeled "Instances" and contains 7 small icons representing individual data points. The last column is labeled "Target Attribute". The other four columns are labeled "Input Attributes". Below the table, arrows point from "Numeric" to the second column, "Nominal" to the third column, and "Ordinal" to the fourth column.

Face Recognition

Training examples of a person

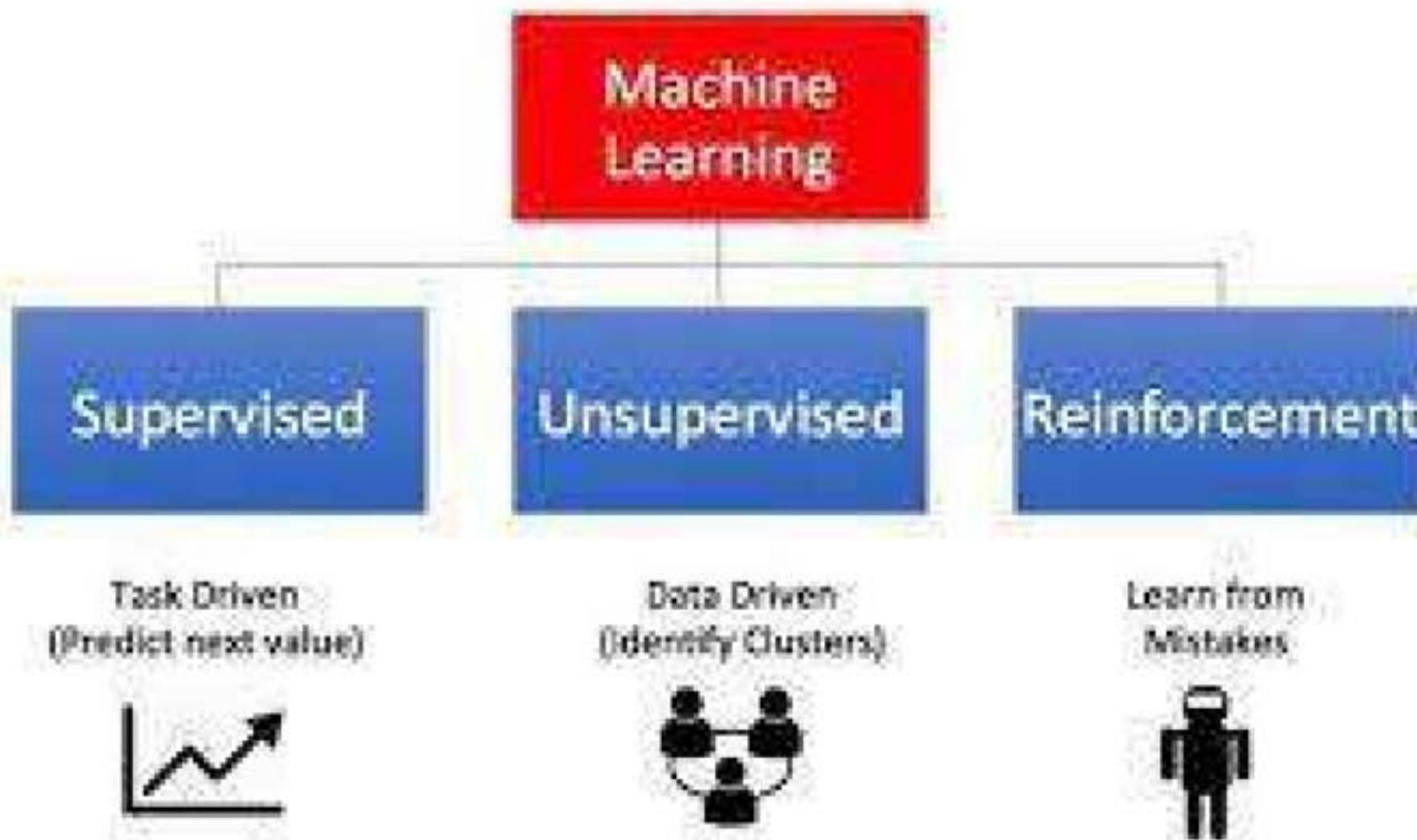


Test images



AT&T Laboratories, Cambridge UK
<http://www.uk.research.att.com/facedatabase.html>

Types of Machine Learning



Predictive/supervised learning

- Goal is to learn a mapping from inputs x to outputs y , given a labeled set of input-output pairs
 - D is the training set
 - N is the number of training examples
 - x_i is the i th input vector of M dimensions in the training set
 - Each dimension represents a feature (or attribute).
- Each y_i is a single value
- Classification
 - y_i is categorical or nominal variable from some finite set
 - {male, female}
 - {spam, not-spam}
 - {cloudy, sunny, rainy}
 - {rose,jasmine,lotus,hibiscus}
- Regression
 - y_i is numerical
 - Temperature
 - share price

Some applications of supervised learning

- Document classification



Some applications of supervised learning

- Email spam filtering



Some applications of supervised learning

- Handwriting Recognition

true class = 7



true class = 2



true class = 1



true class = 0



true class = 4



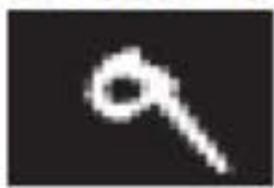
true class = 1



true class = 4



true class = 9



true class = 5

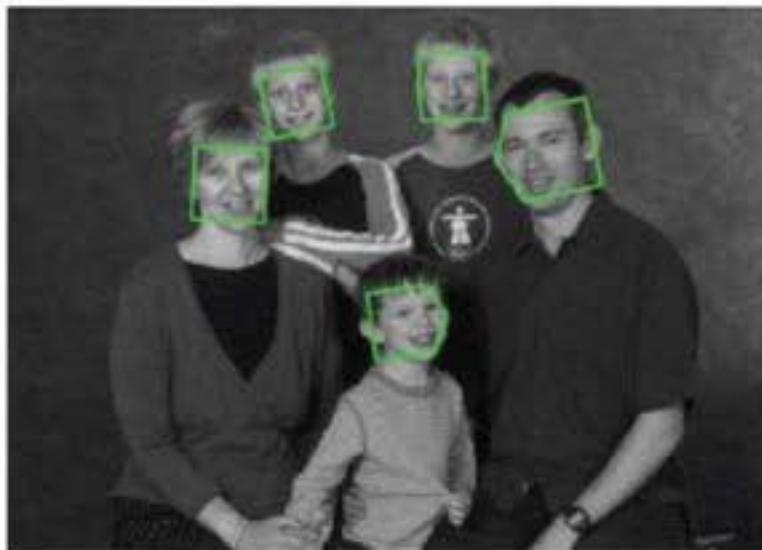


Some applications of supervised learning

- Face detection



(a)



(b)

Figure 1.6 Example of face detection. (a) Input image (Murphy family, photo taken 5 August 2010). Used with kind permission of Bernard Diedrich of Sherwood Studios. (b) Output of classifier, which detected 5 faces at different poses. This was produced using the online demo at <http://demo.pittpatt.com/>. The

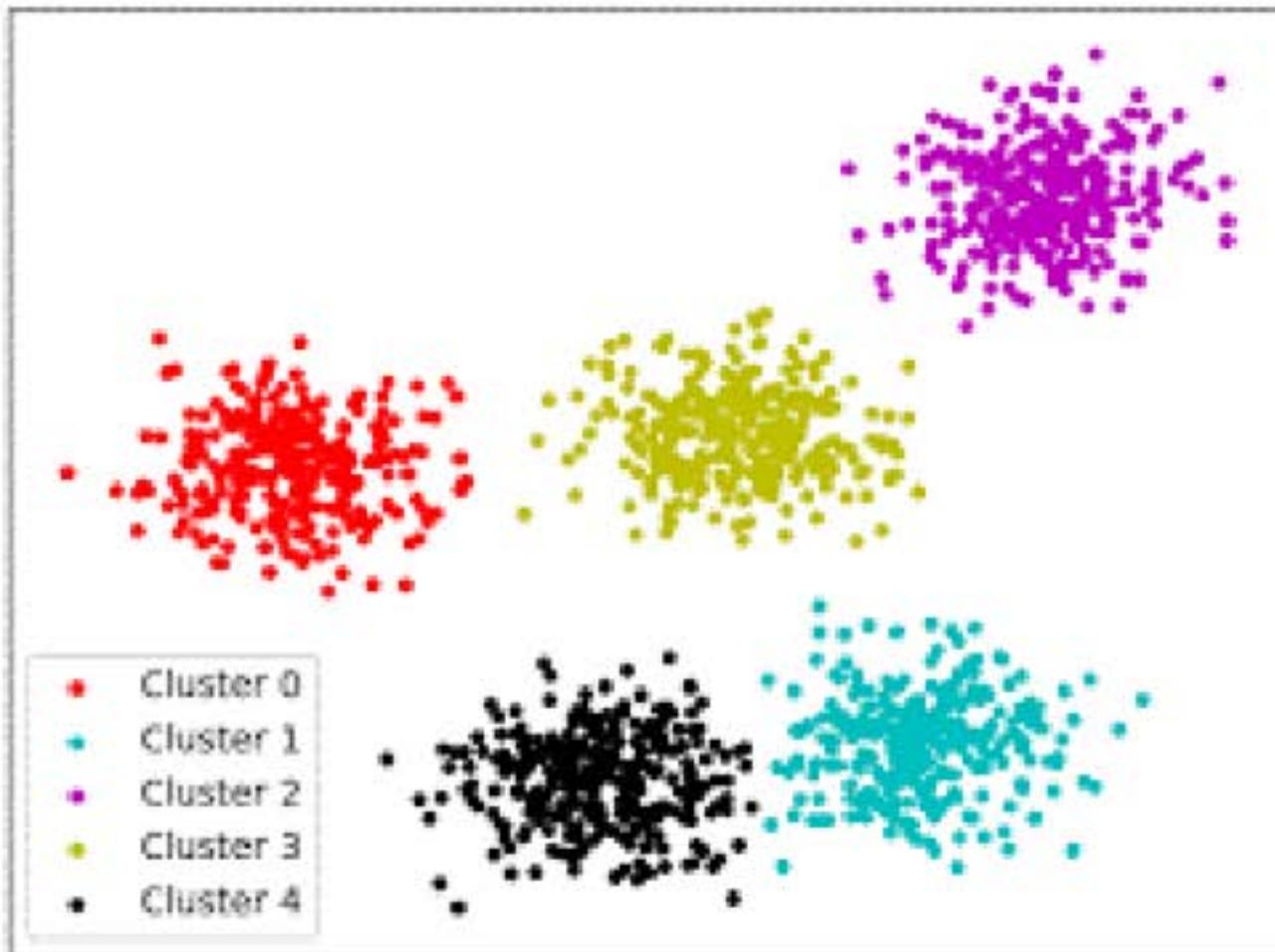
Regression - Examples

- Predict tomorrow's stock market price given current market conditions and other possible side information.
- Predict the age of a viewer watching a given video on YouTube.
- Predict the location in 3d space of a robot arm end effector, given control signals (torques) sent to its various motors.
- Predict the amount of prostate specific antigen (PSA) in the body as a function of a number of different clinical measurements.
- Predict the temperature at any location inside a building using weather data, time, door sensors, etc.

Unsupervised learning

- Training set is not available
- Knowledge discovery
- Find “interesting patterns” in the data
 - Clustering
 - Recommender Systems
 - Market basket analysis

Clustering



Find “interesting patterns”

- Recommender Systems
 - Collaborative filtering
- Market basket analysis



Recommendations



Examples:



Reinforcement learning

- Somewhere between supervised and unsupervised
- Uses occasional reward or punishment signals
- Examples
 - Playing chess
 - Robot navigation

Machine Learning (19CSE305)

Linear Algebra



Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Matrices
- Vectors
- Rank
- Invertibility

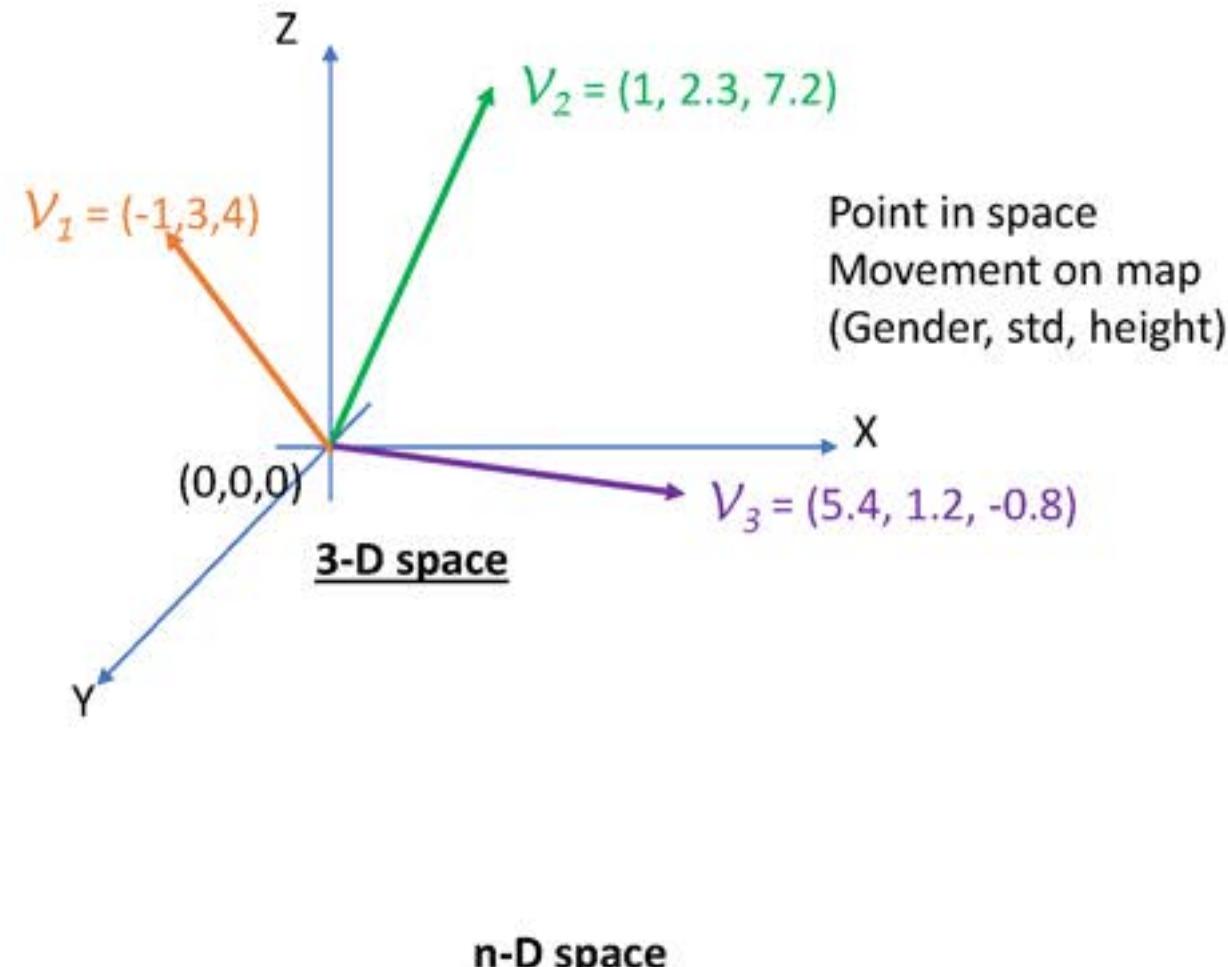
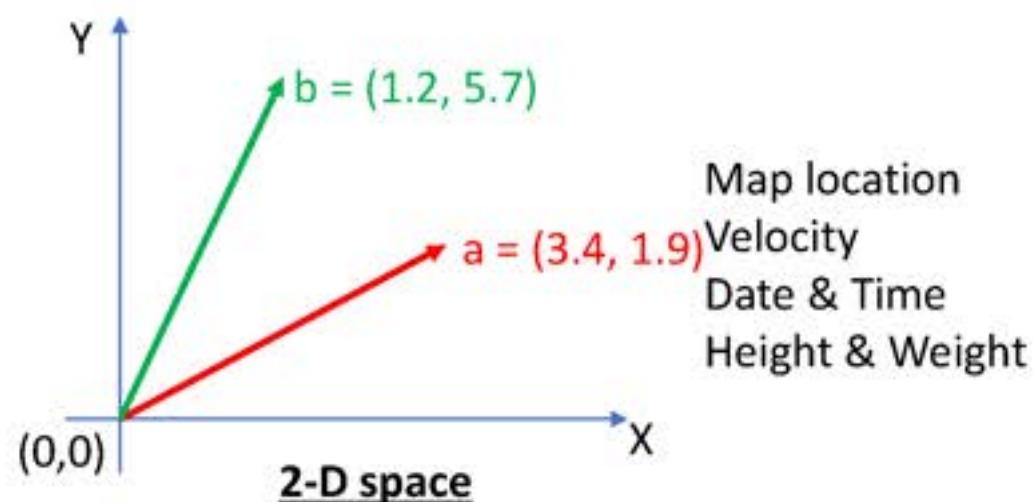
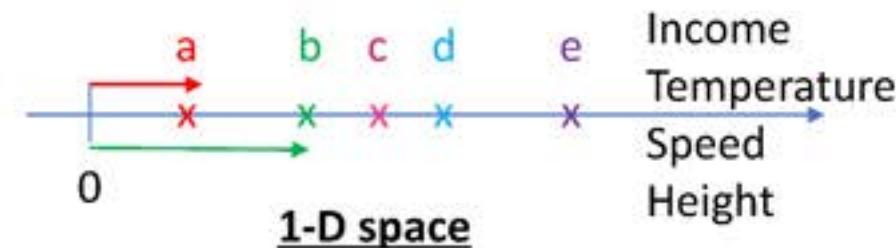
Question



A seller has 3 products to choose from and buy. We are provided with the purchase details (count of items selected and payment made) of 10 customers.

Find the cost of each product.

Vectors



Mathematical representation →
 $V = (x_1, x_2, \dots, x_n)$

Vectors

{2}, {13.7}, {x}... {z}

$$\xleftarrow{\hspace{1cm}} \mathbb{R}^1$$

{3,1.5}, {12, 5.6}, ... {x,y}

$$\xleftarrow{\hspace{1cm}} \mathbb{R}^2$$

{1,2,3}, {56, 32, 111}, ... {x,y,z}

$$\xleftarrow{\hspace{1cm}} \mathbb{R}^3$$

{ $p_1, p_2, p_3, \dots, p_n$ }, { $q_1, q_2, q_3, \dots, q_n$ }

$$\xleftarrow{\hspace{1cm}} \mathbb{R}^n$$

{1,2,3}, {56, 32, 3}, ... {x,y,3}

$$\xleftarrow{\hspace{1cm}} \mathbb{R}^?$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

A vector is usually written as a column.

Matrix

Cell: a_{ij} or $a_{i,j}$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

Row 1

Row m

Col 2

Col n

When $m = n$, the matrix is called a Square matrix.

Matrix Addition

2 matrices A & B can be added only when:

$$A, B \in \mathbb{R}^{m \times n}$$

$$A = \begin{bmatrix} 1 & 2 & 4 \\ 3 & 7 & 9 \end{bmatrix}$$

$$A + B = D \in \mathbb{R}^{m \times n}$$

$$B = \begin{bmatrix} 5 & 9 & 2 \\ 4 & 3 & 6 \end{bmatrix}$$

$$A + B = D = \begin{bmatrix} 1+5 & 2+9 & 4+2 \\ 3+4 & 7+3 & 9+6 \end{bmatrix}$$

Matrix Multiplication

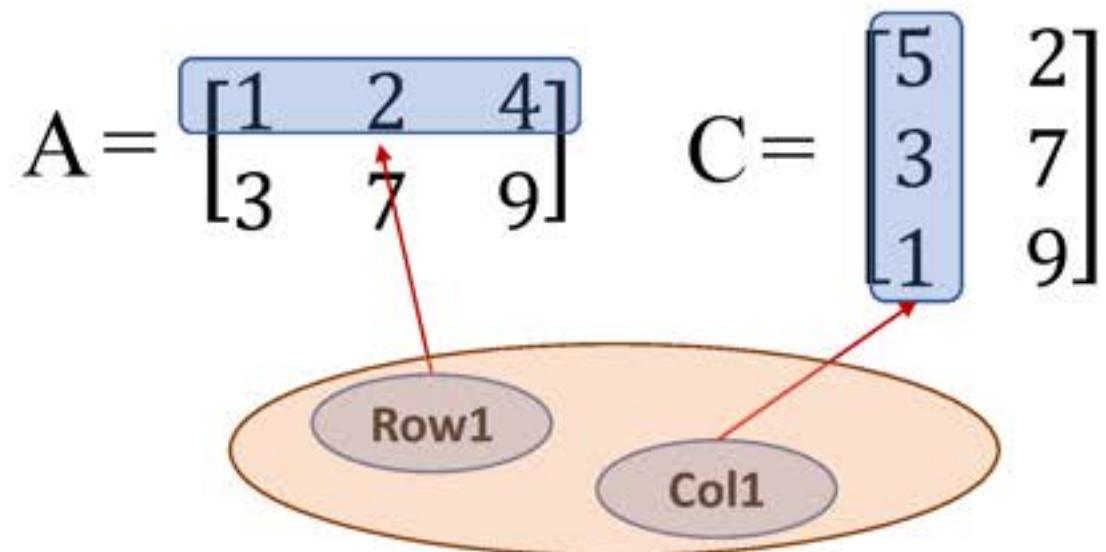
$$A \in \mathbb{R}^{m \times n}$$

$$C \in \mathbb{R}^{p \times q}$$

- A & C can be multiplied only when $n = p$
- The sequence of multiplication matters

$$AC = D \in \mathbb{R}^{m \times q}$$

- Pre-multiplication / post-multiplication
- In this example, CA is invalid
- Only when A & C are square matrices of same size ($m \times m$), both pre and post-multiplication possible



$$\begin{aligned} D_{1,1} = & A_{1,1} \times C_{1,1} \\ & + A_{1,2} \times C_{2,1} \\ & + A_{1,3} \times C_{3,1} \end{aligned}$$

$$AC = D = \begin{bmatrix} 15 & 52 \\ 3 & 7 \end{bmatrix}$$

Matrix Transpose

1

$$A^T \in \mathbb{R}^{n \times m}$$

2

$$(A^T)_{ij} = A_{ji}$$

3

$$A = \begin{bmatrix} 1 & 2 & 4 \\ 3 & 7 & 9 \end{bmatrix}$$

4

$$(A^T)^T = A$$

$$(AB)^T = B^T A^T$$

$$(A + B)^T = A^T + B^T$$

5

 $A = A^T \Rightarrow A$ is symmetric

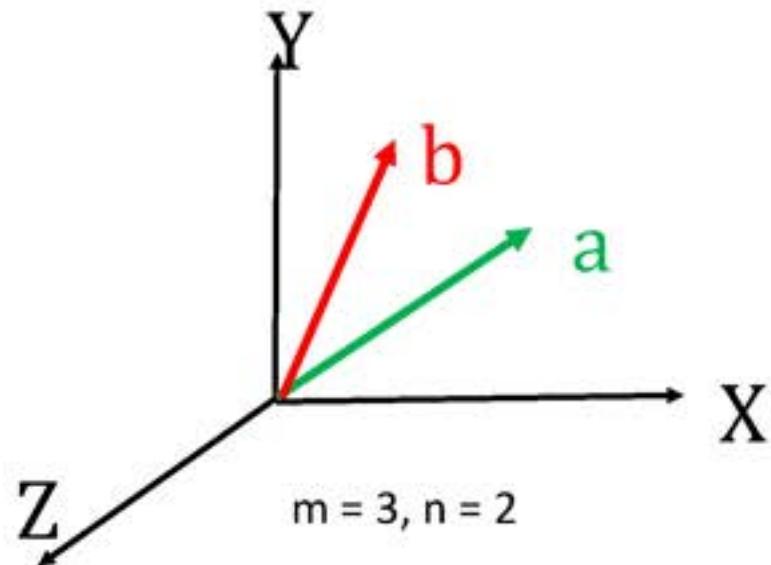
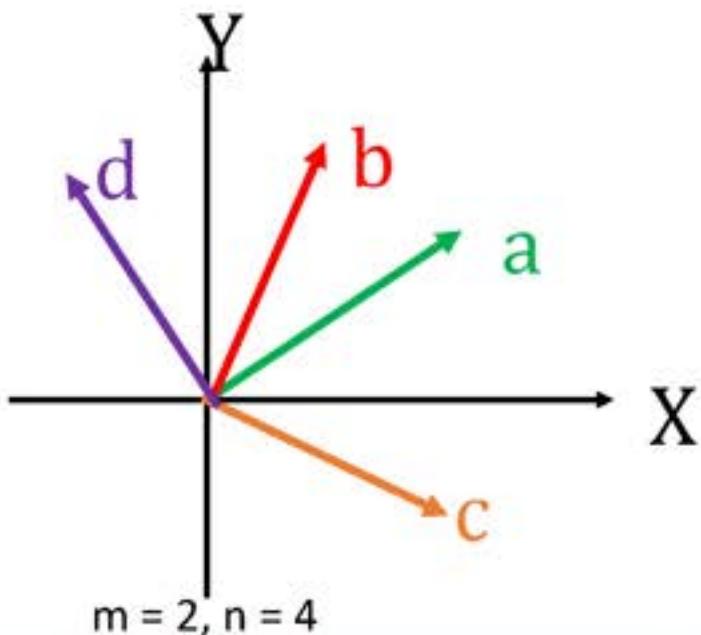
$$A^T = \begin{bmatrix} 1 & 3 \\ 2 & 7 \\ 4 & 9 \end{bmatrix}$$

Rank of a Matrix

- Rank of a matrix is the number of independent vectors present in the matrix
- Its also the number of non-zero rows present in a row-echelon matrix
- It indicates the dimensionality of the space spanned by the vectors in the matrix
- When the $\text{rank}(A) < m$; the vectors present don't span the \mathbb{R}^m . This indicates the vectors cover a subspace.
- $\text{rank}(A) < m$, the data set is a good candidate for dimensionality reduction (to be covered later).

$$A \in \mathbb{R}^{m \times n}$$

$$\text{rank}(A) \leq \min(m, n)$$



Matrix Determinant

Determinants

$$\det A = |A| = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

($\because R^{2 \times 2}$ case)

$$\det(A) = |A| = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11} \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12} \begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13} \begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$

What happens to determinant when $\text{rank}(A) < m$?

What is a singular matrix?

$$\det = 0$$

Matrix Inversion

Inverse of a number is one when multiplied to the number, the product becomes 1. It's also called reciprocal.

$$n \times (1/n) = 1$$

Similarly, for a square matrix A:

$$A^{-1}A = A A^{-1} = I$$

$$I_n = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

$$A^{-1} = \frac{1}{\det(A)} adj(A)$$

Linear Equations & Matrices

$$x + 2y + 4z = 33$$

$$4x + 3y - z = 29$$

$$-x - y + 2z = -2$$



1	2	4	33	Row 1
4	3	-1	29	Row 2
-1	-1	2	-2	Row 3

$$\begin{bmatrix} 1 & 2 & 4 \\ 4 & 3 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ -1 \end{bmatrix}$$

$A \quad X \quad C$

$$AX = C$$

1	2	4	33	• Row2 - 4 x Row1
0	-5	-17	-103	• Row3 + Row1
0	1	6	31	• Row2 & Row3 swap • Row3 + 5 x Row2

1	2	4	33	• Row3 / 13
0	1	6	31	
0	0	13	52	

Solution with Matrix Inversion

$$A^{-1}A = A \quad A^{-1} = I$$

$$A^{-1} = \frac{1}{\det(A)} \text{adj}(A)$$

$$x + 2y + 4z = 33$$

$$4x + 3y - z = 29$$

$$-x - y + 2z = -2$$

$$\begin{bmatrix} 1 & 2 & 4 \\ 4 & 3 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ -1 \end{bmatrix}$$

$$AX = C$$

$$(A^{-1}A)X = A^{-1}C$$

$$IX = X = A^{-1}C =$$

$$\begin{bmatrix} 3 \\ 7 \\ 4 \end{bmatrix}$$

What happens to inverse when $\text{rank}(A) < m$?

What happens to inverse of a rectangular matrix?

Rectangular Matrix Inversion

$$\begin{bmatrix} 2 & 5 & 7 & 8 \\ 1 & 2 & 3 & 1 \\ 4 & 5 & 0 & 1 \end{bmatrix}$$

No inverse exists mathematically.
Engineers are smart... pseudo-inverse helps

Principal Component Analysis (PCA) is the approach.
This works on the principle of minimum error.

Singular Value Decomposition (SVD) is the algorithm.

Real life example:

Customers of a store can choose to purchase from 3 available products. We have the item count & payment data for 10 purchases made. Find the cost of each product.

Questions

- What happens when a matrix is transposed? Think from space point of view.
- What is the relationship between column and row ranks?
- What information does the rank convey about the matrix? Think from space span point of view.
- When the rank of a matrix is much lower than $\min(m,n)$, how does it impact the design of our ML systems?



Thank you !!!!!

Machine Learning (19CSE305)

Statistics & Probability



Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Agenda

- Recap of last session
- Variables
- Mean, median & mode
- Standard Deviation and variance
- Probability & MLE
- Conditional Probability
- Joint Probability

Linear Equations & Matrices

$$A^{-1}A = A \quad A^{-1} = I$$

$$A^{-1} = \frac{1}{\det(A)} \text{adj}(A)$$

$$\begin{aligned}x + 2y + 4z &= 33 \\4x + 3y - z &= 29 \\-x - y + 2z &= -2\end{aligned}$$



$$\begin{bmatrix} 1 & 2 & 4 \\ 4 & 3 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ -1 \end{bmatrix}$$



1	2	4	33	Row 1
4	3	-1	29	Row 2
-1	-1	2	-2	Row 3

$$AX = C$$

$$(A^{-1}A)X = A^{-1}C$$

$$IX = X = A^{-1}C =$$

$$\begin{bmatrix} 3 \\ 7 \\ 4 \end{bmatrix}$$

Rectangular Matrix Inversion

$$\left[\begin{array}{cccc} 2 & 5 & 7 & 8 \\ 1 & 2 & 3 & 1 \\ 4 & 5 & 0 & 1 \end{array} \right]$$

No inverse exists mathematically.
Engineers are smart... pseudo-inverse helps

Principal Component Analysis (PCA) is the approach.
This works on the principle of minimum error.

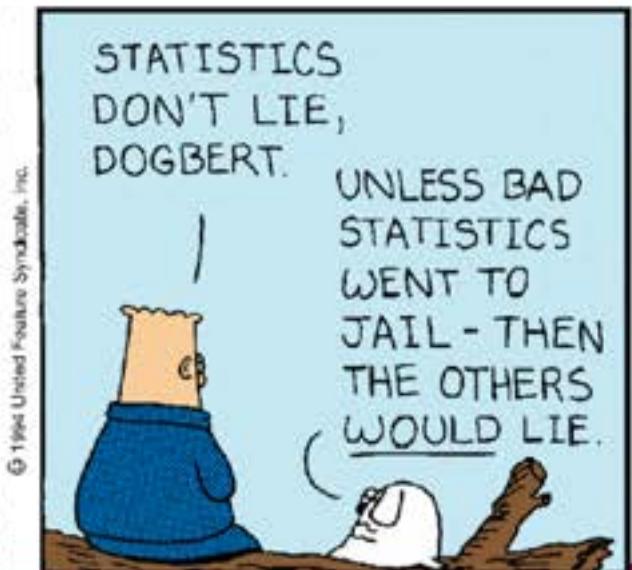
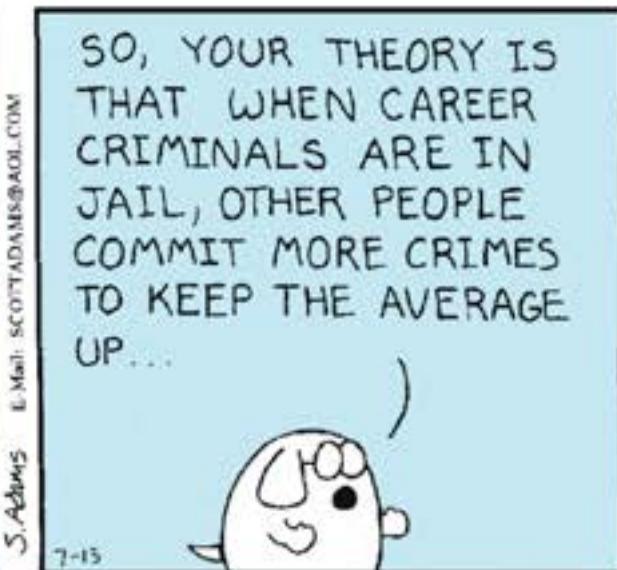
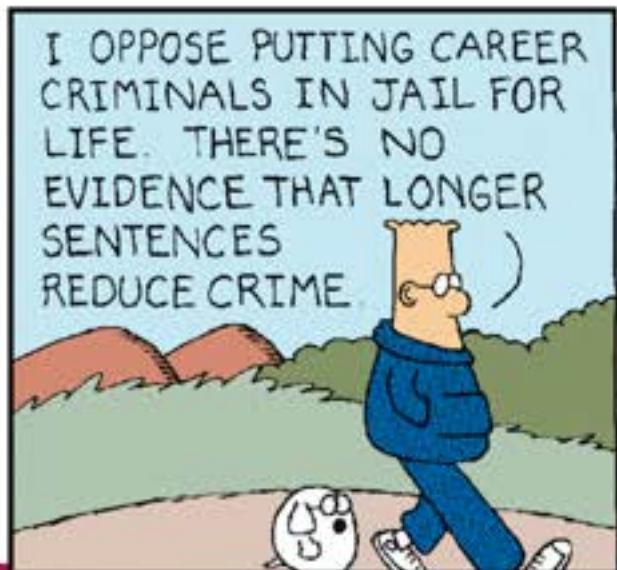
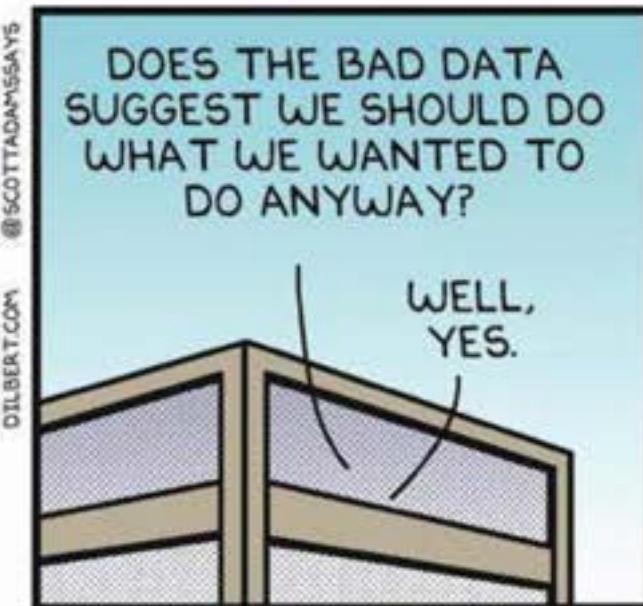
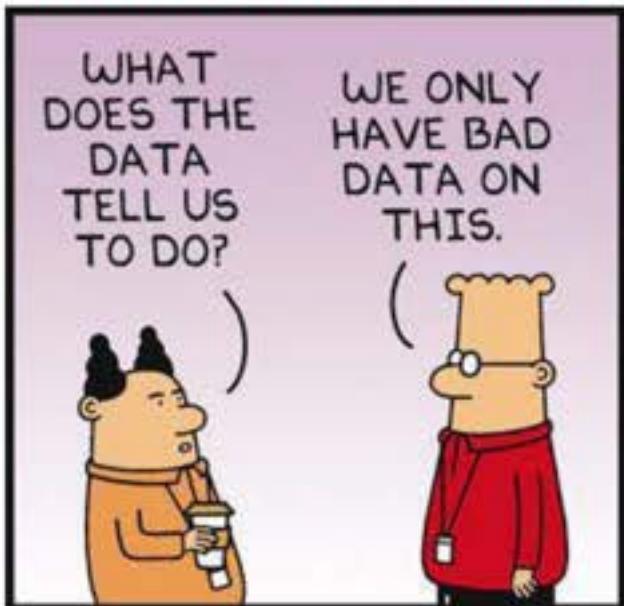
Singular Value Decomposition (SVD) is the algorithm.

Real life example:

Customers of a store can choose to purchase from 3 available products. We have the item count & payment data for 10 purchases made. Find the cost of each product.

And today we deal with lots of data...

Fun with Data



Fun with Data

**Believe me...! P value greater than
0.05 indicates chance of your
drowning is not significant.**



Probability vs Statistics



Statistics

Probability

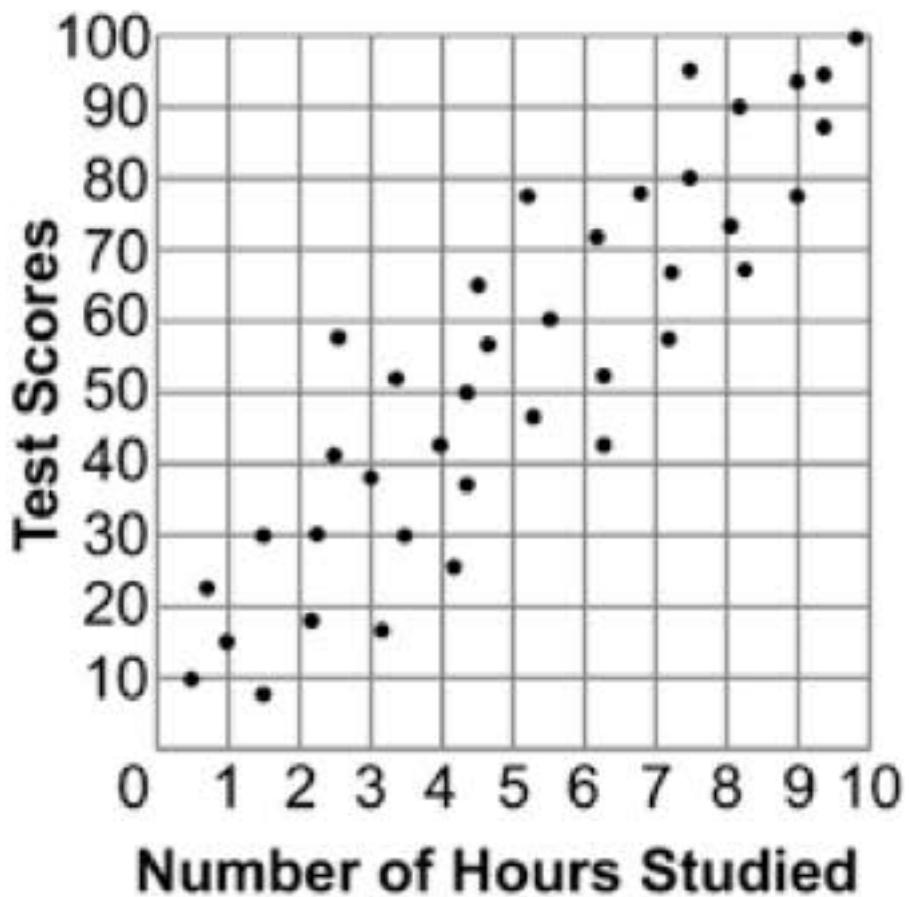
- Analyze and interpret available data
- Deals with events that has happened (past)
- Science of Data

- Science of prediction and chances (future)
- Distributions and implications
- Deals with mathematical formulas

Collection of Data

Student Height (inches)

S_1	35
S_2	36
S_3	30
S_4	31
S_5	36
S_6	35
S_7	32
S_8	35
S_9	32
S_10	34
S_11	34
S_12	34
S_13	34
S_14	32
S_15	32
S_16	34
S_17	35
S_18	34
S_19	30
S_20	36
S_21	34
S_22	31



Mean / Average & Variance

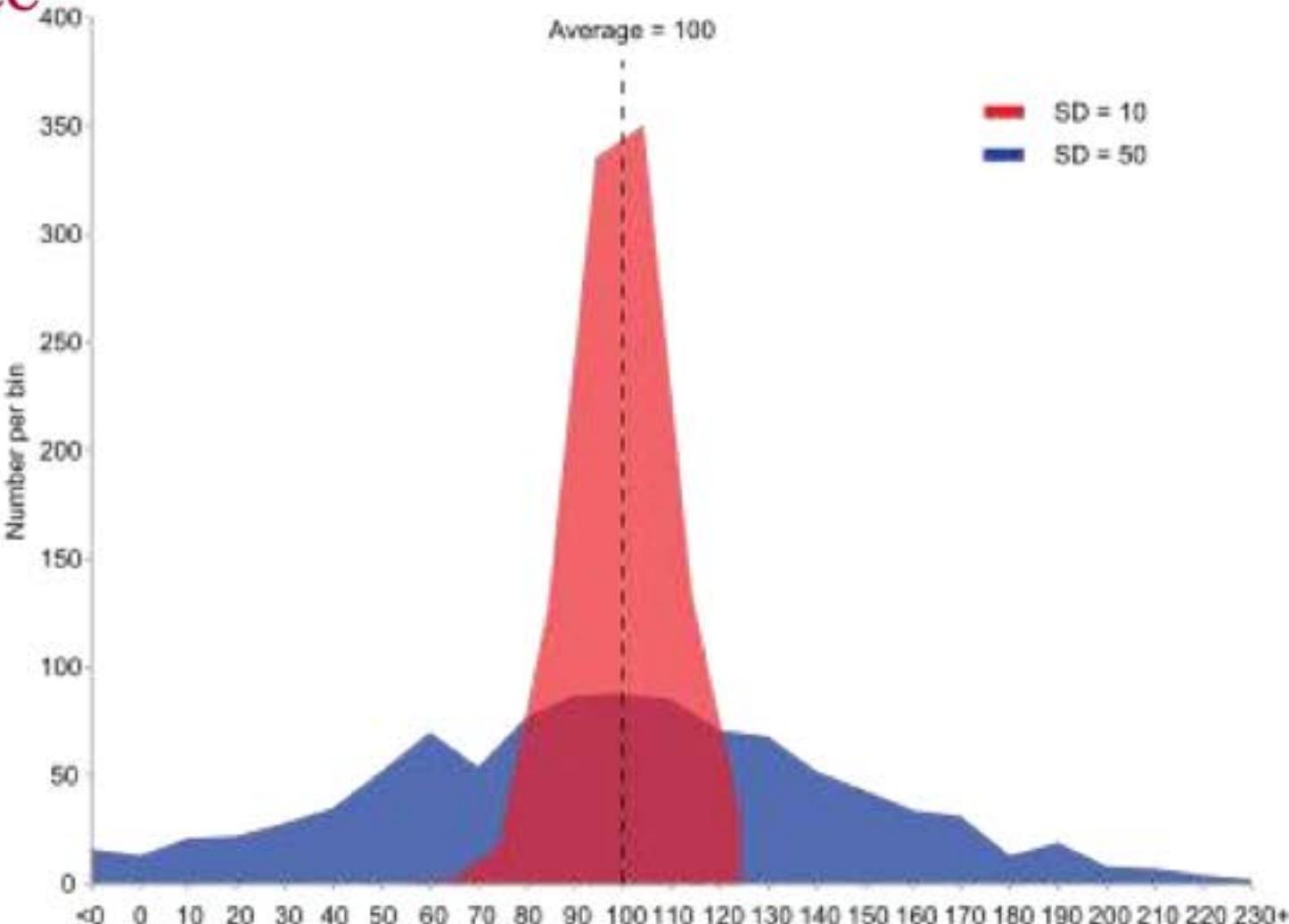
Mean or Average

$$\mu = \frac{\sum_{i=1}^N X_i}{N} = \frac{\sum X}{N}$$

Variance

$$\sigma^2 = \frac{\sum (X - \mu)^2}{N}$$

Standard deviation (σ) is square root of variance.



Mean, Median, Mode

Student Height (inches)

S_3	30
S_19	30
S_4	31
S_22	31
S_7	32
S_9	32
S_14	32
S_15	32
S_10	34
S_11	34
S_12	34
S_13	34
S_16	34
S_18	34
S_21	34
S_1	35
S_6	35
S_8	35
S_17	35
S_2	36
S_5	36
S_20	36

Mean → average of a data set

$$\text{Mean } (\mu) = \frac{\text{sum of all terms}}{\text{count of terms}}$$

$$\mu = 33.45455$$

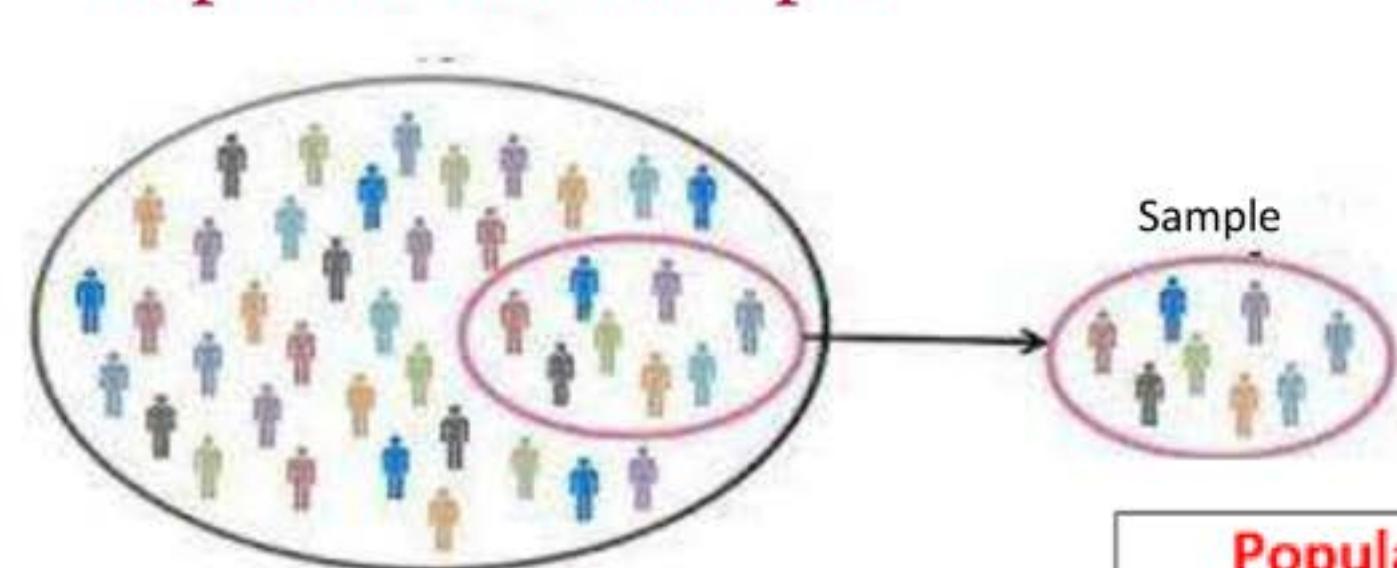
Median → middle value of the set of numbers.

$$\text{Median} = 34$$

Mode → most common number in a data set.

$$\text{Mode} = 34$$

Population & Sample



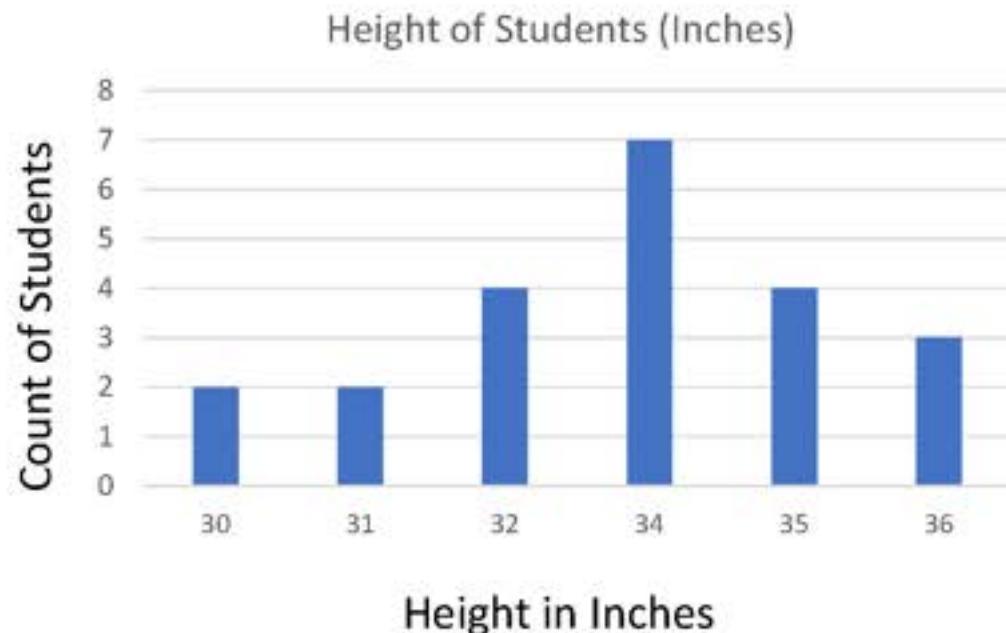
Population

- For blood CBC, would you draw few drops or drain the blood completely from a person?
- If you want to measure the average length of your hair, would you take a few hair samples or shave your head off?
- Average weekly hours spent in watching TV by Indians.

Population Mean	Sample Mean
$\mu = \frac{\sum_{i=1}^N x_i}{N}$ <p>N = number of items in the population</p>	$\bar{X} = \frac{\sum_{i=1}^n x_i}{n}$ <p>n = number of items in the sample</p>

Histogram

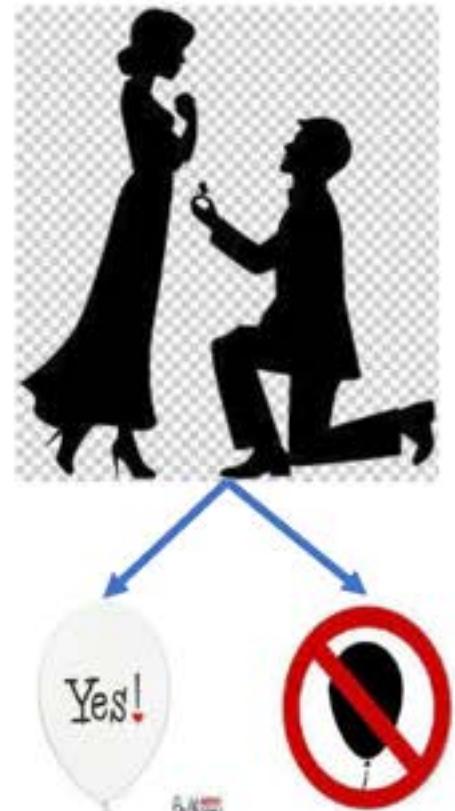
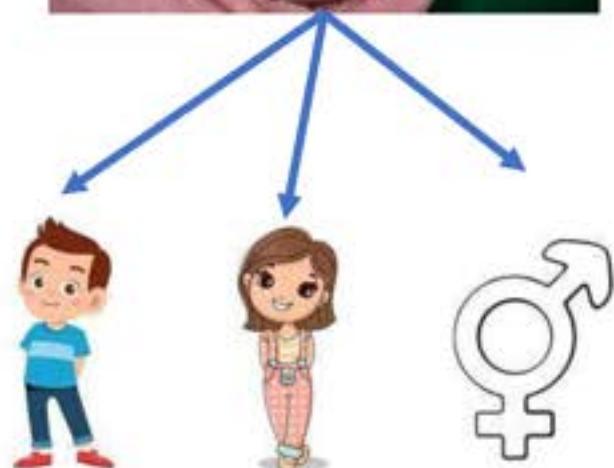
- Frequency distribution of data
- If continuous data, create buckets and find the membership in each bucket



How shall we deal with continuous data?

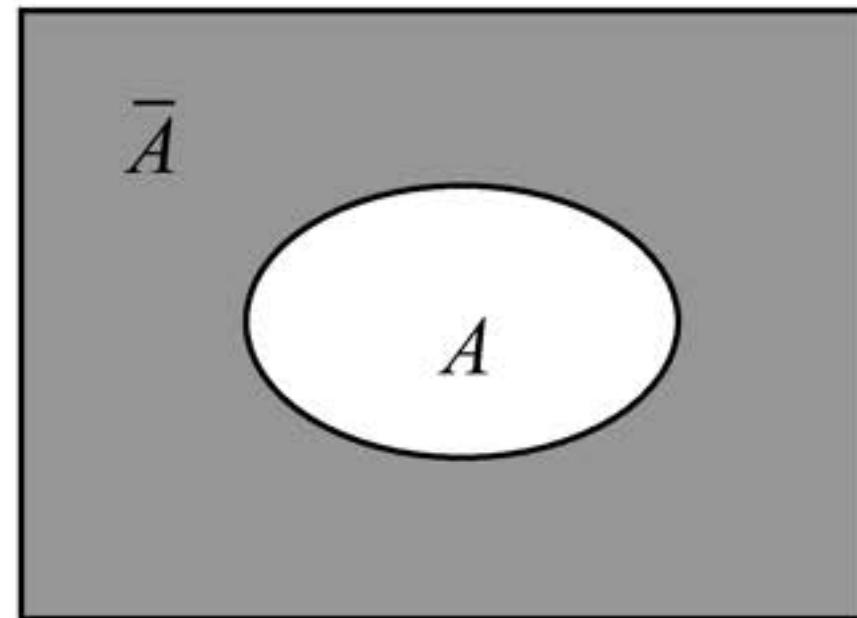
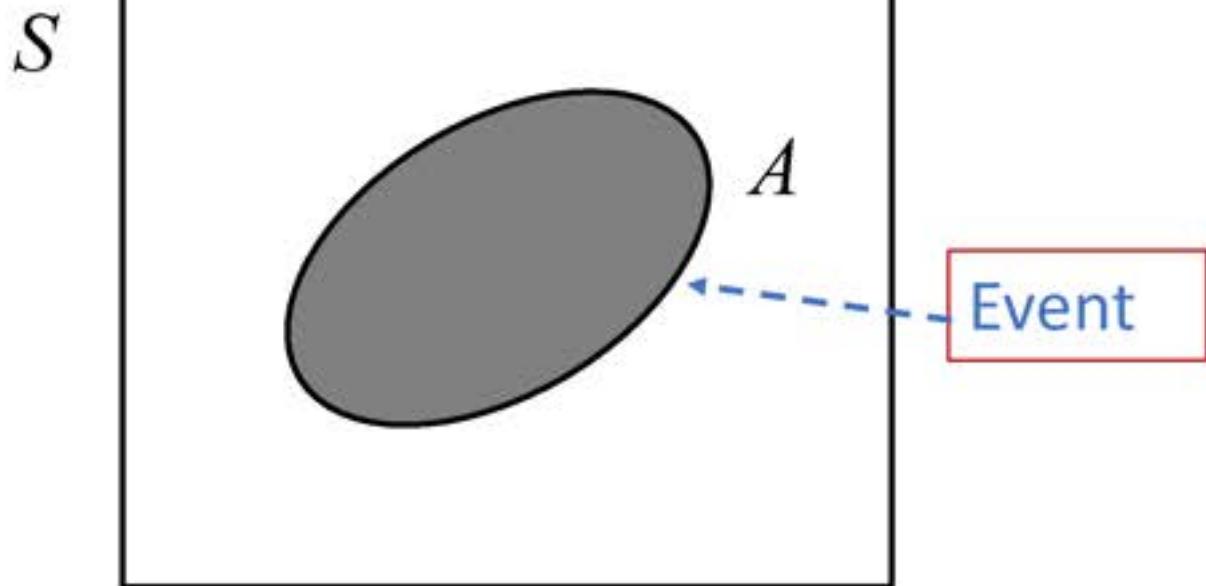
Student	Height (inches)
S_3	30
S_19	30
S_4	31
S_22	31
S_7	32
S_9	32
S_14	32
S_15	32
S_10	34
S_11	34
S_12	34
S_13	34
S_16	34
S_18	34
S_21	34
S_1	35
S_6	35
S_8	35
S_17	35
S_2	36
S_5	36
S_20	36

Observations and Events



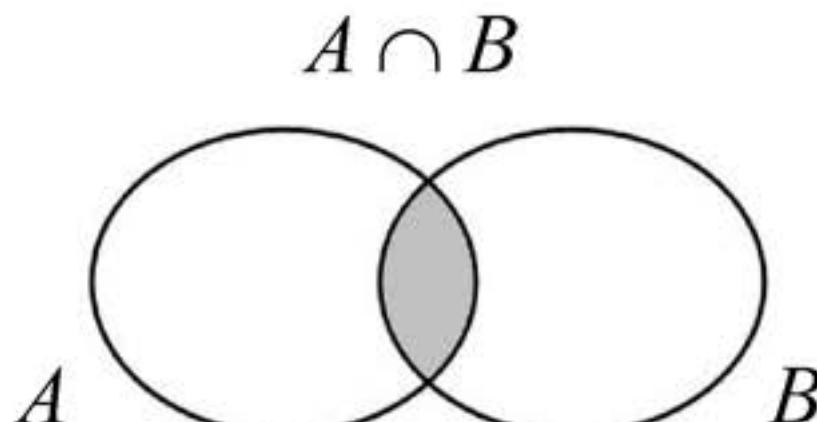
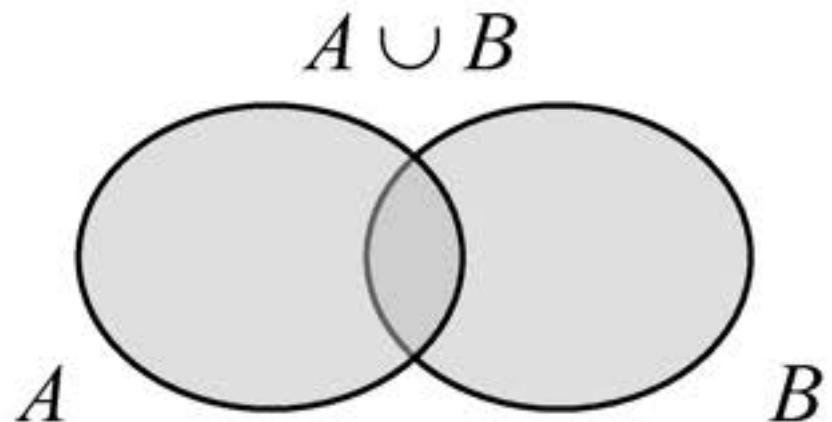
Venn Diagram (Sample Space & Event)

- All observations made form the sample space
- Event is the observation category
 - Baby is girl
 - Price of a stock increased
 - "YES" to marriage proposal



Venn diagram

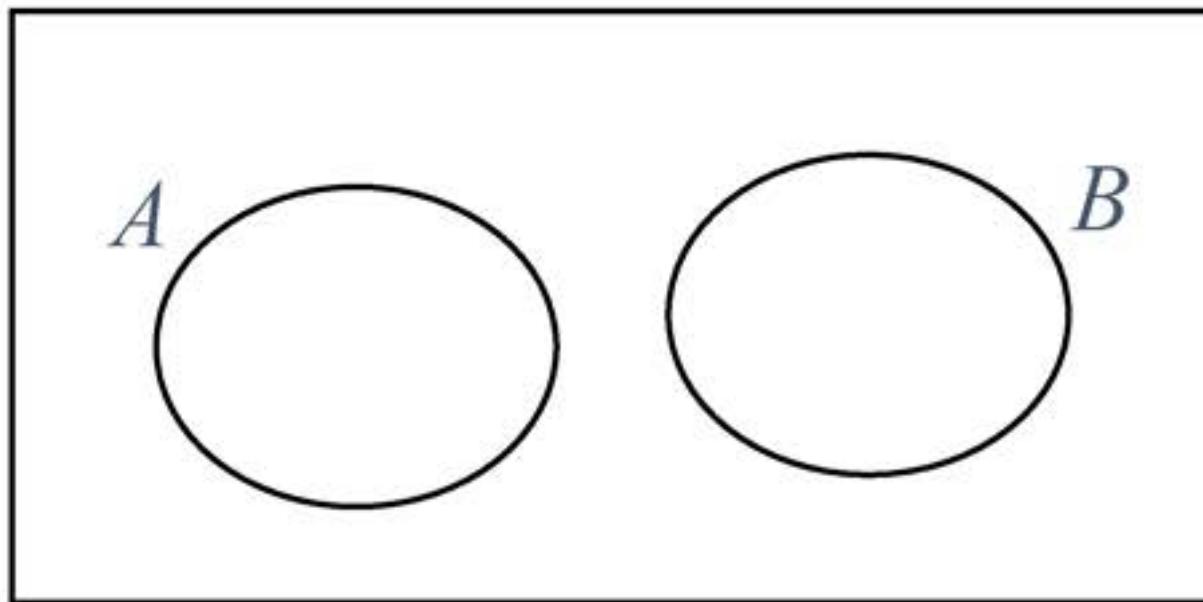
Operations on Sets



$A \cup B = \{\text{event belongs either to } A \text{ or } B\}$

$A \cap B = \{\text{event belongs both } A \text{ and } B\}$

Mutually Exclusive Sets



$$A \cap B = \emptyset$$

Null Set

Probability



500 newborn inspected



shutterstock.com - 1309129402



272

225

3

$$P[E] = \frac{n(E)}{n(S)} = \frac{n(E)}{N} = \frac{\text{no. of outcomes in } E}{\text{total no. of outcomes}}$$

$$P(\text{Boy}) = \frac{272}{500} = 0.54$$

$$P(\text{Girl}) = \frac{225}{500} = 0.45$$

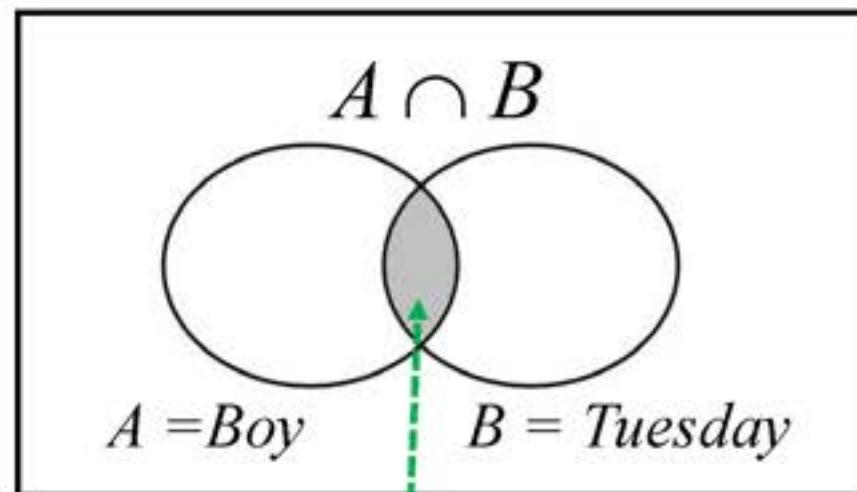
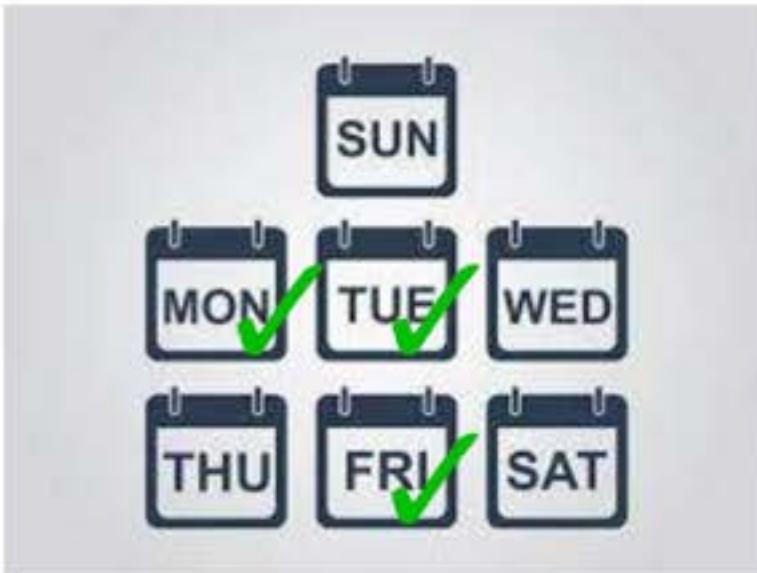
$$P(\text{Trans}) = \frac{3}{500} = 0.01$$



Conditional Probability



shutterstock.com - 150912948



Boys born
on Tuesday

What is the probability of a new born being a “*Boy*” given that today is “*Tuesday*”?

$$P(A|B)=P(A \cap B)/P(B)$$

Conditional Probability



	Monday	Tuesday	Friday	Total	Prob
Boys	272	232	176	680	0.53
Girls	225	248	137	610	0.47
Trans	3	0	0	3	0.00
Total	500	480	313	1293	
Prob	0.39	0.37	0.24		



Total babies on all 3 days = 1293

Boys born on Tuesday = 232

$$P(A \cap B) = \frac{232}{1293} = 0.18$$

$$P(A|B) = P(A \cap B)/P(B)$$

Prob of a new born being boy given that today is Tuesday

$$\rightarrow P(\text{Boy} | \text{Tuesday})$$

$$= \frac{0.18}{0.37} = 0.49$$

Joint Probability



Joint probability of 2 events is said to be the probability of both events happening at the same time.

Ex: You randomly visit the hospital to inspect. The probability of the new born is boy and today is Tuesday is referred as joint probability.

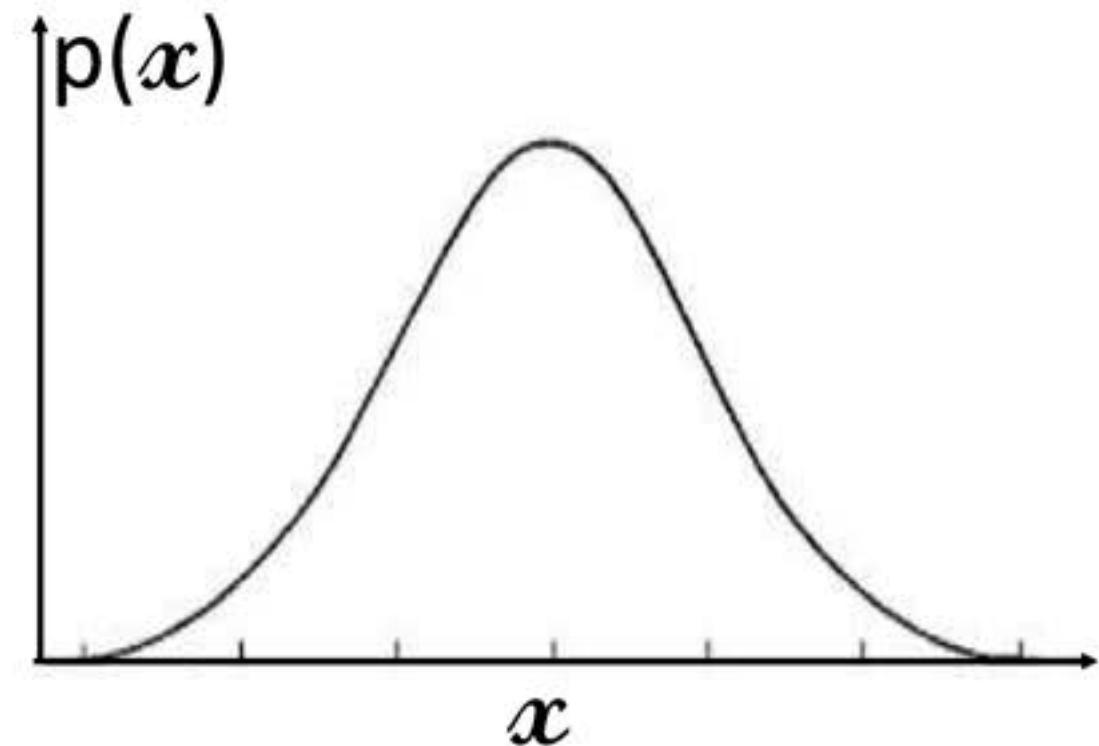
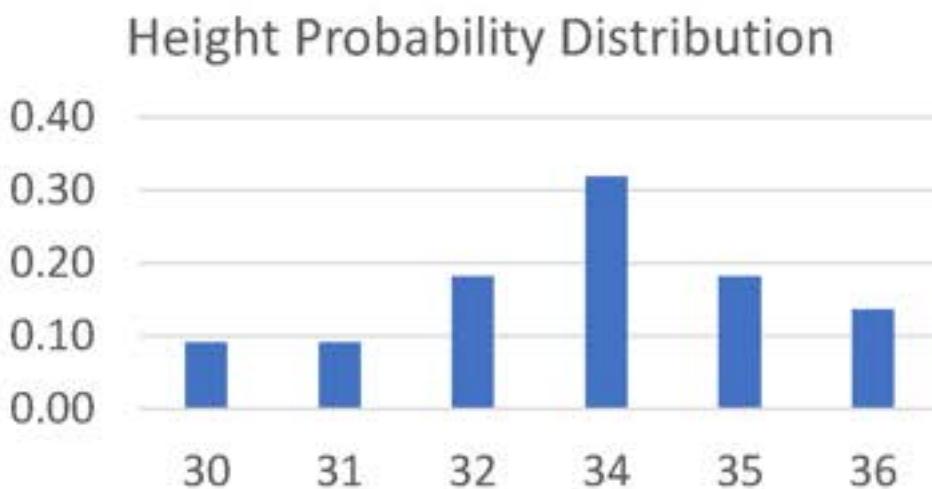


$$P(A \text{ and } B) = P(A | B) P(B) = P(A \cap B)$$

$P(A \text{ and } B)$ is also denoted as $P(A, B)$

Probabilities for discrete and continuous variables

Student	Height (inches)
S_3	30
S_19	30
S_4	31
S_22	31
S_7	32
S_9	32
S_14	32
S_15	32
S_10	34
S_11	34
S_12	34
S_13	34
S_16	34
S_18	34
S_21	34
S_1	35
S_6	35
S_8	35
S_17	35
S_2	36
S_5	36
S_20	36



Thank you !!!!!

Machine Learning (19CSE305)

Features, Distance & Similarity



Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Recap of covered sessions
- Features / attributes
- Types of Data & Characteristics
- Distances & Similarities

Linear Equations & Matrices

$$\begin{aligned}x + 2y + 4z &= 33 \\4x + 3y - z &= 29 \\-x - y + 2z &= -2\end{aligned}$$

$$A \quad X = C$$
$$\begin{bmatrix} 1 & 2 & 4 \\ 4 & 3 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ -1 \end{bmatrix}$$

$$A^{-1}A = A A^{-1} = I$$

$$A^{-1} = \frac{1}{\det(A)} \text{adj}(A)$$

$$\begin{bmatrix} 2 & 5 & 7 & 8 \\ 1 & 2 & 3 & 1 \\ 4 & 5 & 0 & 1 \end{bmatrix}$$

No inverse exists mathematically for rectangular matrices. However, Singular Value Decomposition (SVD) algorithm (and such others) work on principle of error minimization and help us arrive at a pseudo-inverse. This pseudo-inverse meets most of day-to-day needs.

Statistics & Probability

Mean → average of a data set

Median → middle value of the set of numbers.

Mode → most common number in a data set.

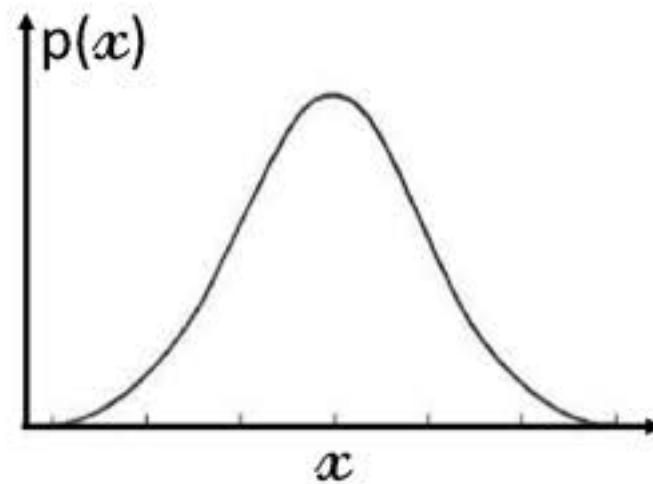
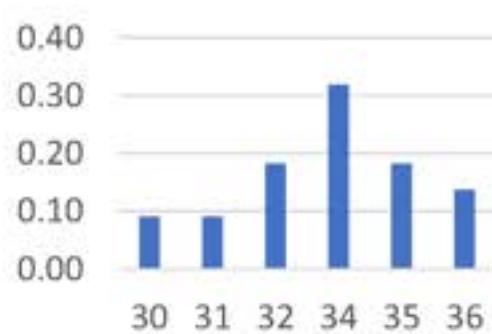
Variance & Standard Deviation & their relation

Population Mean	Sample Mean
$\mu = \frac{\sum_{i=1}^N x_i}{N}$	$\bar{X} = \frac{\sum_{i=1}^n x_i}{n}$
N = number of items in the population	n = number of items in the sample

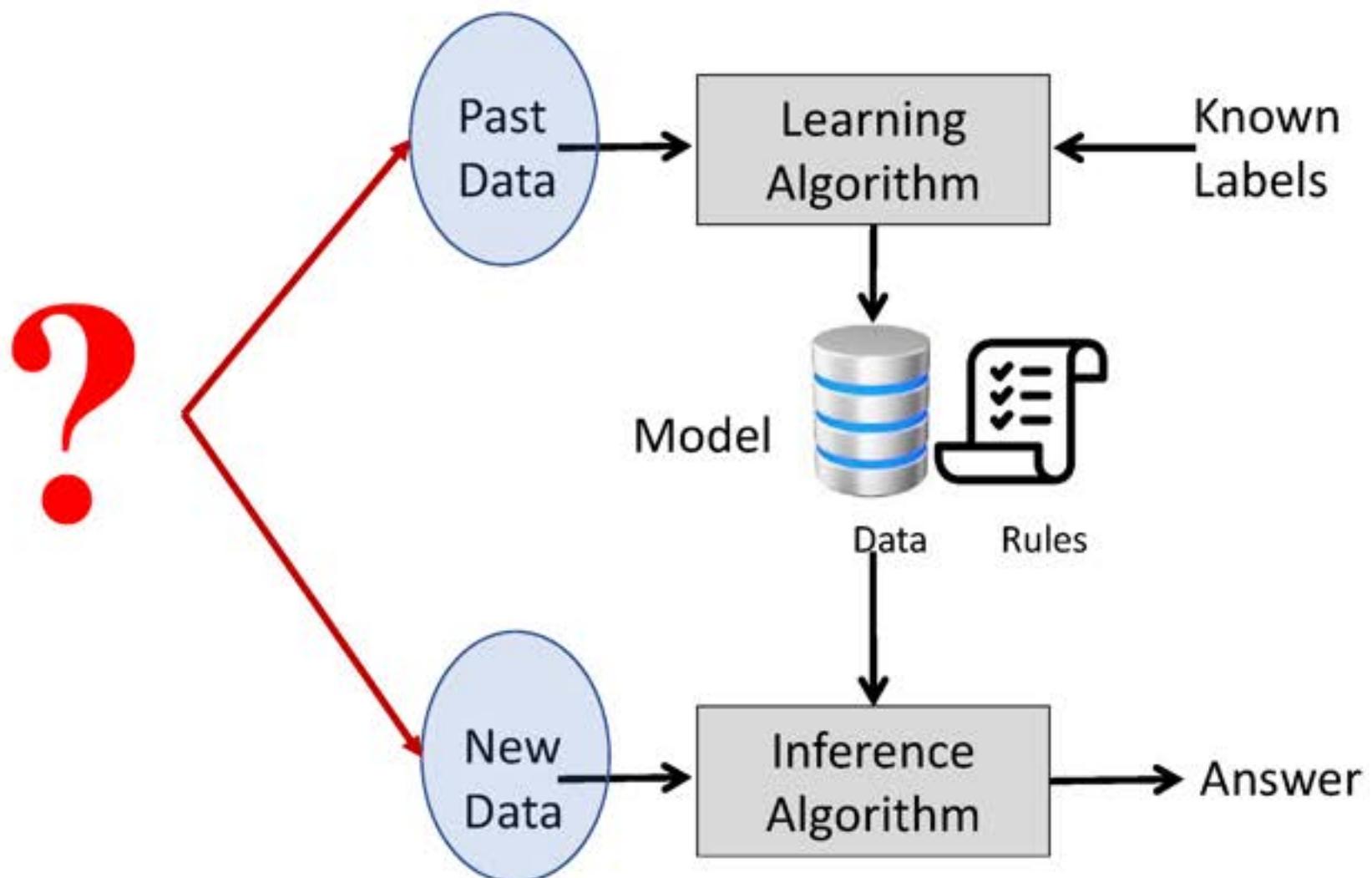
$$P[E] = \frac{n(E)}{n(S)} = \frac{n(E)}{N} = \frac{\text{no. of outcomes in } E}{\text{total no. of outcomes}}$$

$$P(A|B) = P(A \cap B)/P(B)$$

$$P(A \text{ and } B) = P(A|B) P(B) = P(A \cap B)$$



Machine Learning System



Features



Do I have fever?



Will it rain today?



How much will my money grow?

Will I be able to reach college for my job interview?

How long would it take for me to pay my loan?

Will this medicine create adverse reactions?

How can I reduce my monthly expenses?

Will the monsoon be good this year?

Will I be happy with this person?

Will this borrower default?

Shall I get a heart stroke?

How does an autonomous vehicle move?

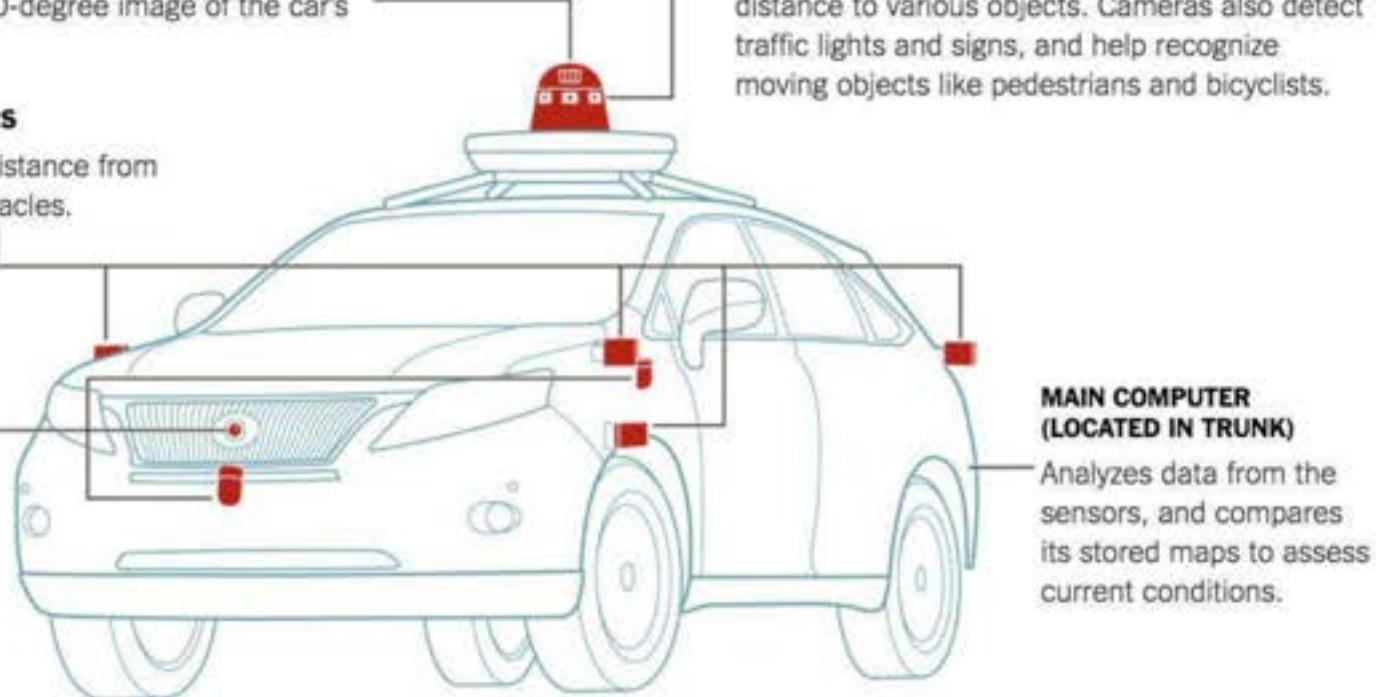
LIDAR UNIT

Constantly spinning, it uses laser beams to generate a 360-degree image of the car's surroundings.

RADAR SENSORS

Measure the distance from the car to obstacles.

ADDITIONAL LIDAR UNITS



CAMERAS

Uses parallax from multiple images to find the distance to various objects. Cameras also detect traffic lights and signs, and help recognize moving objects like pedestrians and bicyclists.

What is important?

Sensors or decision makers?

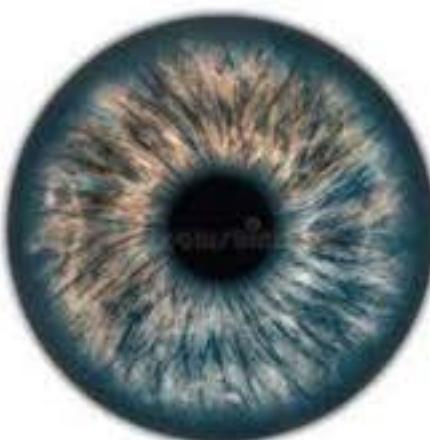
Feature

Feature is an attribute (Qualitative or Quantitative) that represents an object.

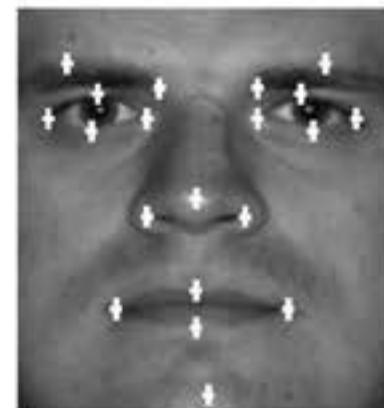
- Attribute is a property or characteristic of the object; varies from object to another
- Qualitative features: Color, Shape etc.
- Quantitative features: measured values such as height, weight, frequency etc.



Color: Orange
Shape: spherical
Texture: dimpled
Hardness: Soft & moist



Color, pattern, shape,
Rings, spots, stripes



Contours
Distances
Lengths & widths
Skin color etc

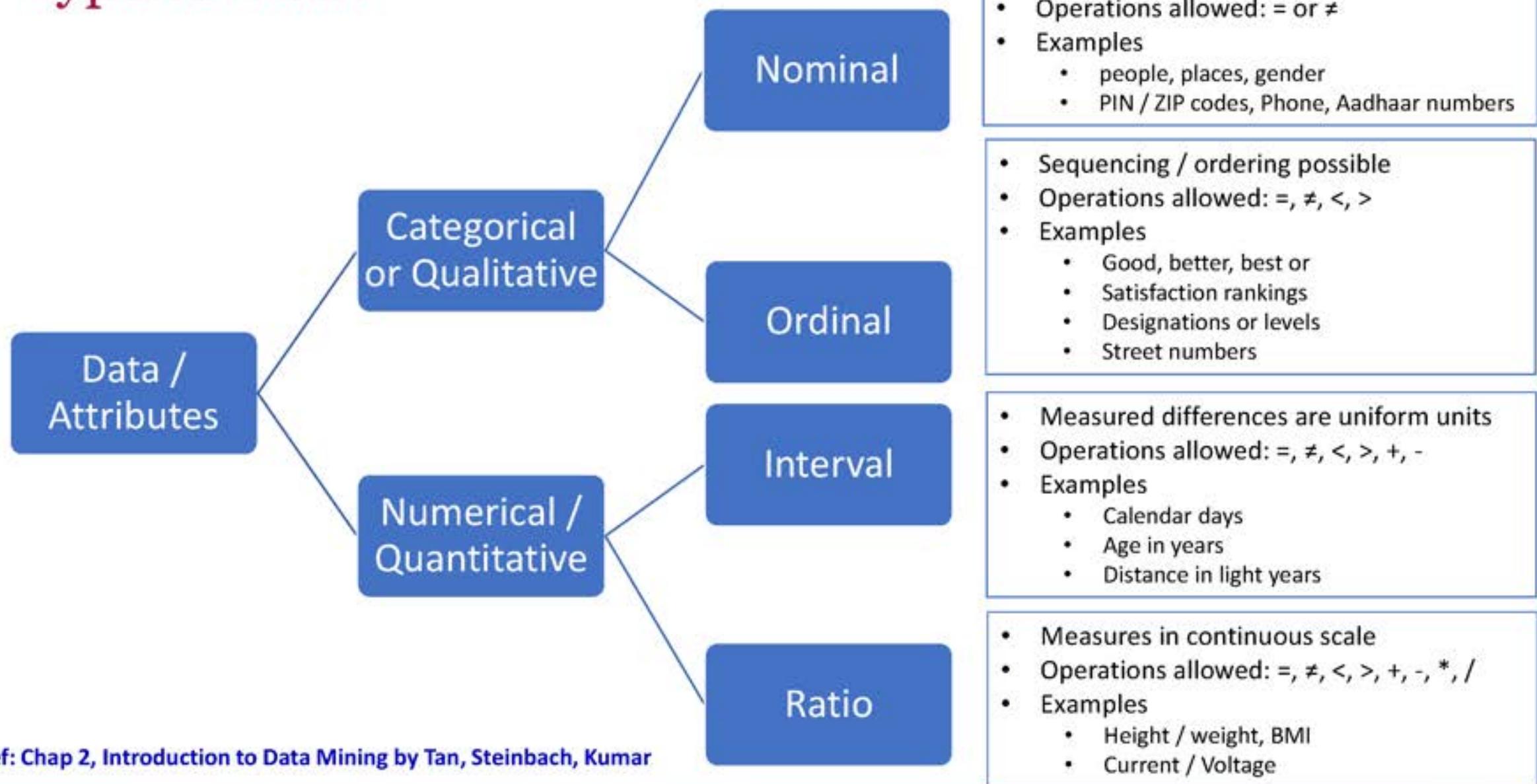
Data

- Referred to as vectors, points, instances, objects, entities, patterns, events, cases etc.
- Real life data is far from being perfect
 - Issues with capture
 - Improper management
 - Noise presence during capture
- Various pre-processing steps employed

W's of Dataset

- Who (or what) does the observation represent? The most important question to be addressed.
- What are the variables?
- Why was the data collected?
- How was the data collected?
- When/Where was the data collected?

Types of Data



Ref: Chap 2, Introduction to Data Mining by Tan, Steinbach, Kumar

Data Characteristics

- Dimensionality
 - # of attributes the object possesses
 - Count of parameters in the feature vector
 - Dimensionality of the vector space
- Sparsity
 - Amount of zero values present
 - Sparse data set allows for computational efficiency
 - May help in dimensionality reduction
- Resolution
 - Closeness of observation in time or space
 - Temperature measured every second or hour
 - Low resolution may lead to loss of information
 - *Battle tank movements may not be observed with 10 meter resolution camera on a satellite*
 - High resolution may add noise or lead to loss of data
 - *Stock price change every minute may not help in deciding on investment*

79	17	44	87	6	15	84
97	64	16	55	58	86	93
6	37	75	53	81	50	24
84	15	31	59	30	35	77
91	89	2	78	3	45	80
3	17	11	61	82	92	51
95	22	67	29	27	68	35
92	64	13	59	72	18	16
73	24	76	39	9	15	5
78	46	14	75	43	73	97
59	6	21	13	54	74	86

Dense Data Matrix

2	1	4	0	0	1	0
4	0	2	0	0	3	1
0	0	2	0	0	0	0
0	0	3	5	3	0	0
5	2	1	0	0	0	0
1	5	1	5	2	0	0
0	0	1	0	0	0	0
0	0	5	0	0	5	5
0	0	1	0	0	0	5
1	1	0	0	3	0	1
2	1	3	3	0	3	1

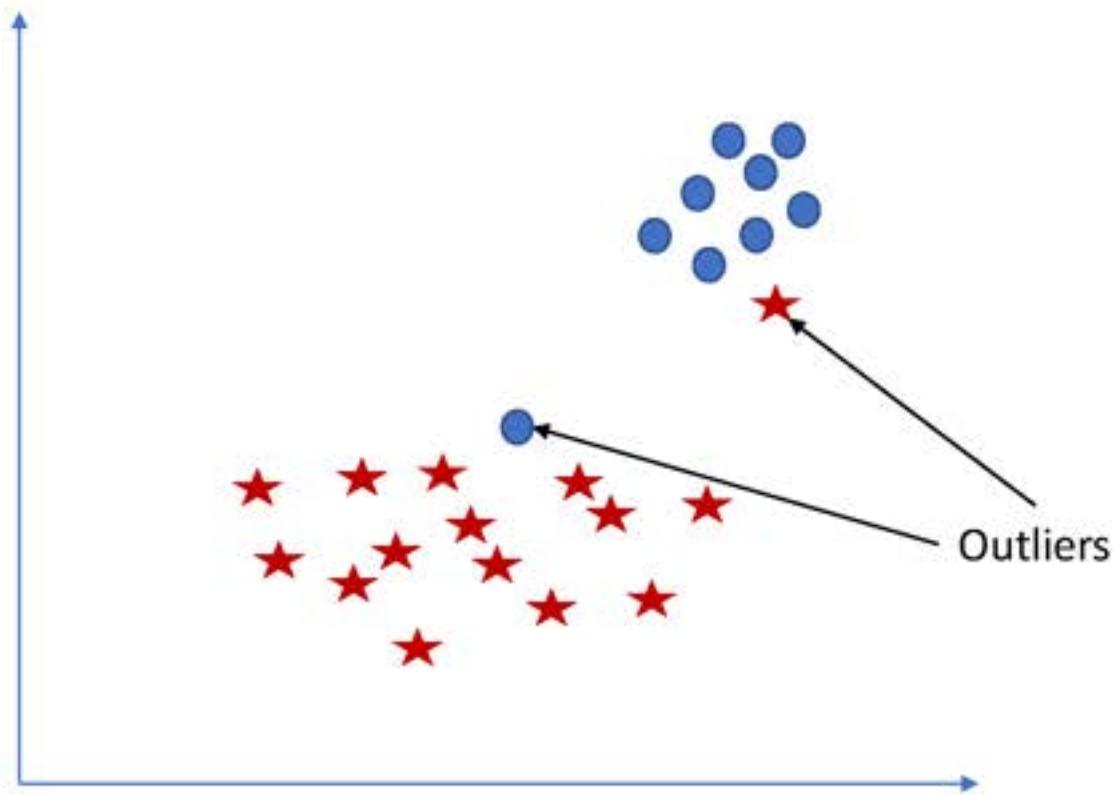
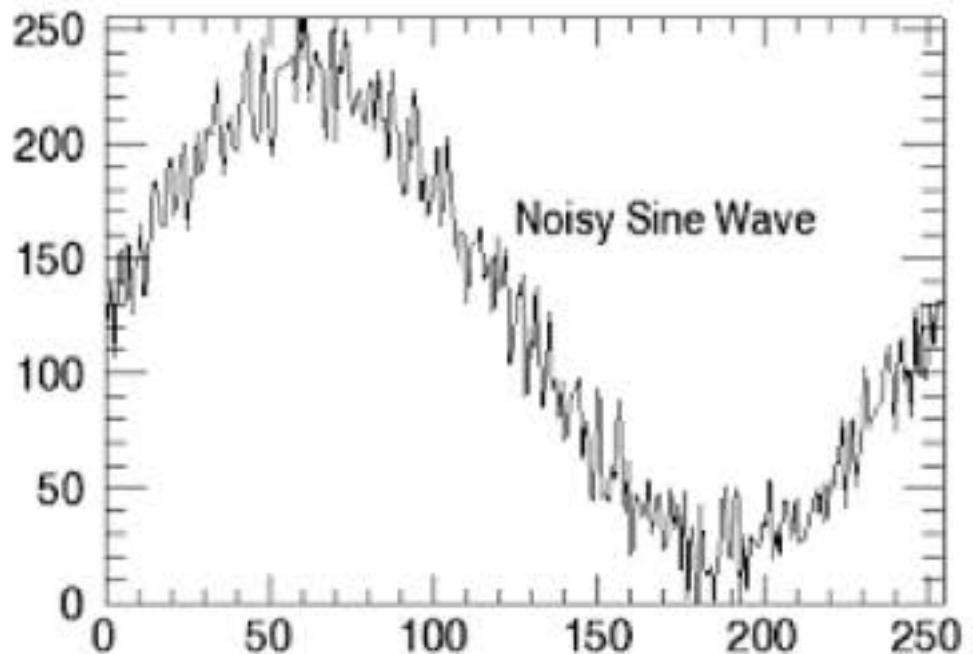
Sparse Data Matrix

Feature Vector

Card Type	Transaction Amount	Place of residence	Place of Transaction	POS / On-line	Avg Tx Amount
Silver	350	Bangalore	Gurgaon	POS	235
Gold	12,499	Agra	Delhi	On-line	2546
Gold	450	Hyd	Hyd	On-line	1123
Silver	35,000	Chennai	Bhopal	POS	279

	prompt	happy	Satisfy	eager	quickly	kind	attentive	delay	rude	wait	sad	unhappy	bad	
Document 1	1	2					1							I am very happy with the prompt customer support I have received from you. My queries were resolved without much delay and I am so happy for that.
Document 2								1	1	1				You guys are pathetic. You made me wait for hours without any help and the agent was rude. I am so unhappy.
Document 3	1	1			1	1								Very happy with your prompt service. The executive was kind and attentive to my needs.

Noise and Outliers



Thank you !!!!!



Machine Learning (19CSE305)

Distance, Similarity & Independence

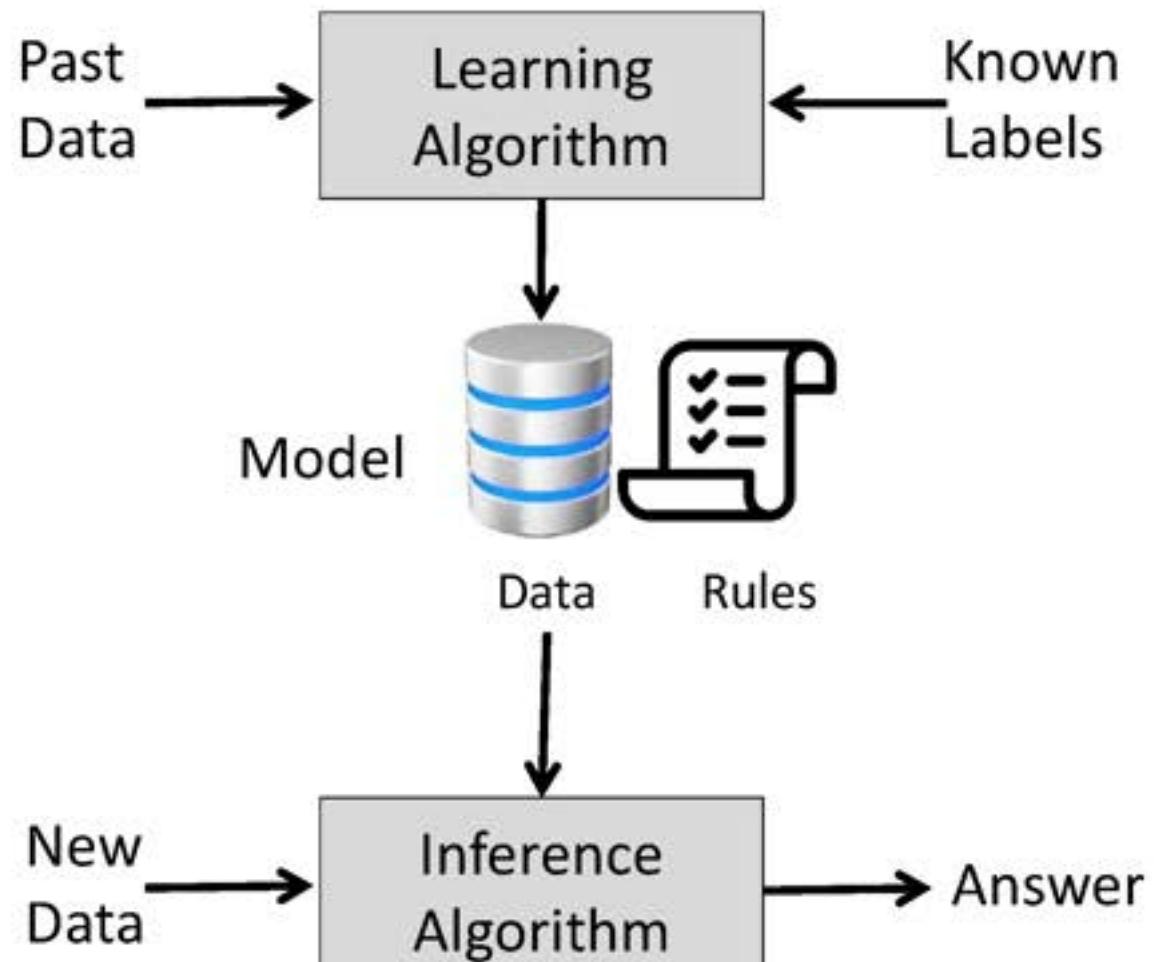


Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

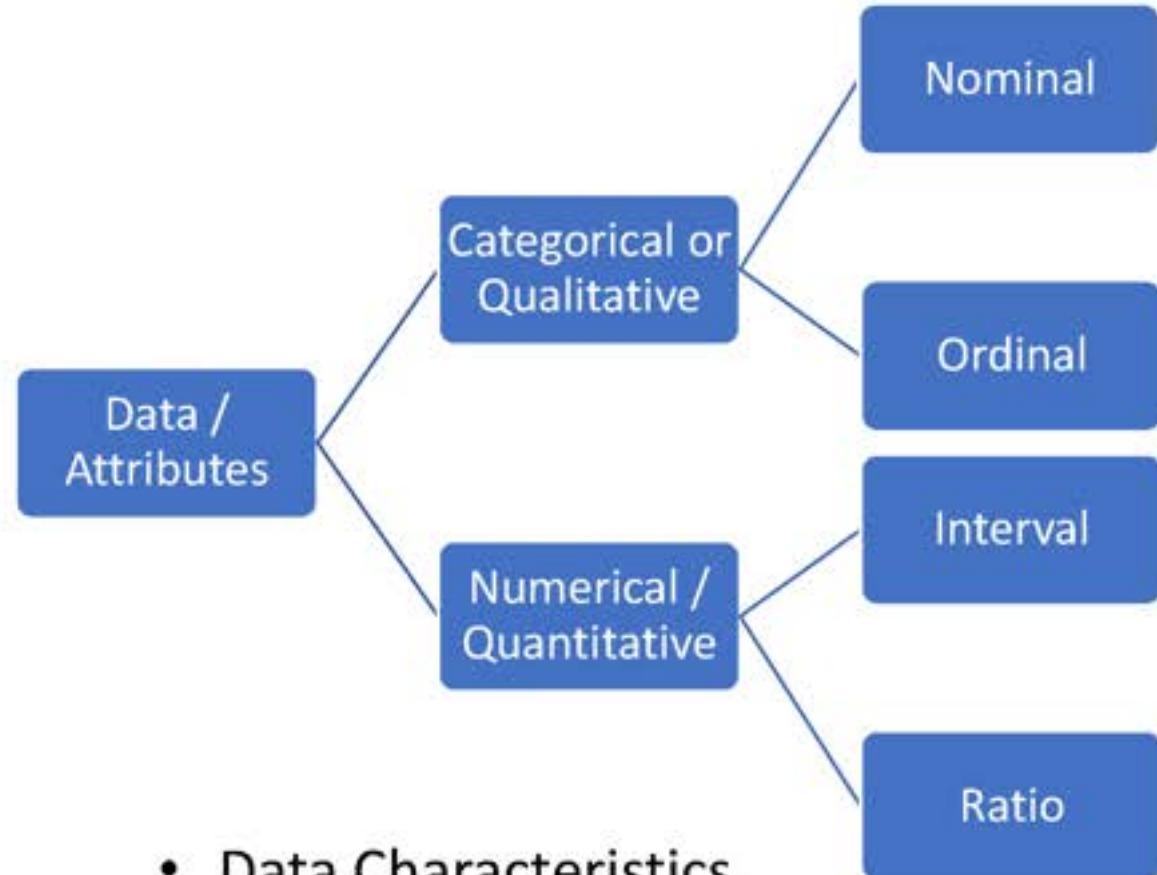
Topics

- Recap of Data & Characteristics
- Distances & Similarities
- Projections
- Independence & Orthogonality
- Observable and Controllable
- Time Invariant Systems

Data, Types & Characteristics

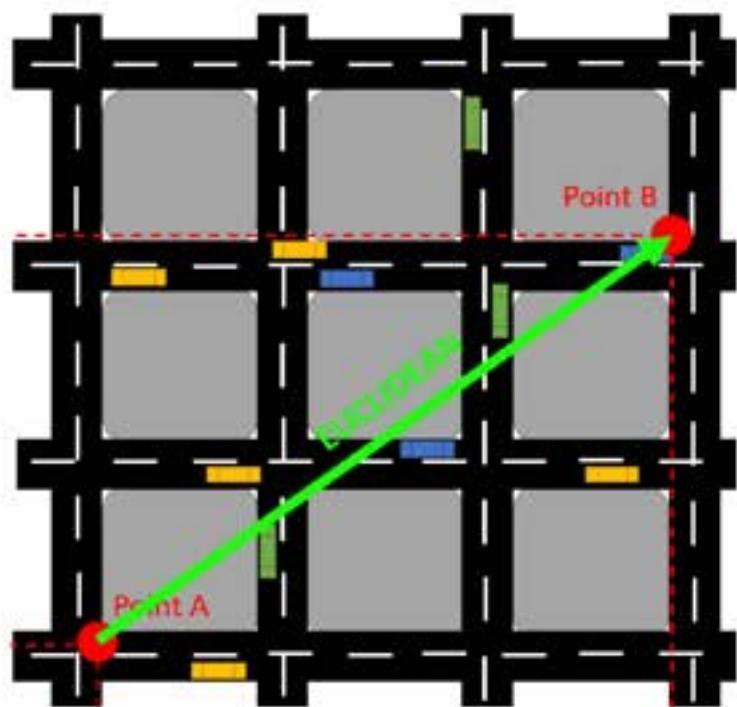


Data is as important as the algorithm that uses the data.

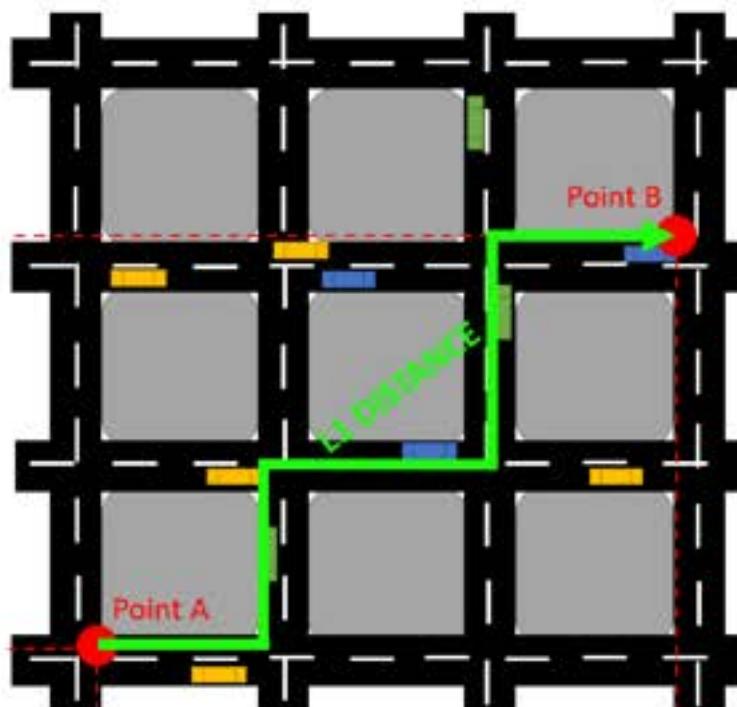


- **Data Characteristics**
 - Dimensionality
 - Sparsity
 - Resolution
- **Noise & Outliers**
- **W's of the dataset**

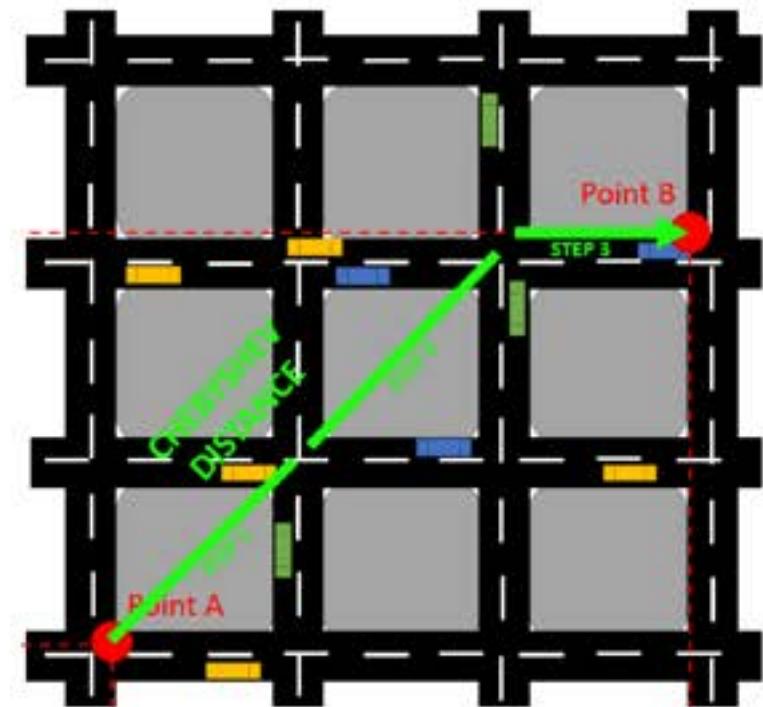
Distance Measures



Euclidean



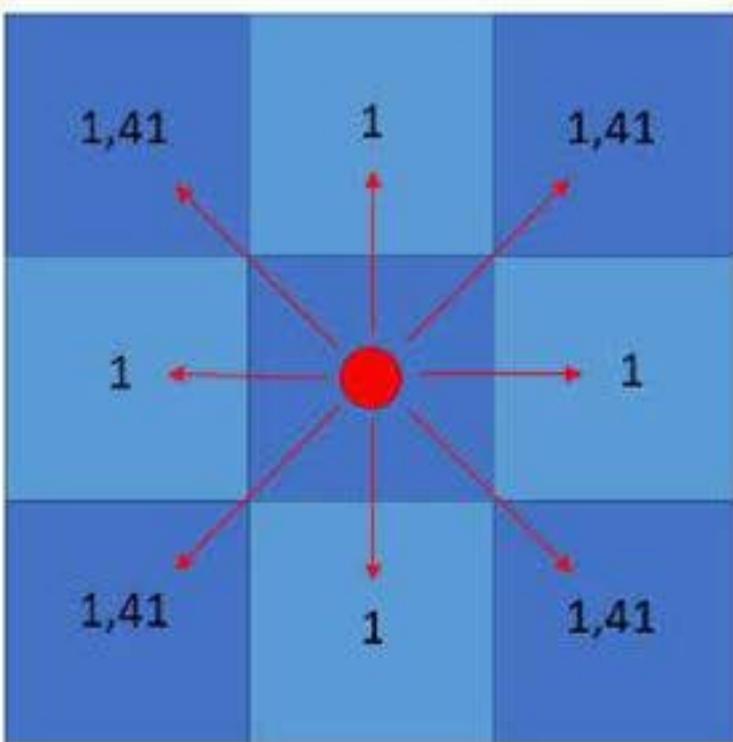
City-block or
Manhattan



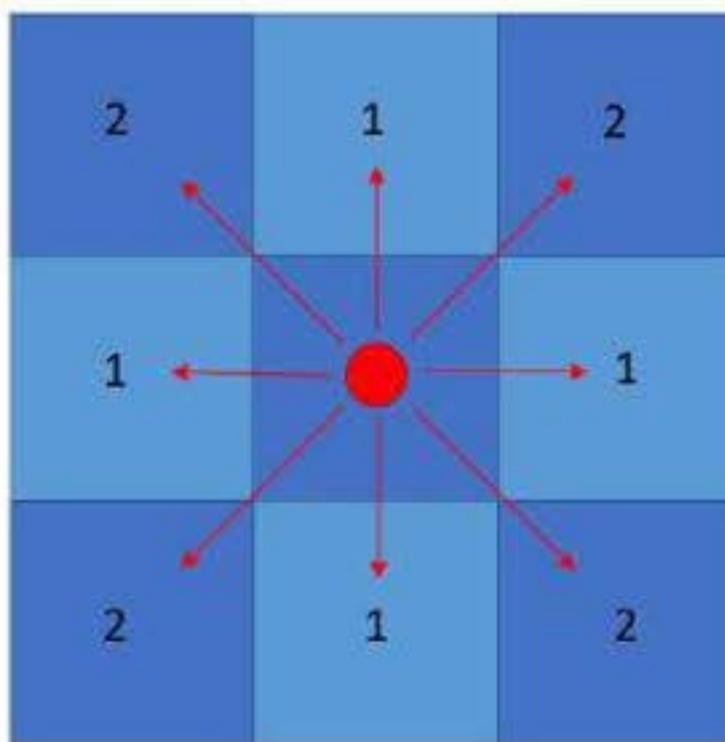
Chebyshev

Distance Measures

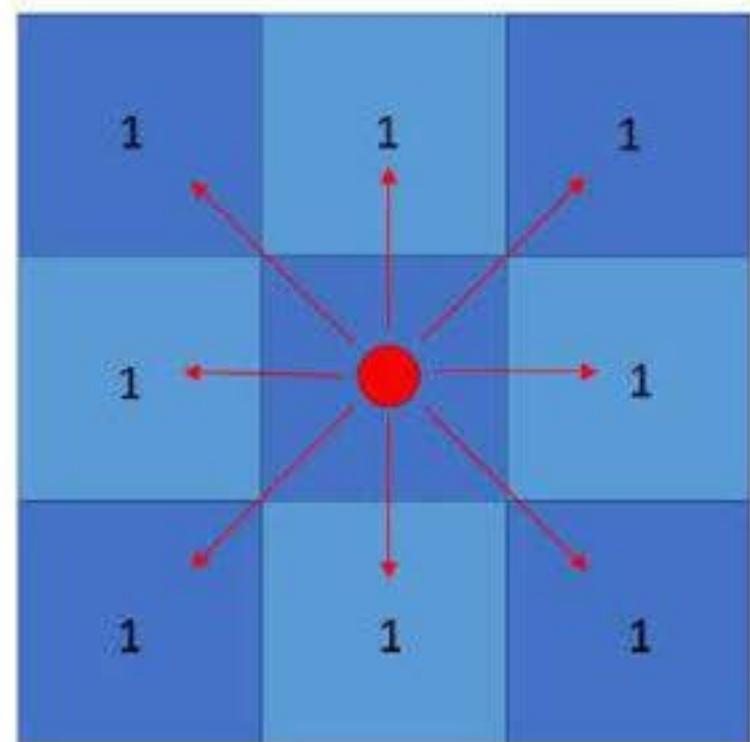
EUCLIDEAN (STRAIGHT LINE)



L1 (CITY BLOCK)

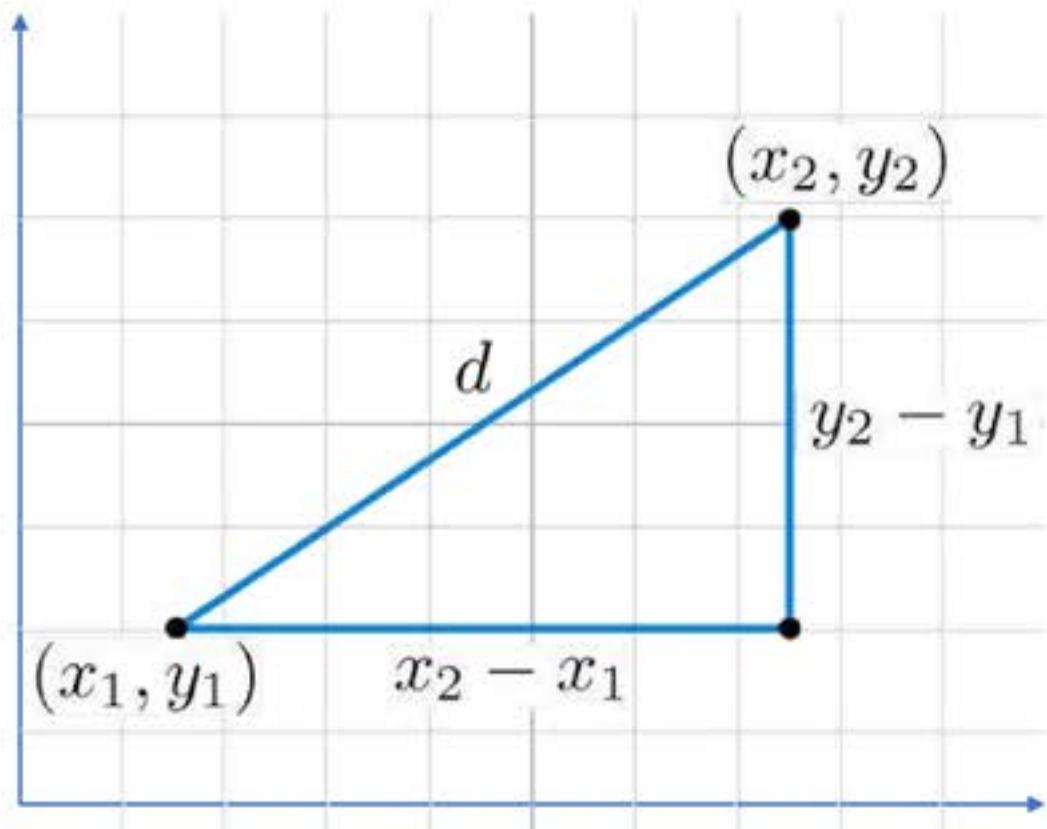


CHEBYSHEV (CHESSBOARD)



Source: <https://towardsdatascience.com/>

Distance Measures – Euclidean & Minkowski



Minkowski Distance

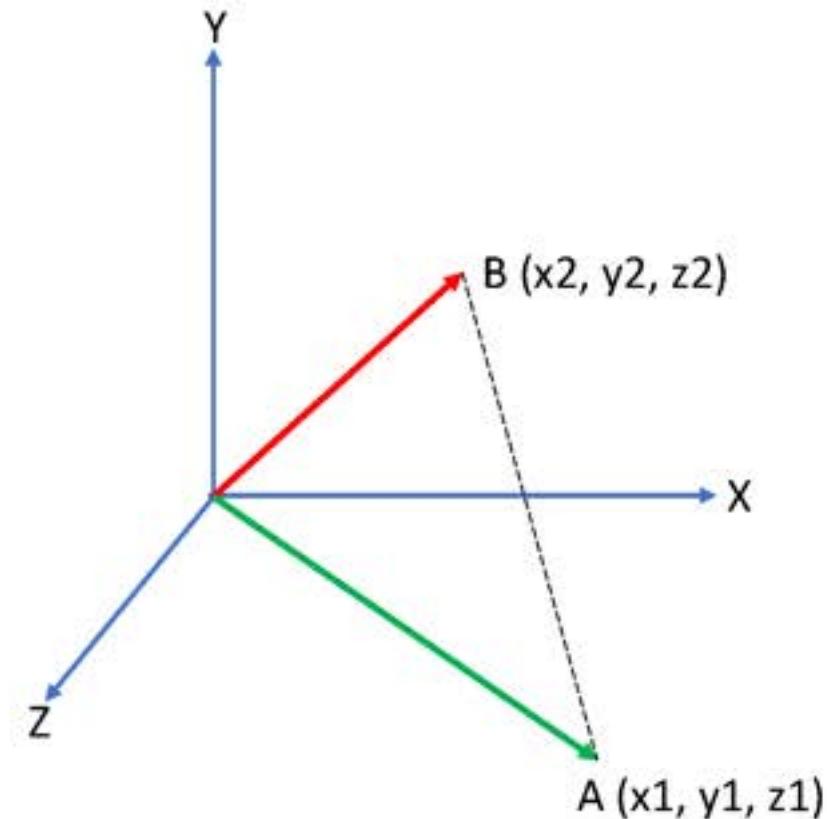
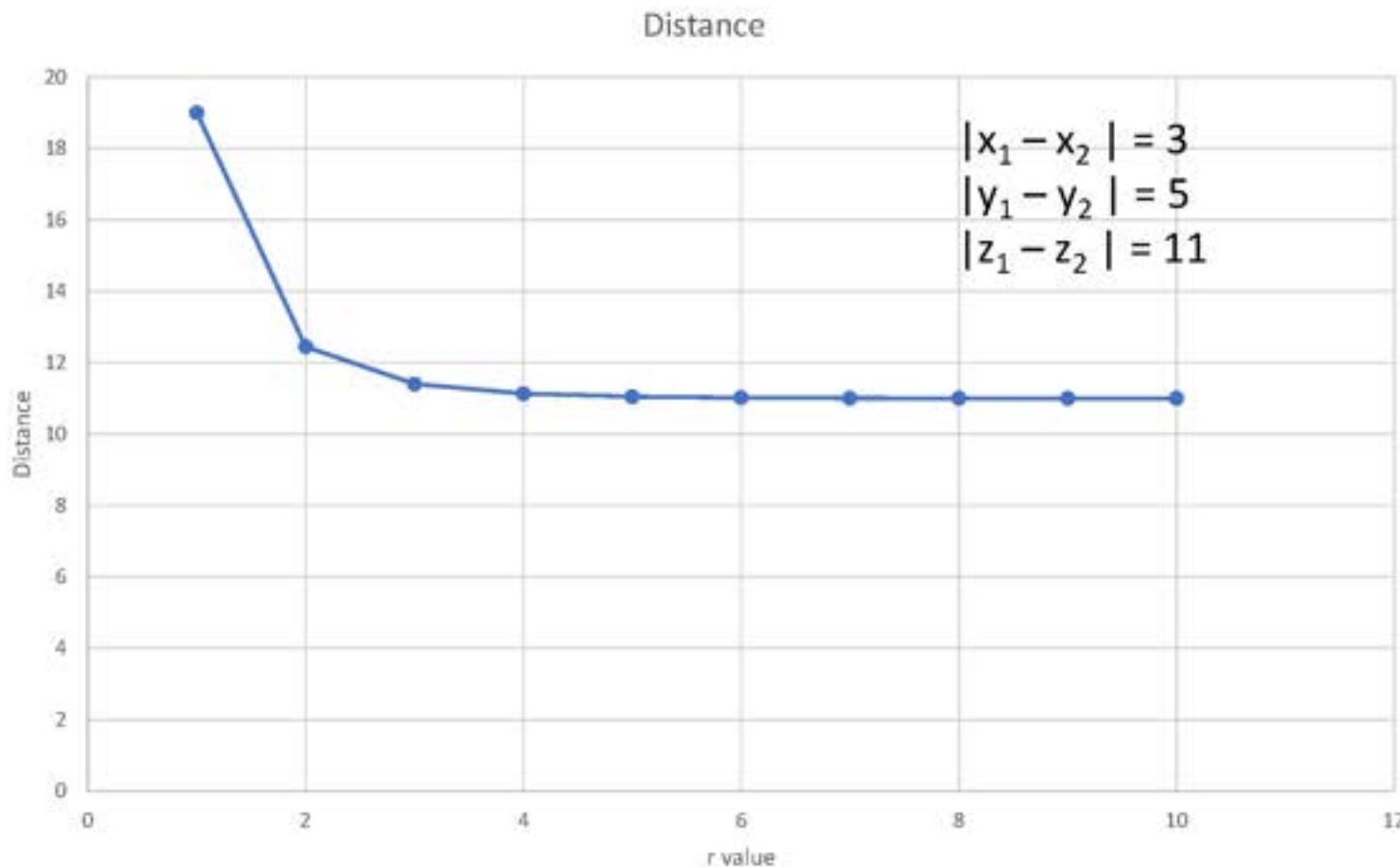
$$d(\mathbf{x}, \mathbf{y}) = \left(\sum_{k=1}^n |x_k - y_k|^r \right)^{1/r}$$

Minkowski distance is a generalization of Manhattan & Euclidean distances.
Minkowski with $r=1 \rightarrow$ city-block / Manhattan
Minkowski with $r=2 \rightarrow$ Euclidean

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{k=1}^n (x_k - y_k)^2}$$

Euclidean Distance

Minkowski Distance with 'r' variation



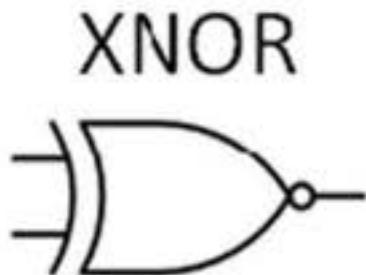
Properties of Distances

- Distance properties
 - $d(x, y) \geq 0$ for all x and y and $d(x, y) = 0$ if and only if $x = y$.
 - $d(x, y) = d(y, x)$ for all x and y . (Symmetry)
 - $d(x, z) \leq d(x, y) + d(y, z)$ for all points x , y , and z . (Triangle Inequality)
- A distance that satisfies these properties is a **metric**

Similarity & Properties

- Similarity
 - measure of how alike two objects are
 - higher value when two objects are similar
- Similarity properties
 - $0 \leq s(x, y) \leq 1$ for all x and y
 - $s(x, y) = 1$ when $x = y$ (mostly*)
 - $s(x, y) = s(y, x)$ for all x and y . (Symmetry)

Binary Vectors: Simple Matching



INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	1

- f_{01} = number of attributes where **x** was 0 and **y** was 1
 f_{10} = number of attributes where **x** was 1 and **y** was 0
 f_{00} = number of attributes where **x** was 0 and **y** was 0
 f_{11} = number of attributes where **x** was 1 and **y** was 1

Similarity Matching Coefficient (SMC)

$$\begin{aligned} &= \text{number of matches} / \text{number of attributes} \\ &= (f_{11} + f_{00}) / (f_{01} + f_{10} + f_{11} + f_{00}) \end{aligned}$$

	b7	b6	b5	b4	b3	b2	b1	b0
X	1	0	0	0	1	0	0	0
Y	1	1	0	1	1	1	1	0
XNOR	1	0	1	0	1	0	0	1

$$\begin{aligned} f_{01} &= 4 \\ f_{10} &= 0 \\ f_{00} &= 2 \\ f_{11} &= 2 \end{aligned}$$

$$\begin{aligned} \text{SMC} &= (f_{11} + f_{00}) / (f_{01} + f_{10} + f_{11} + f_{00}) \\ &= (2 + 2) / 8 = 0.5 \end{aligned}$$

Binary Vectors: Jaccard Similarity

2 - input AND gate



A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

Jaccard Coefficient (JC)

$$\begin{aligned} \text{JC} &= \text{number of } 11 \text{ matches} / \# \text{ attributes with any value } 1 \\ &= f_{11} / (f_{01} + f_{10} + f_{11}) \end{aligned}$$

	b7	b6	b5	b4	b3	b2	b1	b0
X	1	0	0	0	1	0	0	0
Y	1	1	0	1	1	1	1	0
XNOR	1	0	1	0	1	0	0	1

$$\begin{aligned} f_{01} &= 4 \\ f_{10} &= 0 \\ f_{00} &= 2 \\ f_{11} &= 2 \end{aligned}$$

$$\begin{aligned} \text{JC} &= (f_{11}) / (f_{01} + f_{10} + f_{11}) \\ &= (2) / 6 = 0.33 \end{aligned}$$

Cosine Similarity

$$A = \{a_1, a_2, \dots, a_n\}$$

$$B = \{b_1, b_2, \dots, b_n\}$$

If **A** and **B** are two document vectors, then

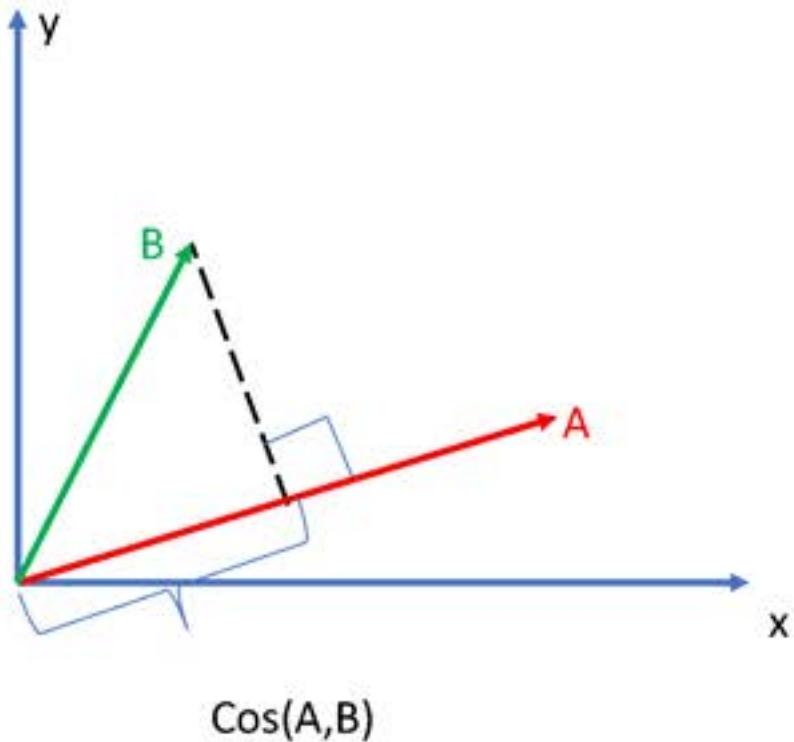
$$\cos(A, B) = \langle A, B \rangle / \|A\| \|B\|$$

$$\langle A, B \rangle = \sum_{k=1}^n a_k * b_k$$

$\|A\|$ and $\|B\|$ are lengths of vectors A & B

$$A = 3 \ 2 \ 0 \ 5 \ 0 \ 0 \ 0 \ 2 \ 0 \ 0$$

$$B = 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 2$$



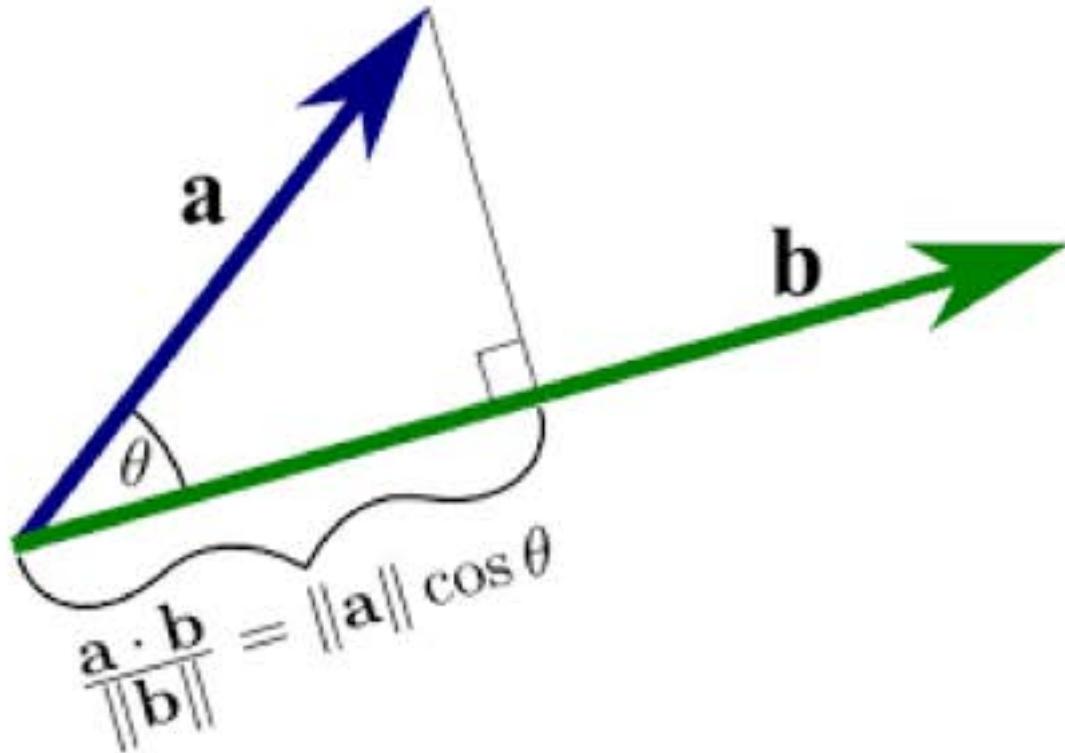
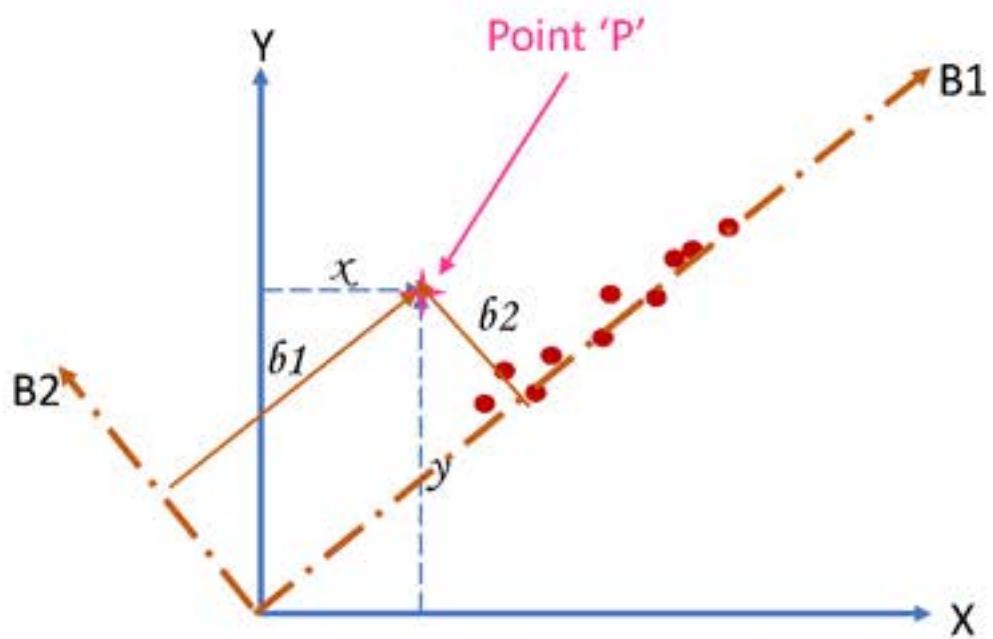
$$\langle A, B \rangle = 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5 \quad \text{Also known as dot product}$$

$$\|A\| = (3^2 + 2^2 + 0^2 + 5^2 + 0^2 + 0^2 + 0^2 + 2^2 + 0^2 + 0^2)^{0.5} = (42)^{0.5} = 6.481$$

$$\|B\| = (1^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2 + 1^2 + 0^2 + 2^2)^{0.5} = (6)^{0.5} = 2.449$$

$$\cos(A, B) = 0.3150$$

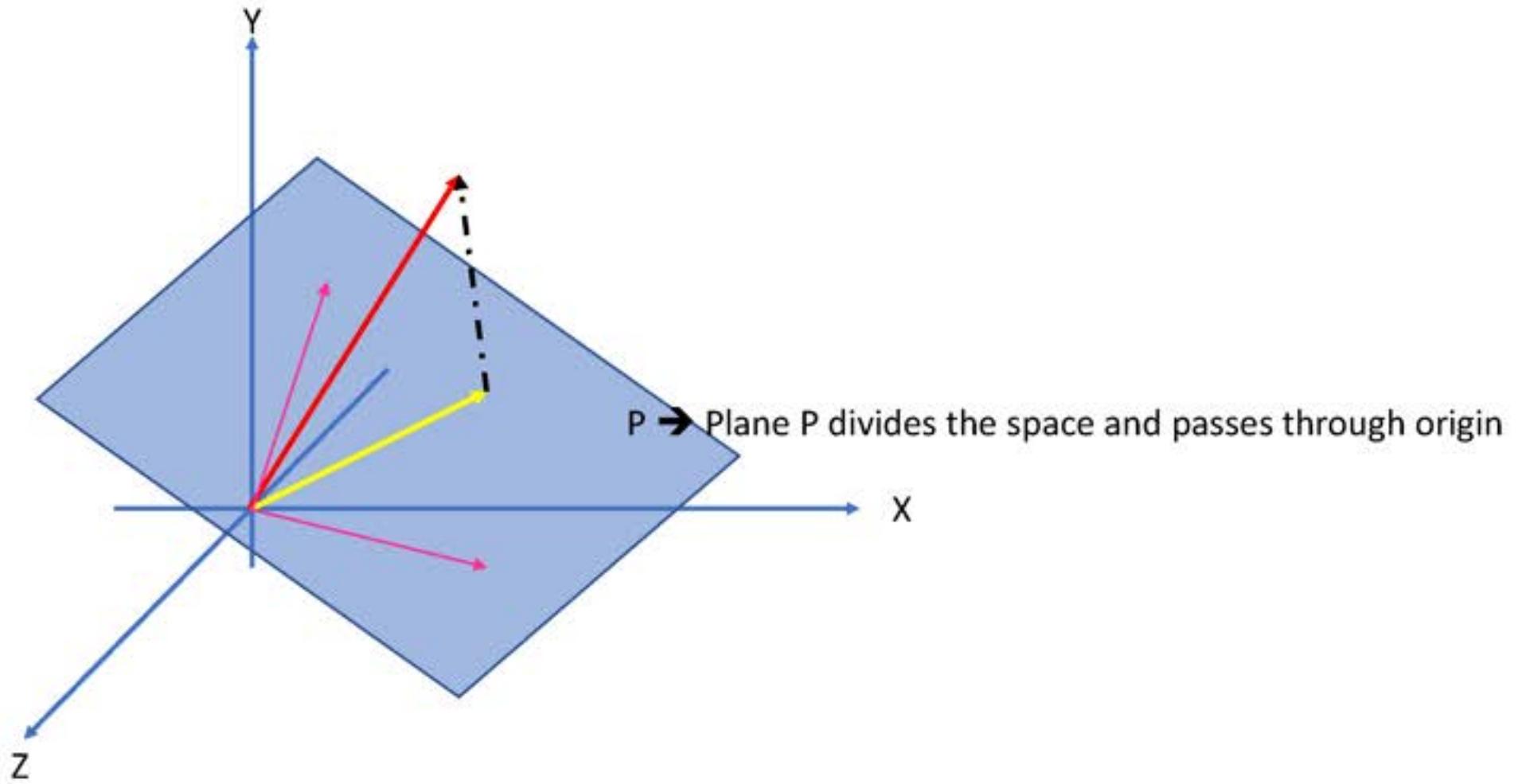
Projections



Point 'P' is represented as (x, y) in X-Y coordinate system

The same point 'P' is presented as (x_1, y_1) in B1-B2 coordinate system.

Projections in Higher Dimensions (Ex: 3D)



Independence

$$A = [v_1 \quad \dots \quad v_n]$$

Let's assume

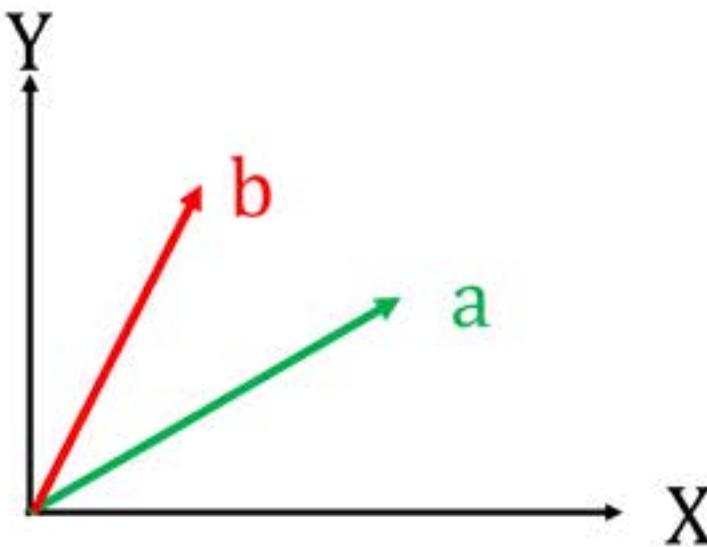
$$\sum_{k=0}^n p_k v_k = 0$$

$$\Rightarrow v_j = \sum_{\substack{k=0 \\ k \neq j}}^n p_k v_k$$

A vector can be represented as linear combination of other vectors

The set of vectors $V = \{v_1, v_2, \dots, v_n\}$ are said to be independent when

$$\sum_{k=0}^n p_k v_k \neq 0$$



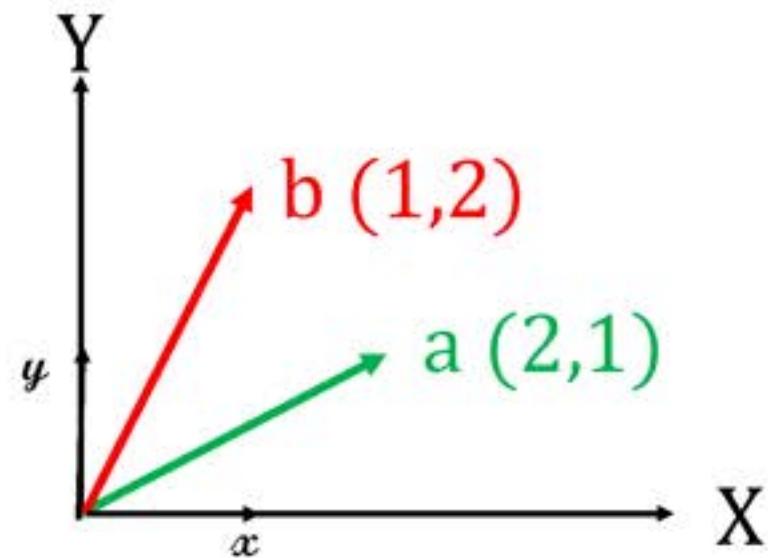
Independence vs Orthogonality

Dot Product

$$a = \{a_1, a_2, a_3, \dots, a_n\}$$

$$b = \{b_1, b_2, b_3, \dots, b_n\}$$

$$a \cdot b = a_1 b_1 + a_2 b_2 + a_3 b_3 + \dots + a_n b_n$$



$$a \cdot b = 2 \times 1 + 1 \times 2 = 4$$

b can never be expressed fully by **a**; hence they are independent.

x is unit vector along X; $x = (1,0)$

y is unit vector along Y; $y = (0,1)$

$$x \cdot y = 1 \times 0 + 0 \times 1 = 0 + 0 = 0$$

Thus, **x** & **y** are orthogonal vectors

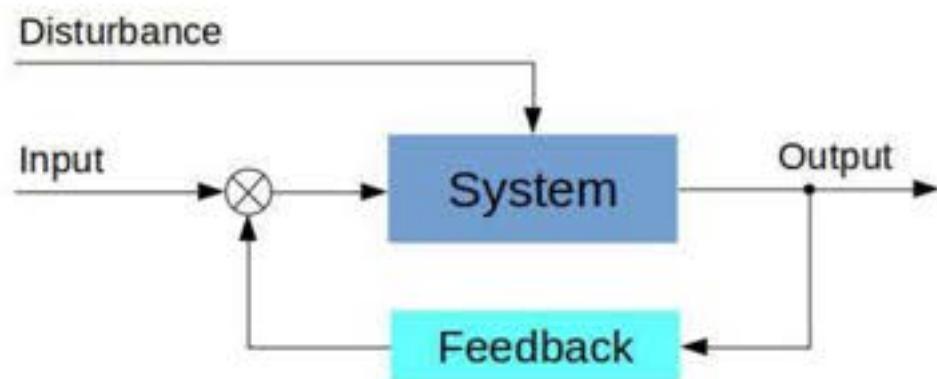
Observability & Controllability

Observable Systems

- System is said to be observable if the behavior of the system can be observed with input-output relationships
- It's also the ability to infer the internal states based on observed outputs

Controllable Systems

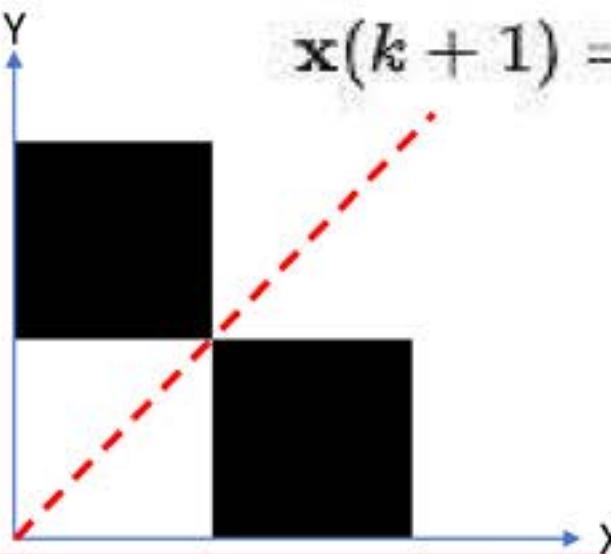
- Controllable system can move from an initial state to a desired end state in finite time



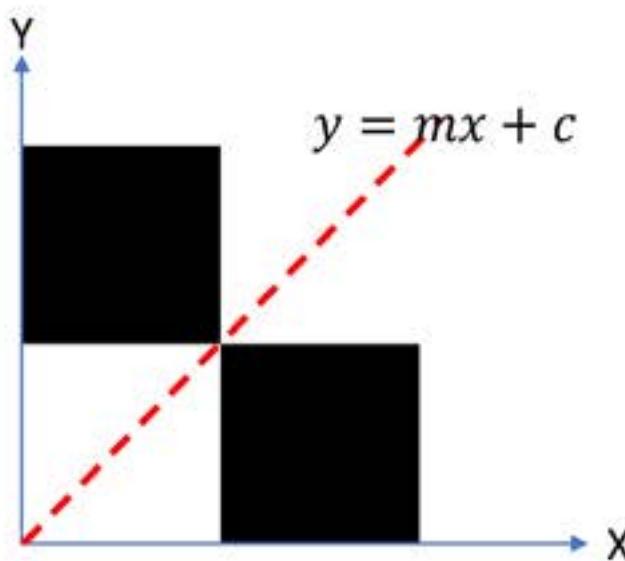
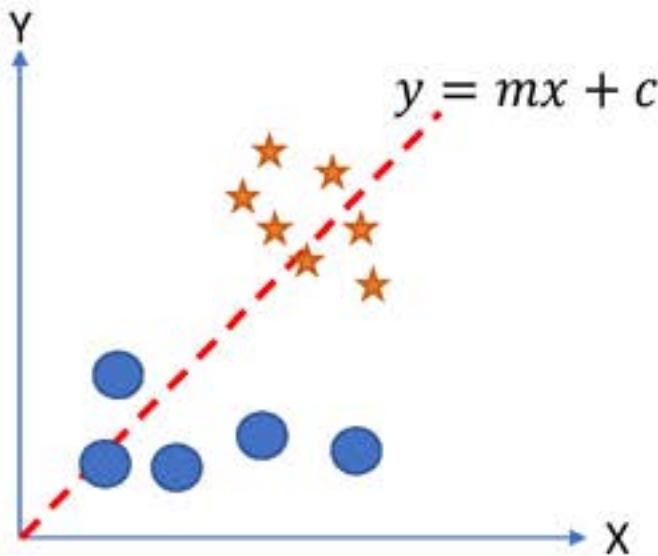
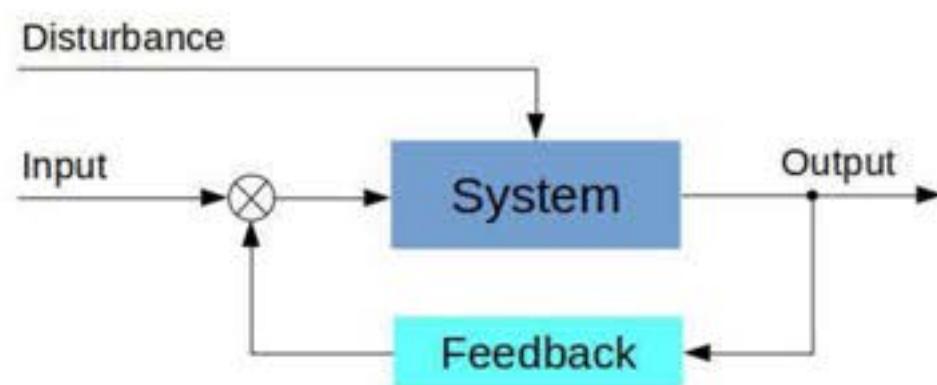
$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t)$$

$$\mathbf{y}(t) = C\mathbf{x}(t) + D\mathbf{u}(t)$$

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k)$$

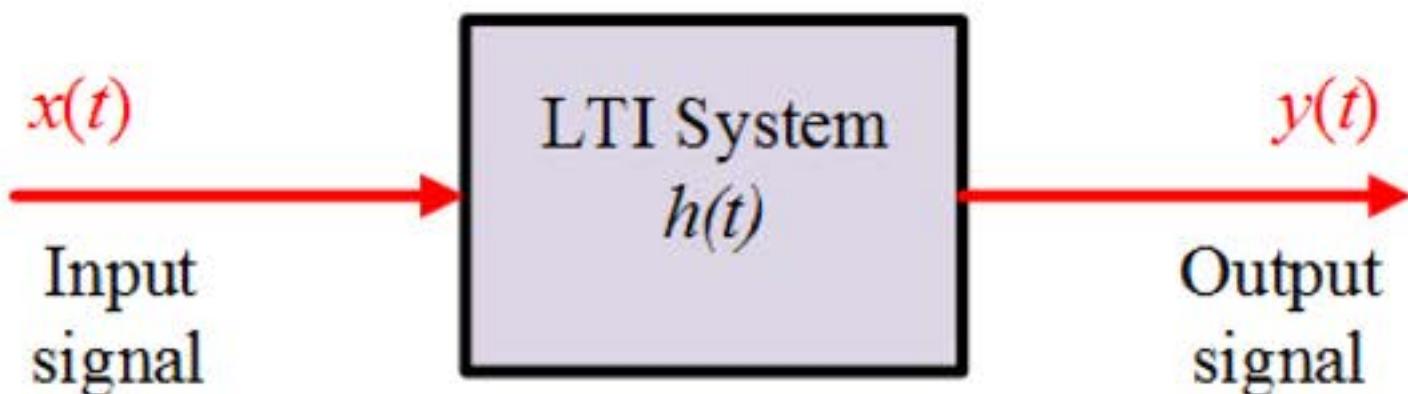


Observability & Controllability



Linear Time-Invariant System

- The output of the system depends on the input and impulse response
- Causal system – current output is from current and past inputs
- Non-causal system – current output from future input



Thank you !!!!!



Machine Learning (19CSE305)

Seperability, Systems, Data Preprocessing

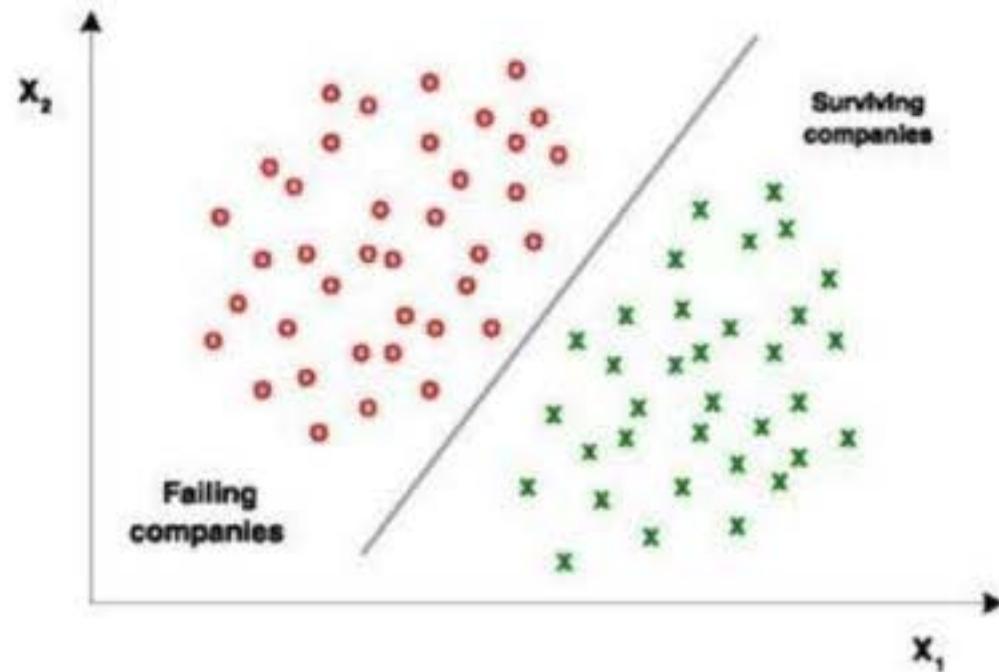


Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

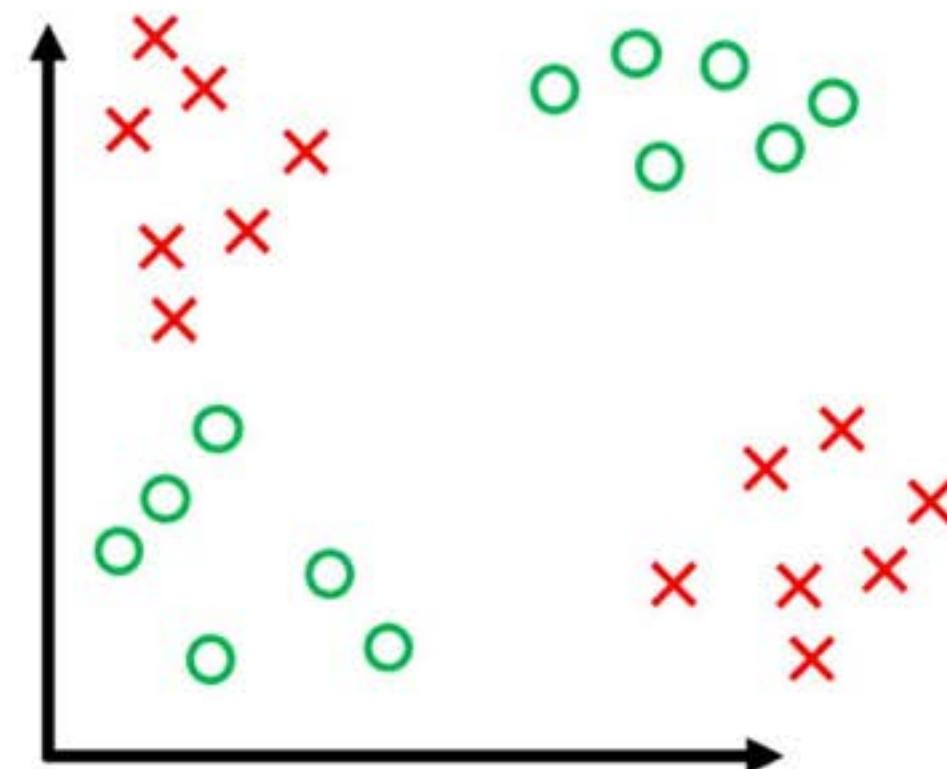
Topics

- Linearly seperable & non-seperable data
- Observable and Controllable
- Time Invariant Systems
- Data Preprocessing

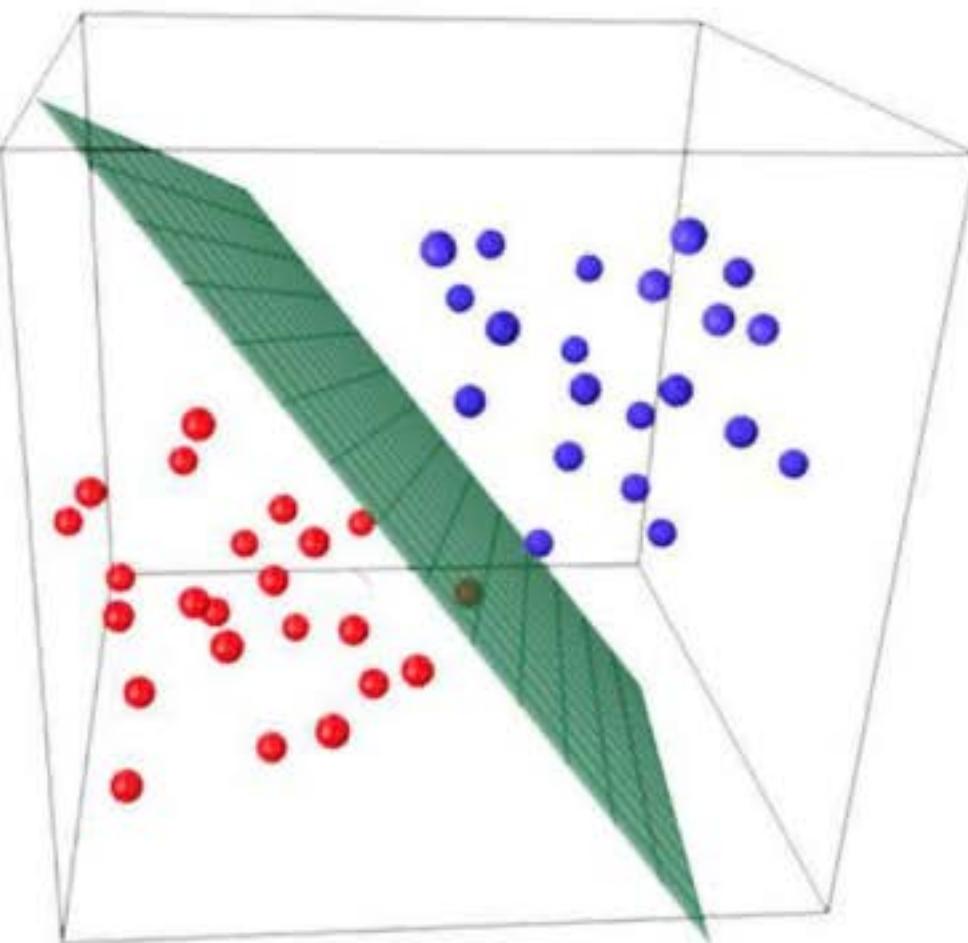
Linearly separable data



Linearly non separable data



Linearly separable



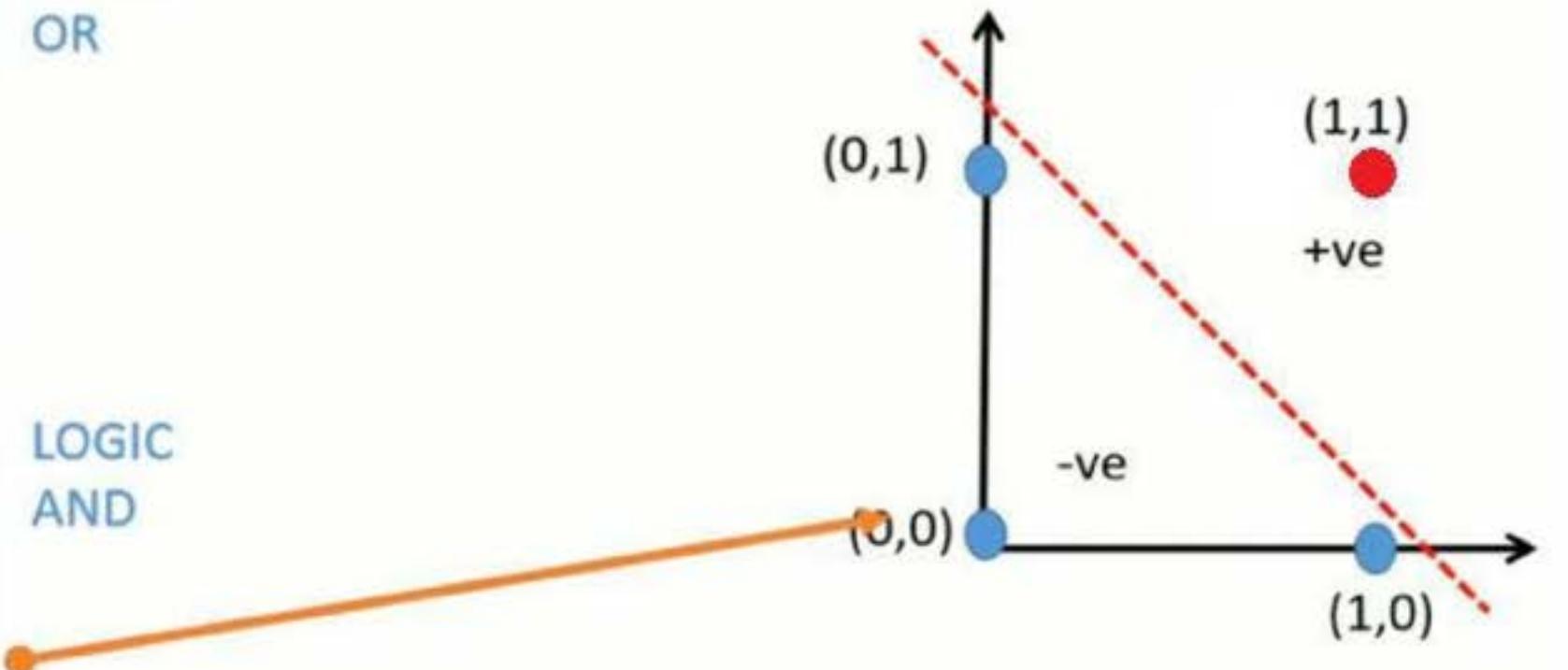
Linearly Separable

x1	x2	y
1	1	1
1	0	1
0	1	1
0	0	0

LOGIC
OR

x1	x2	y
1	1	1
1	0	0
0	1	0
0	0	0

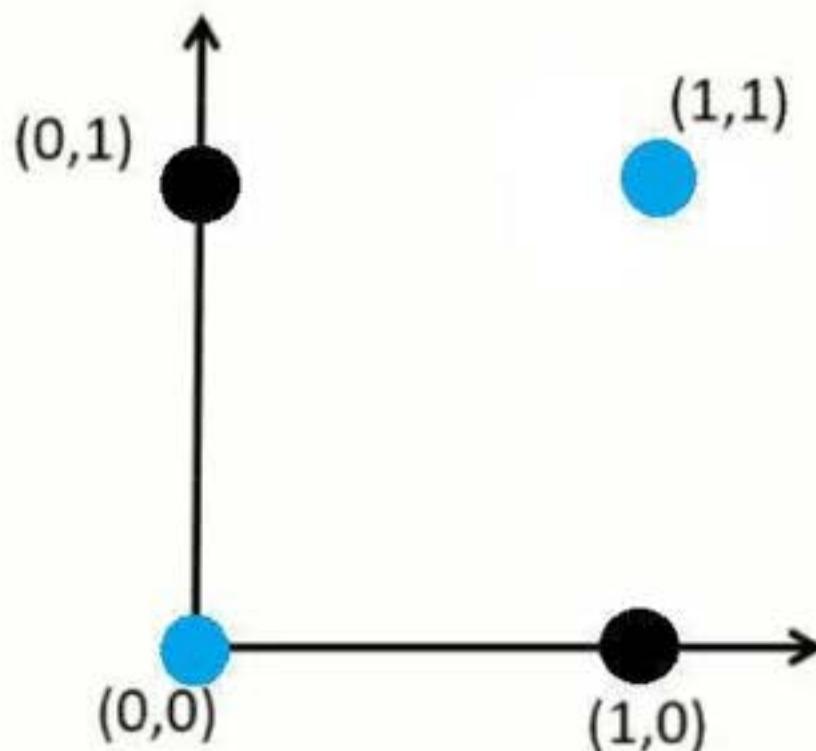
LOGIC
AND



Linearly non-separable

x1	x2	y
1	1	0
1	0	1
0	1	1
0	0	0

LOGIC
XOR



can't separate using single line

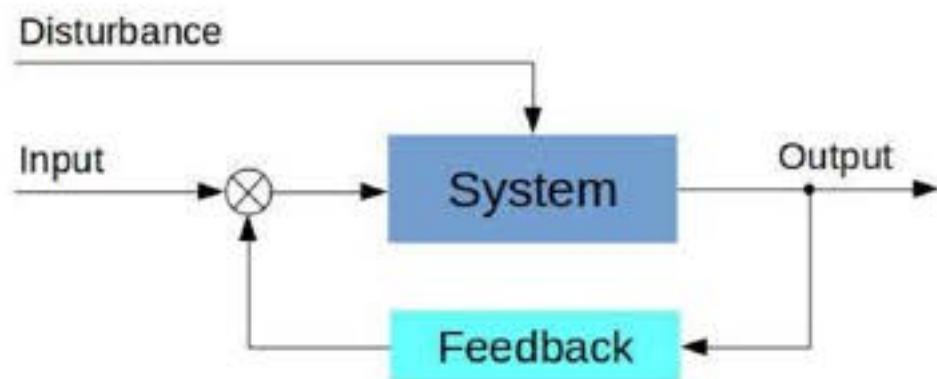
Observability & Controllability

Observable Systems

- System is said to be observable if the behavior of the system can be observed with input-output relationships
- It's also the ability to infer the internal states based on observed outputs

Controllable Systems

- Controllable system can move from an initial state to a desired end state in finite time

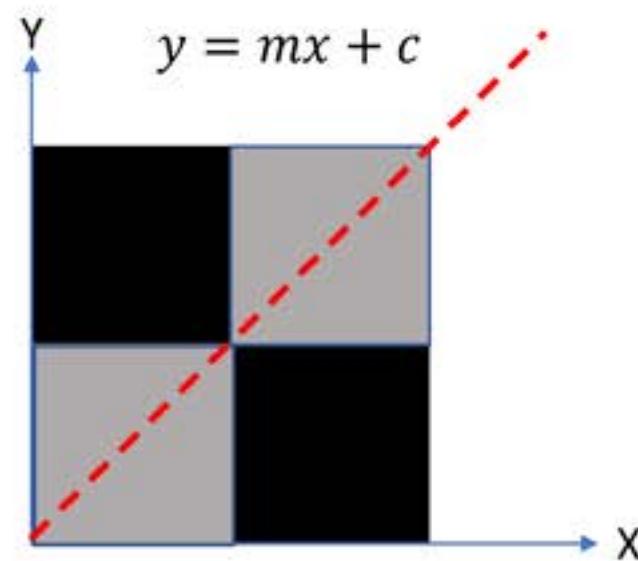
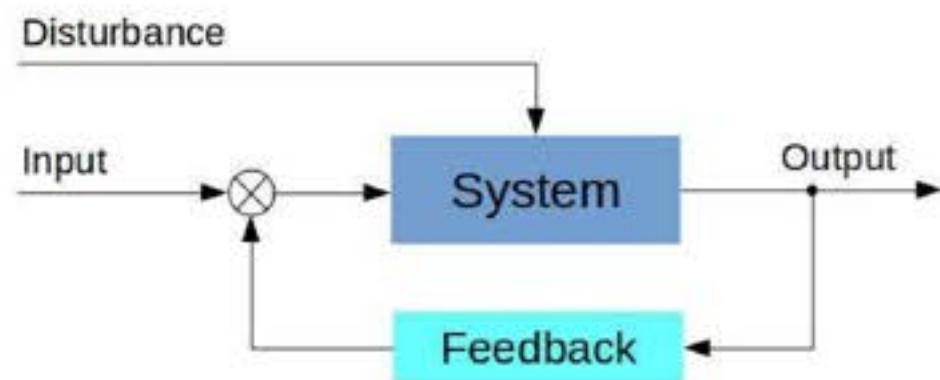
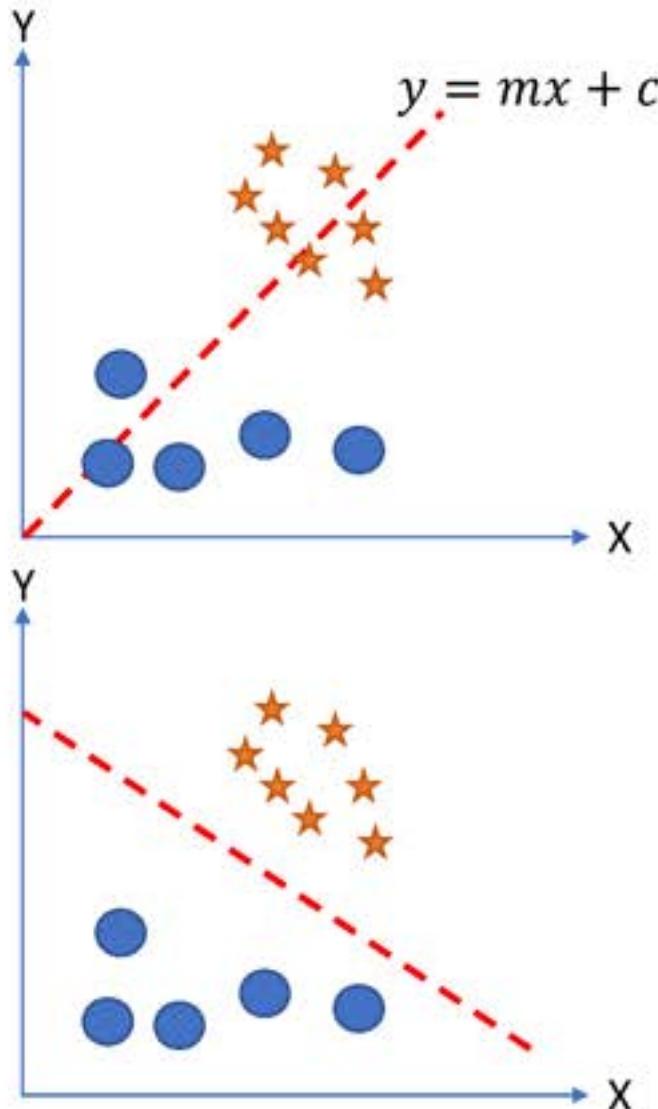


$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t)$$

$$\mathbf{y}(t) = C\mathbf{x}(t) + D\mathbf{u}(t)$$

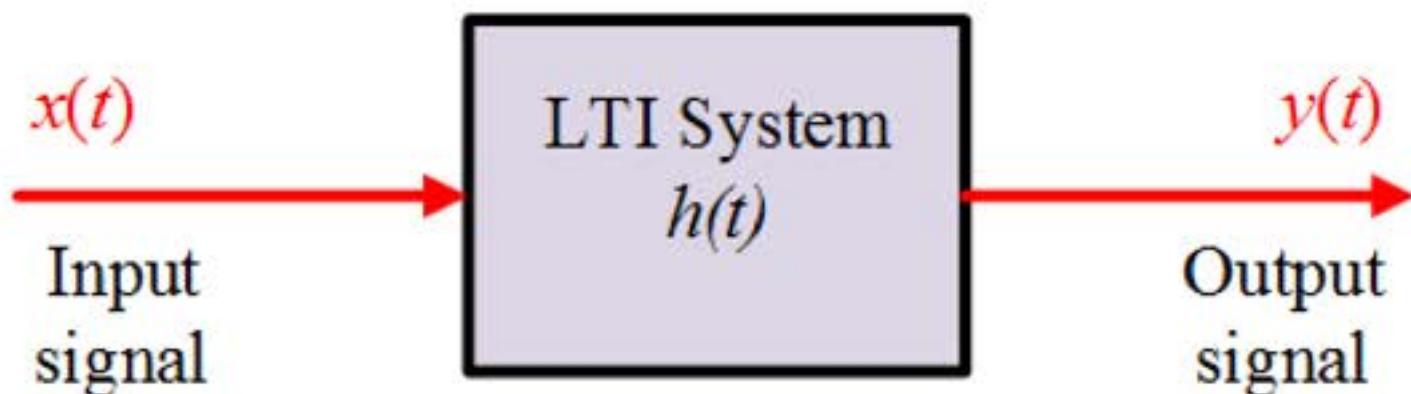
$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k)$$

Observability & Controllability



Linear Time-Invariant System

- The output of the system depends on the input and impulse response
- Causal system – current output is from current and past inputs
- Non-causal system – current output from future input



Data Preprocessing

Data Cleaning

- Data in the Real World Is Dirty: Lots of potentially incorrect data, e.g., instrument faulty, human or computer error, transmission error
 - incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
 - e.g., *Occupation*=“ ” (missing data)
 - noisy: containing noise, errors, or outliers
 - e.g., *Salary*=“-10” (an error)
 - inconsistent: containing discrepancies in codes or names, e.g.,
 - *Age*=“42”, *Birthday*=“03/07/2010”
 - Was rating “1, 2, 3”, now rating “A, B, C”
 - discrepancy between duplicate records
 - Intentional (e.g., *disguised missing data*)
 - Jan. 1 as everyone’s birthday?

How to Handle Missing Data?

- Ignore the tuple: usually done when class label is missing (when doing classification)—not effective when the % of missing values per attribute varies considerably
- Fill in the missing value manually: tedious + infeasible?
- Fill in it automatically with
 - a global constant : e.g., “unknown”, a new class?!
 - the attribute mean
 - the attribute mean for all samples belonging to the same class: smarter
 - mode

Thank you !!!!!



Machine Learning (19CSE305)

Binning ,Discretization & Normalization



Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Binning
- Discretization
- Normalization

Binning

- first sort data and partition into (equal-frequency/equal-width) bins

Equal width binning

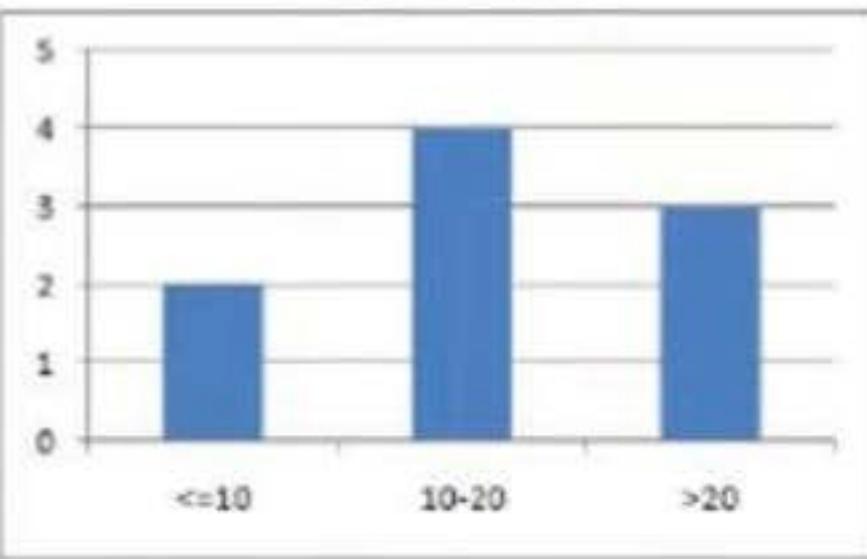
- The algorithm divides the data into k intervals of equal size
- The width of intervals is:
 $w = (\max - \min)/k$
- And the interval boundaries are:
 $\min + w, \min + 2w, \dots, \min + (k-1)w$

Equal frequency binning

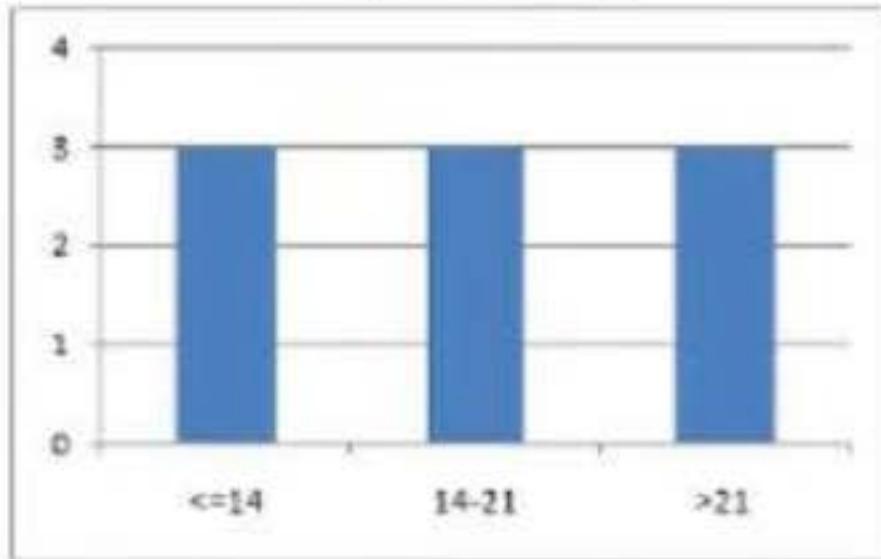
- Divide data into k groups where each group contains approximately same number of values

- Data : 0, 4, 12, 16, 16, 18, 24, 26, 28
- Equal width
 - Bin 1: 0, 4 [-,10)
 - Bin 2: 12, 16, 16, 18 [10,20)
 - Bin 3: 24, 26, 28 [20,*)
- Equal frequency
 - Bin 1: 0, 4, 12 [-, 14)
 - Bin 2: 16, 16, 18 [14, 21)
 - Bin 3: 24, 26, 28 [21, *)

Equal width



Equal frequency



Sorted data for *price* (in dollars): 4, 8, 15, 21, 21, 24, 25, 28, 34

Partition into (equal-frequency) bins:

Bin 1: 4, 8, 15

Bin 2: 21, 21, 24

Bin 3: 25, 28, 34

Smoothing by bin means:

Bin 1: 9, 9, 9

Bin 2: 22, 22, 22

Bin 3: 29, 29, 29

Smoothing by bin boundaries:

Bin 1: 4, 4, 15

Bin 2: 21, 21, 24

Bin 3: 25, 25, 34

Binning Methods for Data Smoothing

- * Sorted data for price (in dollars): 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34
- * Partition into bins:
 - Bin 1: 4, 8, 9, 15
 - Bin 2: 21, 21, 24, 25
 - Bin 3: 26, 28, 29, 34
- * Smoothing by bin means:
 - Bin 1: 9, 9, 9, 9
 - Bin 2: 23, 23, 23, 23
 - Bin 3: 29, 29, 29, 29
- * Smoothing by bin boundaries:
 - Bin 1: 4, 4, 4, 15
 - Bin 2: 21, 21, 25, 25
 - Bin 3: 26, 26, 26, 34

Normalization

Why Normalization?

person_name	Salary	Year_of_experience	Expected Position Level
Aman	100000	10	2
Abhinav	78000	7	4
Ashutosh	32000	5	8
Dishi	55000	6	7
Abhishek	92000	8	3
Avantika	120000	15	1
Ayushi	65750	7	5

The attributes salary and year_of_experience are on different scale and hence attribute salary can take high priority over attribute year_of_experience in the model.

- required when we are dealing with attributes on a different scale, otherwise, it may lead to a dilution in effectiveness of an equally important attribute(on lower scale) because of other attribute having values on larger scale.
- normalizing the data attempts to give all attributes an equal weight.
- particularly useful for classification algorithms involving neural networks or
- distance measurements such as nearest-neighbor classification and clustering

Normalization

- min-max normalization
- z-score normalization
- normalization by decimal scaling

Min-max Normalization

Min-max normalization performs a linear transformation on the original data. Suppose that \min_A and \max_A are the minimum and maximum values of an attribute, A. Min-max normalization maps a value, v_i , of A to v'_i in the range $[new_min_A, new_max_A]$ by computing

- **Min-max normalization:** to $[new_min_A, new_max_A]$

$$v' = \frac{v - \min_A}{\max_A - \min_A} (new_max_A - new_min_A) + new_min_A$$

- Ex. Let income range \$12,000 to \$98,000 normalized to [0.0, 1.0]. Then
\$73,600 is mapped to

$$\frac{73,600 - 12,000}{98,000 - 12,000} (1.0 - 0) + 0 = 0.716$$

Z-score normalization

- **Z-score normalization** (μ : mean, σ : standard deviation):

$$v' = \frac{v - \mu}{\sigma}$$

- Ex. Let $\mu = 54,000$, $\sigma = 16,000$. Then

$$\frac{73,600 - 54,000}{16,000} = 1.225$$

Normalization by decimal scaling

$$v' = \frac{v}{10^j}$$

Where j is the smallest integer such that $\text{Max}(|v'|) < 1$

Decimal scaling. Suppose that the recorded values of A range from -986 to 917 . The maximum absolute value of A is 986 . To normalize by decimal scaling, we therefore divide each value by 1000 (i.e., $j = 3$) so that -986 normalizes to -0.986 and 917 normalizes to 0.917 . ■

Data Discretization

- Continuous values to be discretized
- Height,temperature etc to be discretized
- By different methods like clustering, decision tree algorithm etc

Thank you !!!!!



Machine Learning (19CSE305)

Learning , Hypothesis and testing



Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Supervised vs unsupervised
- Hypothesis and testing

Supervised learning

- The idea of *supervised* learning is that the learning system is given inputs and told which specific outputs should be associated with them.
 - outputs are drawn from a small finite set (classification)
 - a large finite or continuous set (regression).

Classification

- Training data D_n is in the form of a set of pairs
$$\{(x^{(1)}, y^{(1)}), \dots, (x^{(n)}, y^{(n)})\},$$
- where $x^{(i)}$ represents an object to be classified, most typically a d -dimensional vector of real and/or discrete values, and $y^{(i)}$ is an element of a discrete set of values.
- The y values are sometimes called target values.

Classification

- A classification problem is binary or two-class if $y^{(i)}$ is drawn from a set of two possible values; otherwise, it is called multi-class.
- The goal in a classification problem is ultimately, given a new input value $x^{(n+1)}$, to predict the value of $y^{(n+1)}$.
- Classification problems are a kind of supervised learning, because the desired output (or class) $y^{(i)}$ is specified for each of the training examples $x^{(i)}$.
- Regression is like classification, except that $y^{(i)} \in \mathbb{R}^k$.

Unsupervised learning

- Unsupervised learning doesn't involve learning a function from inputs to outputs based on a set of input-output pairs.
- Instead, one is given a data set and generally expected to find some patterns or structure inherent in it.
- Clustering
 - Given $x^{(1)}, \dots, x^{(n)} \in \mathbb{R}^D$,
goal is to find a partitioning (or “clustering”) of the samples that groups together samples that are similar

Hypotheses Testing

- Is one system better than another? With what confidence?
 - Eg.Two network intrusion detection system
- Confidence level -apriori
- What hypothesis to test?
 - Is system better than system2?
 - Is system better than system 2 under high load?
- Exploratory analysis
 - Clustering ,Binning&Histogram analysis,correlation analysis

- Set up an experiment-which variables are important to the question asked.
 - Dependent variable -throughput
 - Independent variable-buffer size
 - Extraneous variable.-time of day
- Avoid sampling bias
 - Checking Algorithm I, Algorithm II better in playing a particular game
 - Average moves taken across games won
 - Picked only samples- where won ---sampling bias

Hypothesis Testing

- Performance on a Population –parameter (eg. avg pred.error on population)
- Test on a sample-statistic(prediction error on the sample)
- Factors affecting –sample size, variance

Hypothesis Testing

- Basic assumption –eg. Both algorithm A & B are same
- Alternate assumption- A is better than B
- Question is should I accept the basic assumption or should I reject in favour of the alternate hypothesis
- If accepted what is the probability it is wrong?

Parameter Estimation

- Eg. average running time
- interval

Thank you !!!!!



Machine Learning (19CSE305)

K-Nearest neighbor

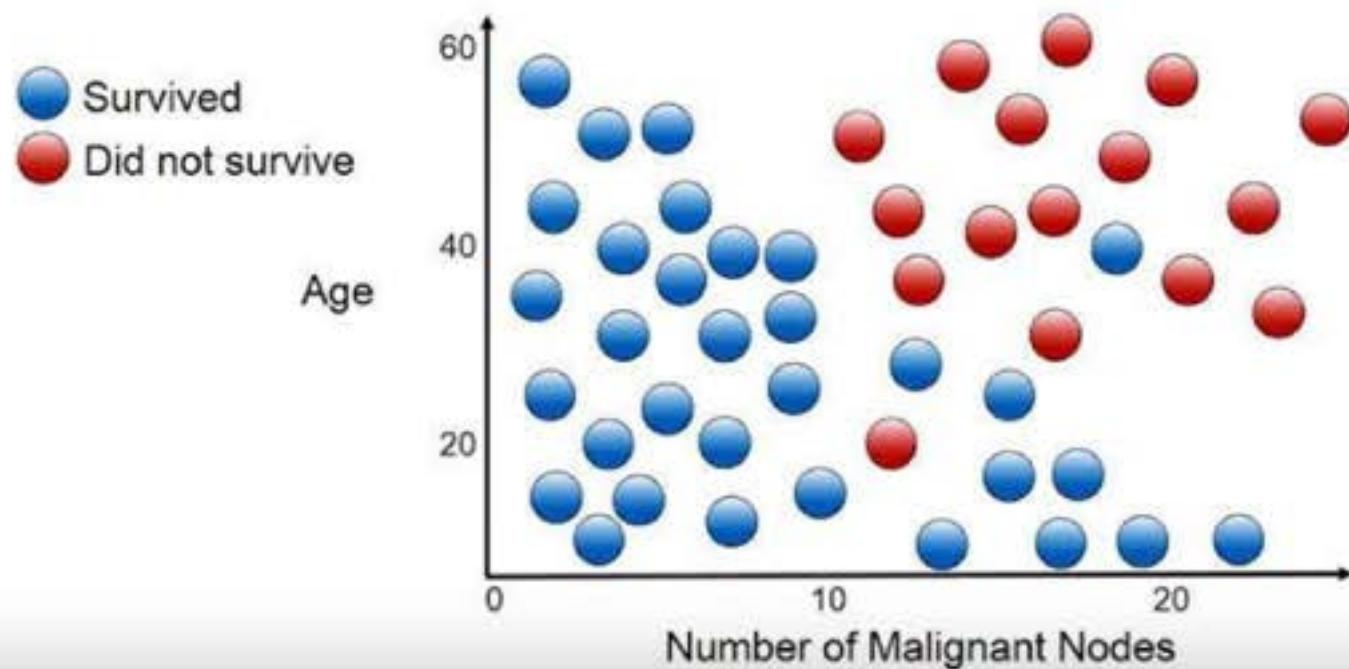


Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

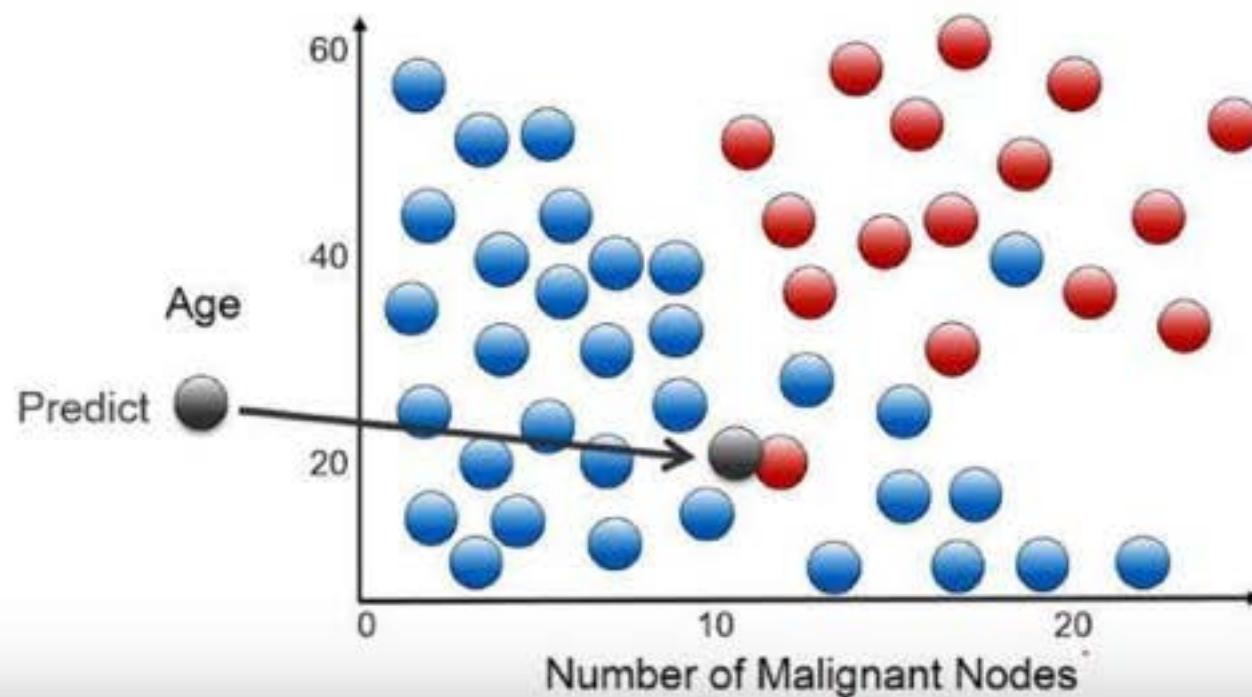
Topics

- K-nearest neighbour classifier
- Distance measure
- Voronoi diagram

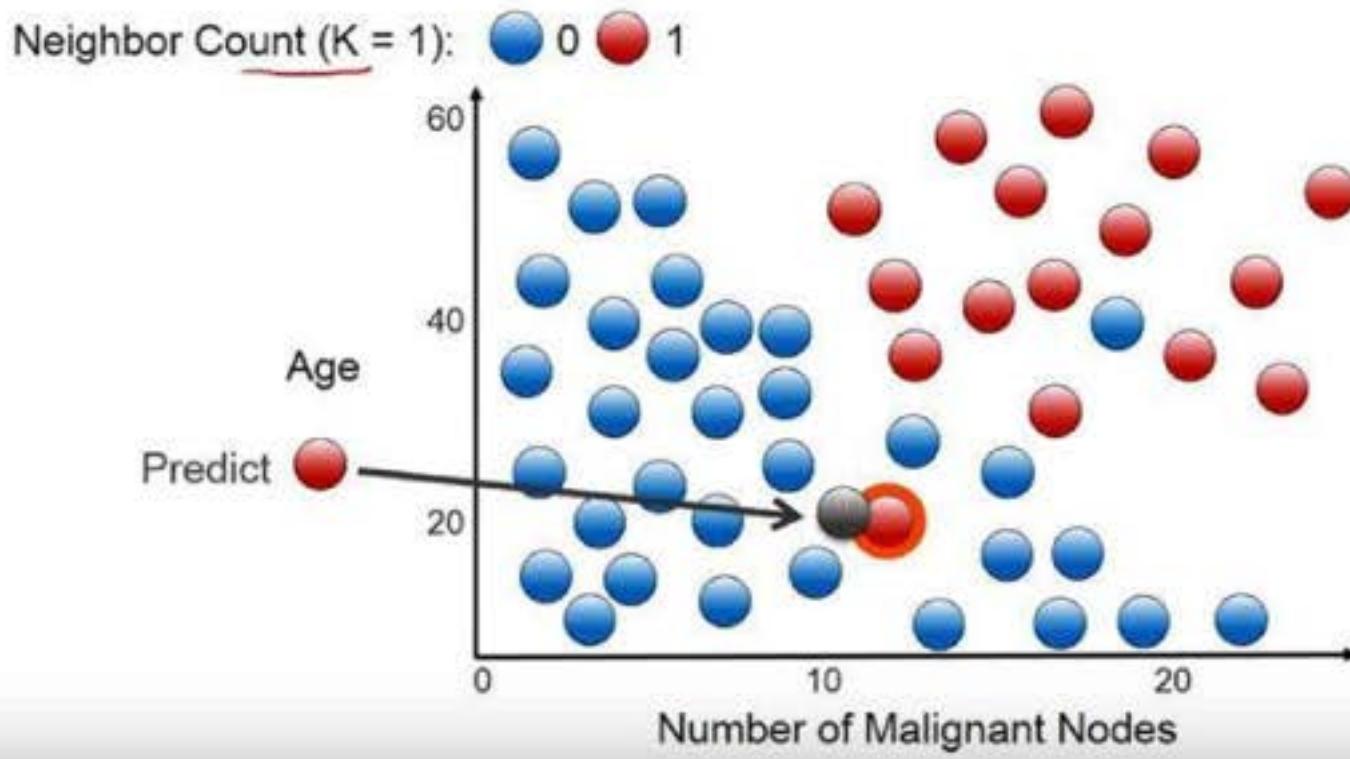
K-Nearest Neighbour- Classification



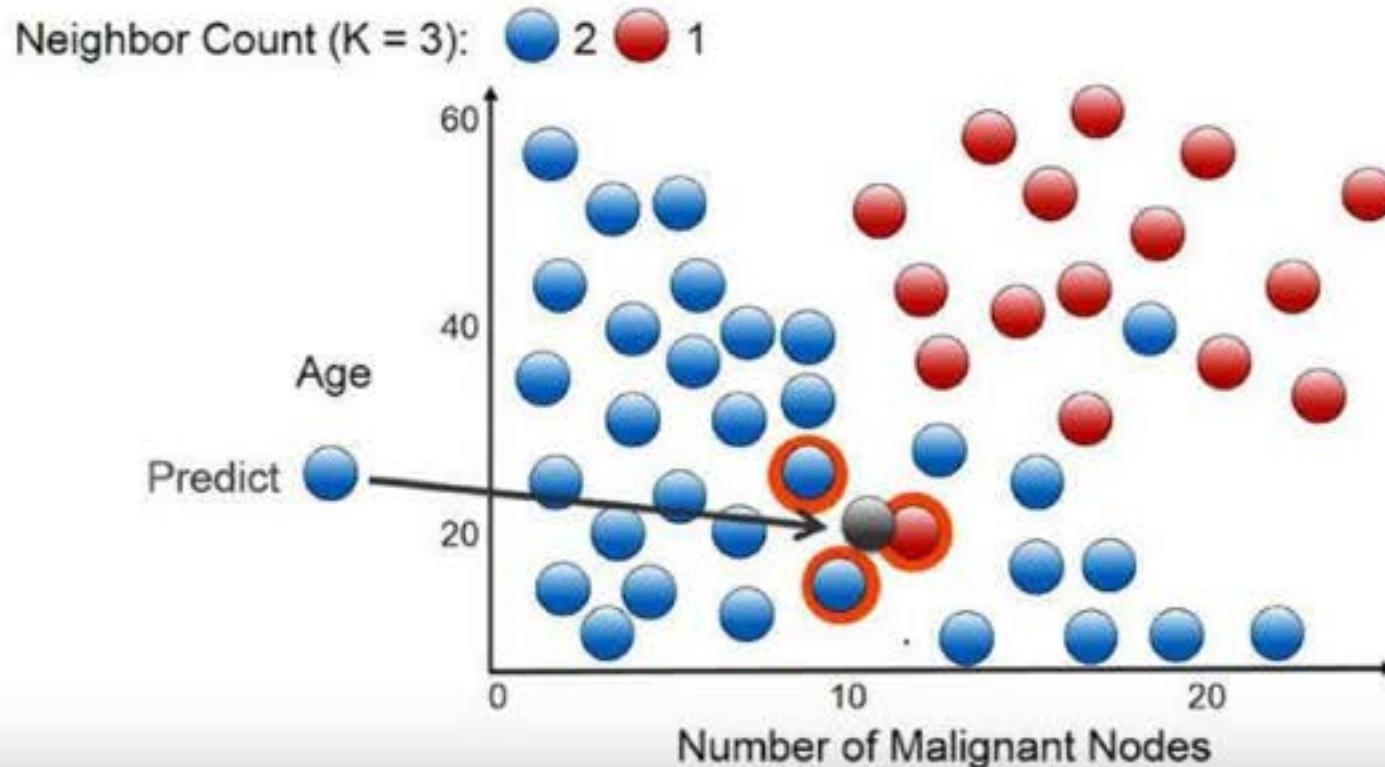
K-Nearest Neighbour- Classification



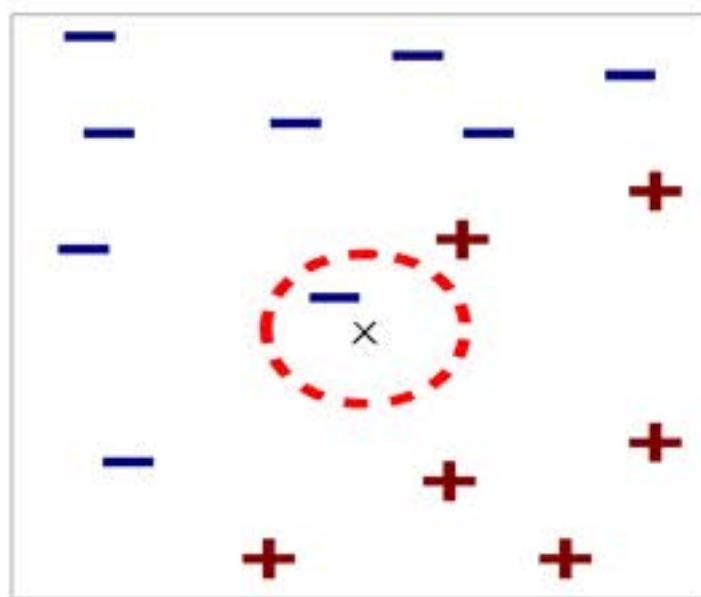
.. Classification



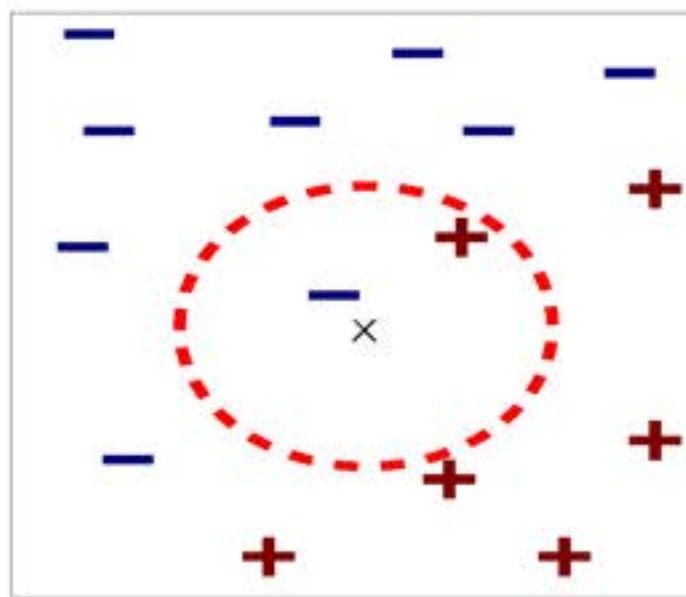
...Classification



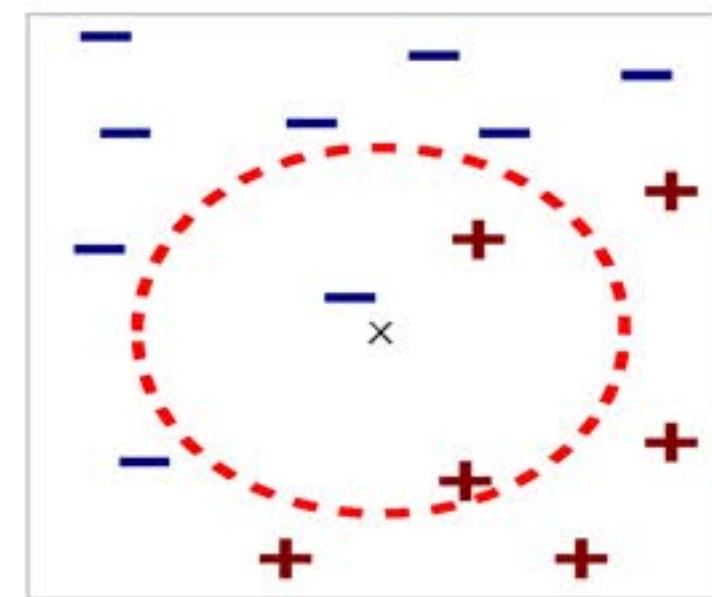
Definition of Nearest Neighbor



(a) 1-nearest neighbor



(b) 2-nearest neighbor



(c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k smallest distance to x

K-nearest neighbour- algorithm

- Training phase : Save the examples
- Prediction phase: Get the test instance

Find the k- training examples

$\{(x_1, y_1), (x_2, y_2), (x_3, y_3) \dots \dots (x_k, y_k)\}$ that is closest to x_t

Classification : predict the majority class from $\{y_1, y_2, y_3 \dots y_k\}$

Regression : predict the average of $\{y_1, y_2, y_3 \dots y_k\}$

Calculating Distance

- Euclidean
- Let A and B are represented by feature vectors $A = (x_1, x_2, \dots, x_n)$ and $B = (y_1, y_2, \dots, y_n)$, where n is the dimensionality of the feature space.

$$dist(A, B) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

Example

We have data from the questionnaires survey (to ask people opinion) and objective testing with two attributes (acid durability and strength) to classify whether a special paper tissue is good or not. Here are four training samples

X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	Y = Classification
7	7	Bad
7	4	Bad
3	4	Good
1	4	Good

Now the factory produces a new paper tissue that passes laboratory test with $X1 = 3$ and $X2 = 7$. Without another expensive survey, can we guess what the classification of this new tissue is?

Assume k = 3

X2 = Strength

X1 = Acid Durability (seconds)

Square Distance to query instance (3, 7)

(kg/square meter)

7

7

$$(7-3)^2 + (7-7)^2 = 16$$

7

4

$$(7-3)^2 + (4-7)^2 = 25$$

3

4

$$(3-3)^2 + (4-7)^2 = 9$$

1

4

$$(1-3)^2 + (4-7)^2 = 13$$

Assume k = 3

X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	Square Distance to query instance (3, 7)	Rank minimum distance	Is it included in 3-Nearest neighbors?
7	7	$(7-3)^2 + (7-7)^2 = 16$	3	Yes
7	4	$(7-3)^2 + (4-7)^2 = 25$	4	No
3	4	$(3-3)^2 + (4-7)^2 = 9$	1	Yes
1	4	$(1-3)^2 + (4-7)^2 = 13$	2	Yes

X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	Square Distance to query instance (3, 7)	Rank minimum distance	Is it included in 3-Nearest neighbors?	Y = Category of nearest Neighbor
7	7	$(7-3)^2 + (7-7)^2 = 16$	3	Yes	Bad
7	4	$(7-3)^2 + (4-7)^2 = 25$	4	No	-
3	4	$(3-3)^2 + (4-7)^2 = 9$	1	Yes	Good
1	4	$(1-3)^2 + (4-7)^2 = 13$	2	Yes	Good

We have 2 good and 1 bad, since $2 > 1$ then we conclude that a new paper tissue that pass laboratory test with $X_1 = 3$ and $X_2 = 7$ is included in Good category.

Voronoi diagram k=1



Lazy vs Eager Learner

- Eager learners
 - when given a set of training tuples, will construct a generalization (i.e., classification) model before receiving new (e.g., test) tuples to classify
- Lazy learners-
 - waits until the last minute before doing any model construction to classify a given test tuple
 - simply stores it (or does only a little minor processing) and waits until it is given a test tuple.
 - Also referred as instance based learner

kNN - advantage

it is relatively straightforward to update the model when new labeled instances become available—we simply add them to the training dataset.

Nearest Neighbor Classification...

- Scaling issues
 - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
 - Example:
 - height of a person may vary from 1.5m to 1.8m
 - weight of a person may vary from 50kg to 110kg
 - income of a person may vary from ₹10K to ₹1crore
 - normalisation

If attributes are non numeric?

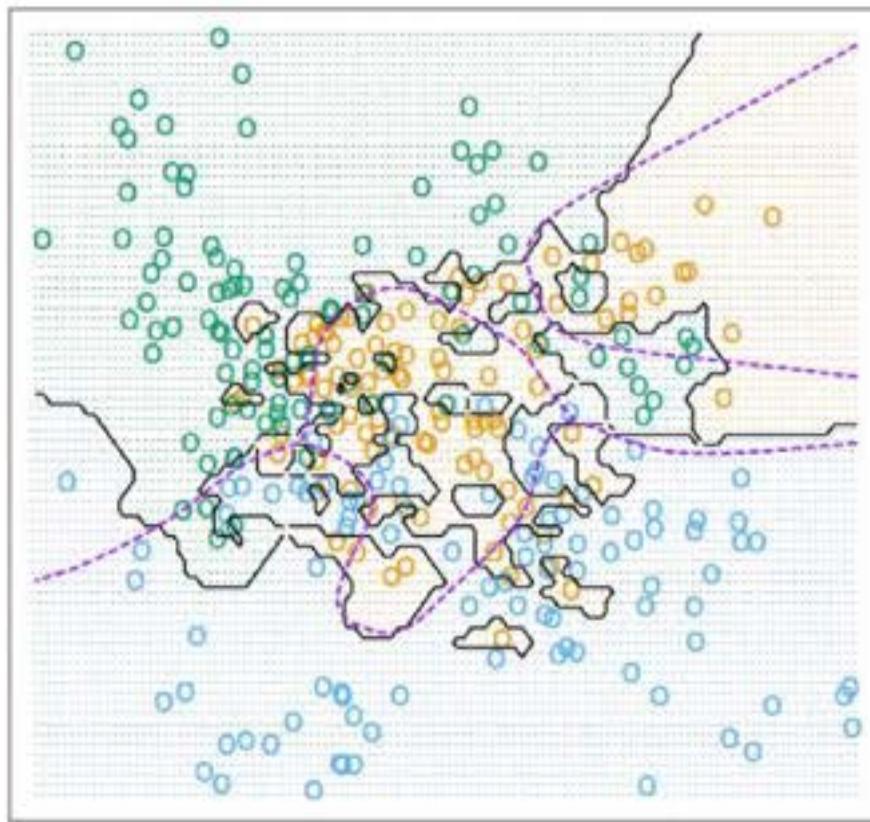
" color?"

- compare the corresponding value of the attribute in tuple X₁ with that in tuple X₂ .
- If the two are identical (e.g., tuples X₁ and X₂ both have the color blue), then the difference between the two is taken as 0.
- If the two are different (e.g., tuple X₁ is blue but tuple X₂ is red), then the difference is considered to be 1.
- Other methods may incorporate more sophisticated schemes for differential grading (e.g., where a larger difference score is assigned, say, for blue and white than for blue and black).

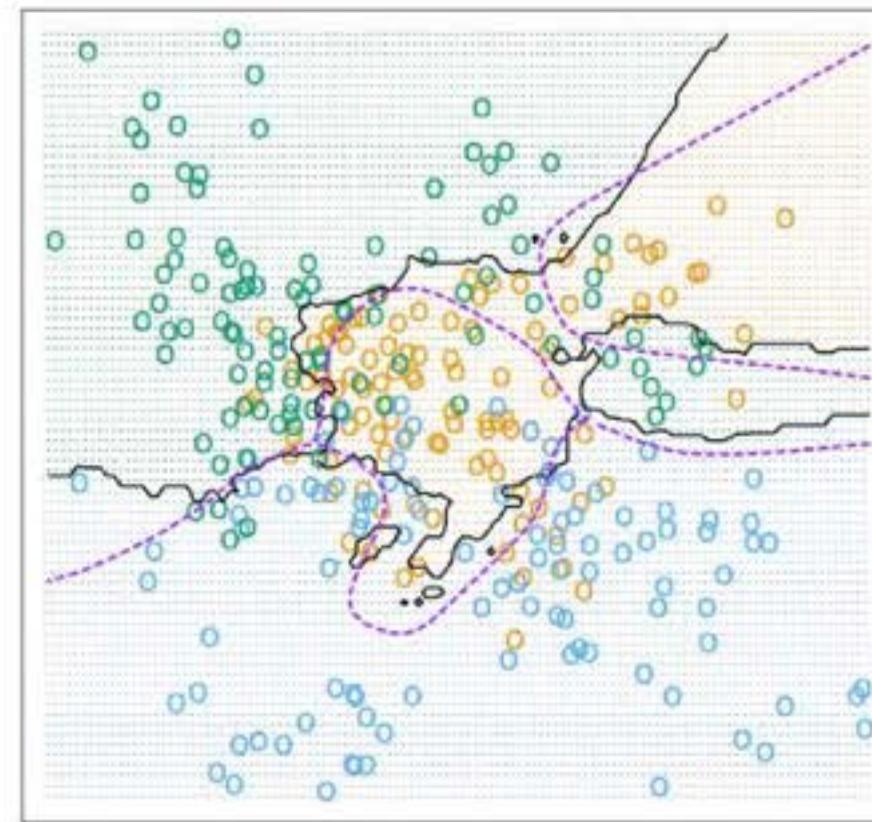
How to determine a good value for k?

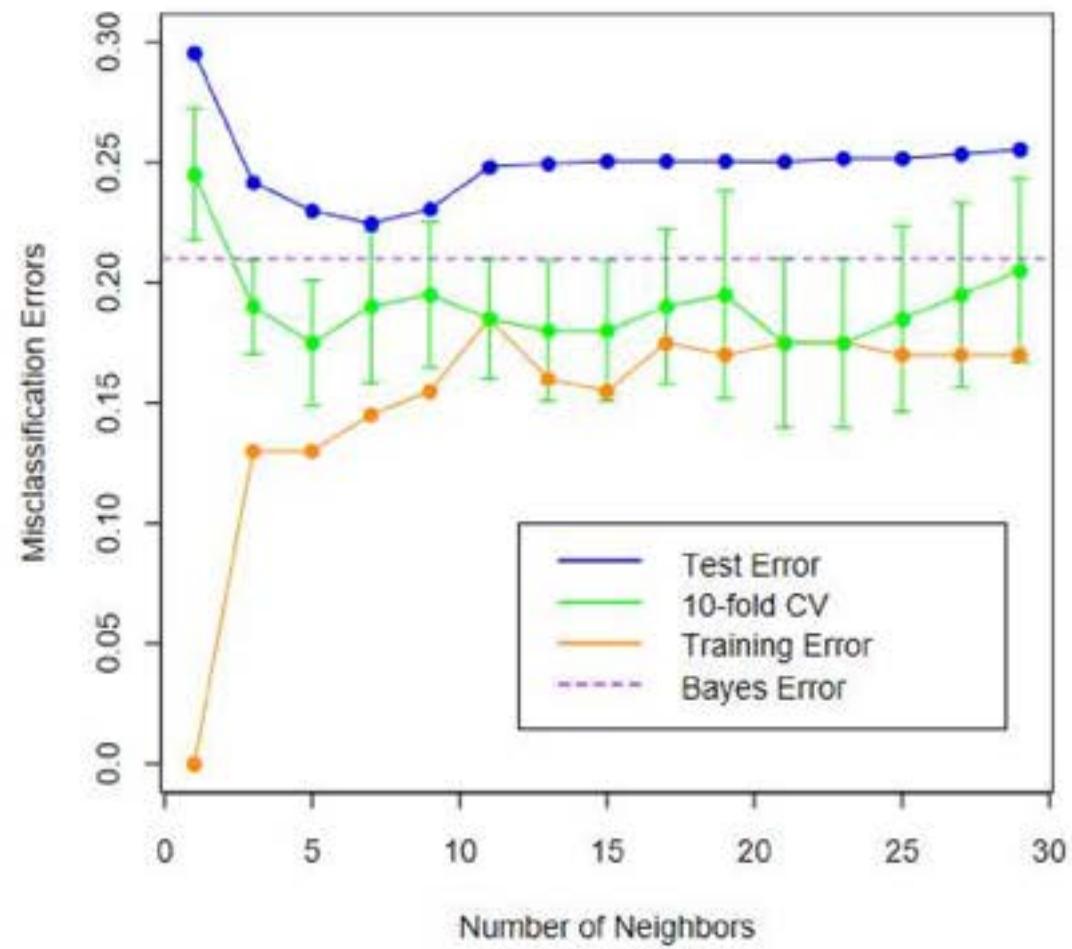
- Starting with $k = 1$, we use a test set to estimate the error rate of the classifier.
- This process can be repeated each time by incrementing k to allow for one more neighbor.
- The k value that gives the minimum error rate may be selected. the larger the number of training tuples, the larger the value of k will be (so that classification and numeric prediction decisions can be based on a larger portion of the stored tuples).

1-Nearest Neighbor

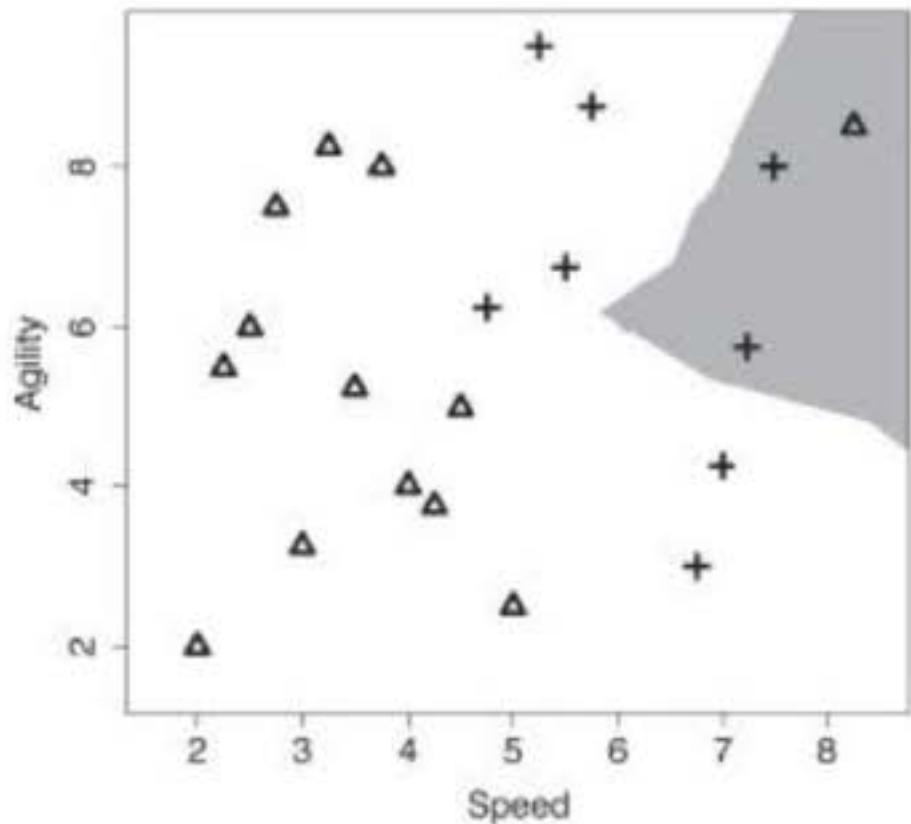


15-Nearest Neighbors





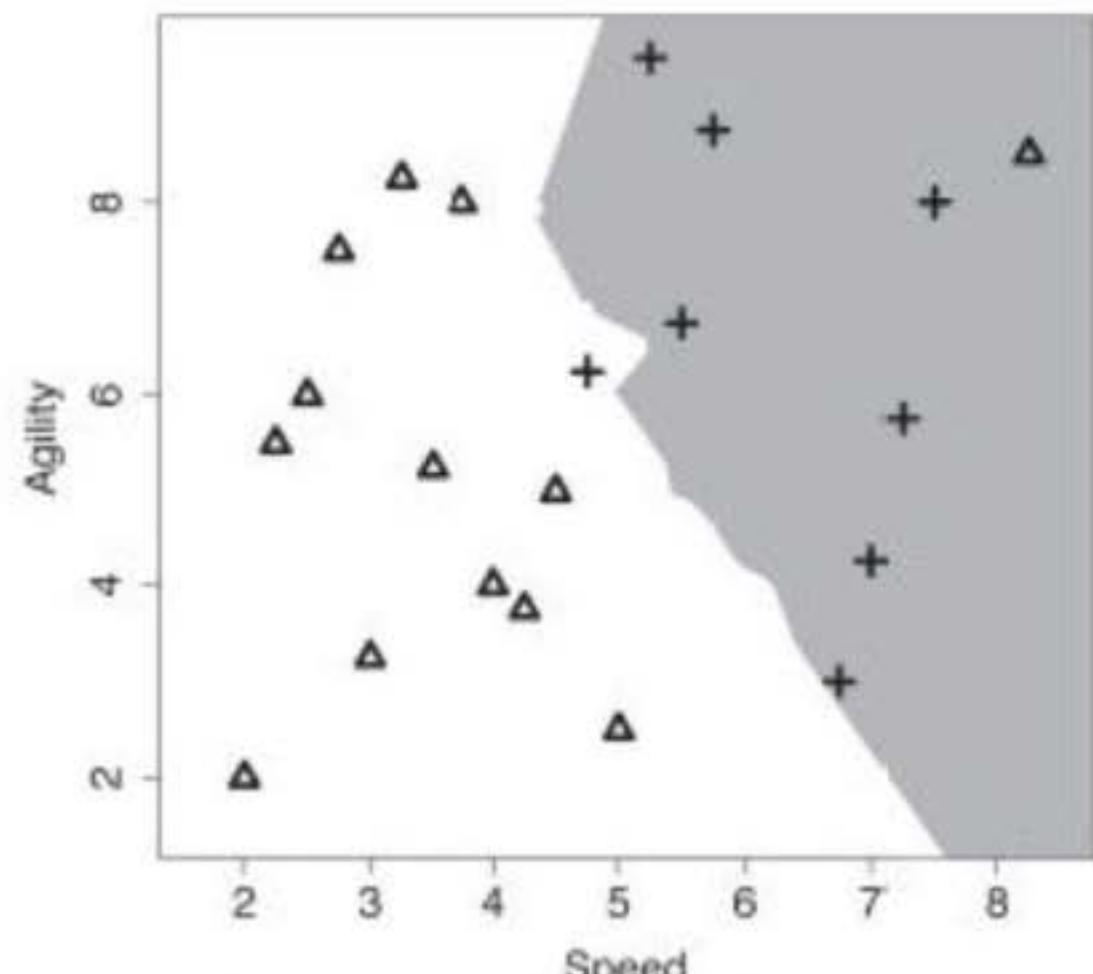
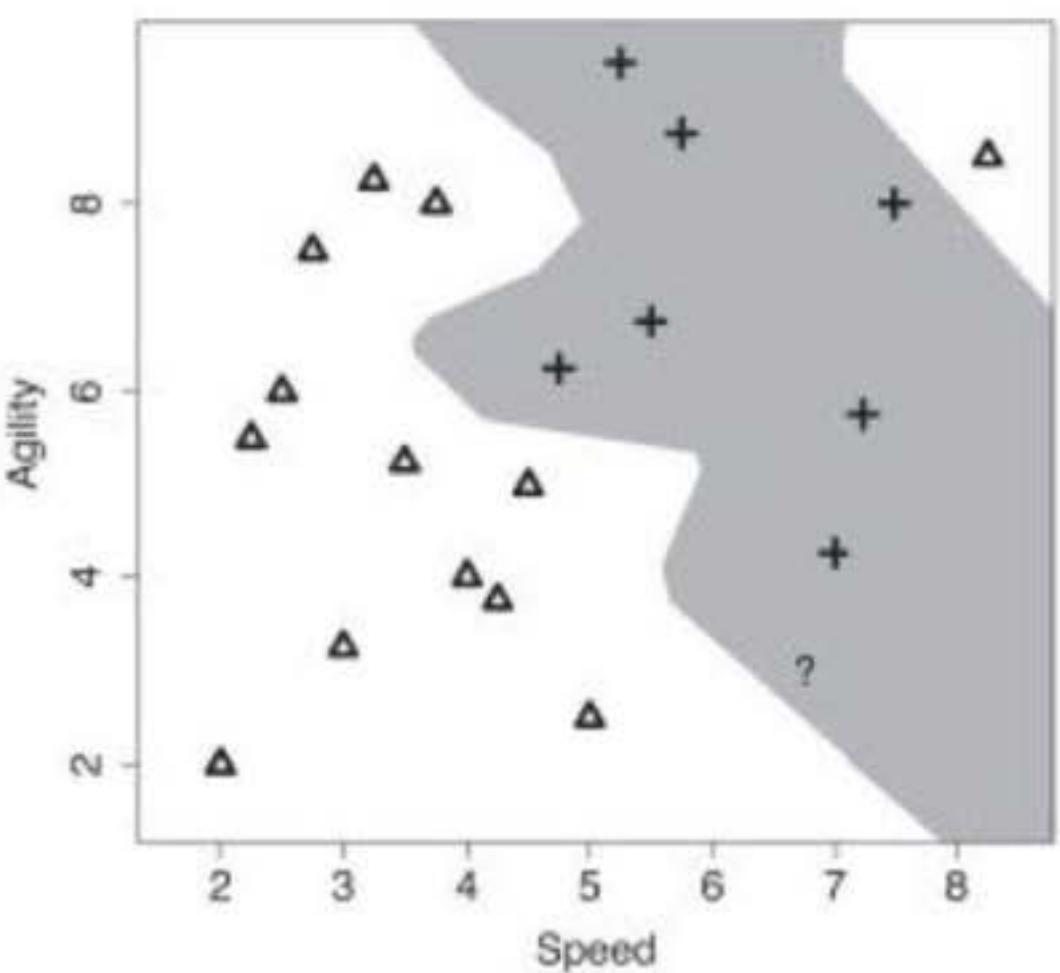
Imbalanced data



(a) Decision boundary ($k = 15$)

- The risks associated with setting k to a high value are particularly acute when we are dealing with an imbalanced dataset.
- as k increases, the majority target level begins to dominate the feature space

Noise in data



(b) Decision boundary ($k = 5$)

Distance-weighted nearest neighbor algorithm

- When a distance weighted k nearest neighbor approach is used, the contribution of each neighbor to the prediction is a function of the inverse distance between the neighbor and the query x_q
 - Give greater weight to closer neighbor

$$w \equiv \frac{1}{d(x_q, x_i)^2}$$

Issues with Distance-weighted knn

- if the dataset is very imbalanced, then even with a weighting applied to the contribution of the training instances, the majority target level may dominate.
- when the dataset is very large, which means that computing the reciprocal of squared distance between the query and all the training instances can become too computationally expensive to be feasible

Improvements

- Weighted Euclidean distance

$$D(c1, c2) = \sqrt{\sum_{i=1}^N w_i \cdot (attr_i(c1) - attr_i(c2))^2}$$

- large weights => attribute is more important
- small weights => attribute is less important
- zero weights => attribute doesn't matter

Efficient Memory Search

- if we are working with a large dataset, the time cost in computing the distances between a query and all the training instances and retrieving the k nearest neighbors may be prohibitive.
- use k-d tree,(k-dimensional tree),.
- A k-d tree is a balanced binary tree in which each of the nodes in the tree (both interior and leaf nodes) index one of the instances in a training dataset.
- The tree is constructed so that nodes that are nearby in the tree index training instances that are nearby in the feature space

Refer for more details

- **FUNDAMENTALS OF MACHINE LEARNING FOR PREDICTIVE DATA ANALYTICS**- Algorithms, Worked Examples, and Case Studies ,John D. Kelleher, Brian Mac Namee, Aoife D'Arcy

Another approach

- speed up classification time include the use of partial distance calculations and editing the stored tuples.
- Compute the distance based on a subset of the n attributes.
- If this distance exceeds a threshold, then further computation for the given stored tuple is halted, and the process moves on to the next stored tuple.
- The editing method removes training tuples that prove useless.
- This method is also referred to as pruning or condensing because it reduces the total number of tuples stored.

Thank you !!!!!



NEURAL NETWORKS

A. SAIRAM SRIKAR - BL.EN.U4CSE19004

AVINASH S - BL.EN.U4CSE19010

K.V.G.ROHITH - BL.EN.U4CSE19073

Topics

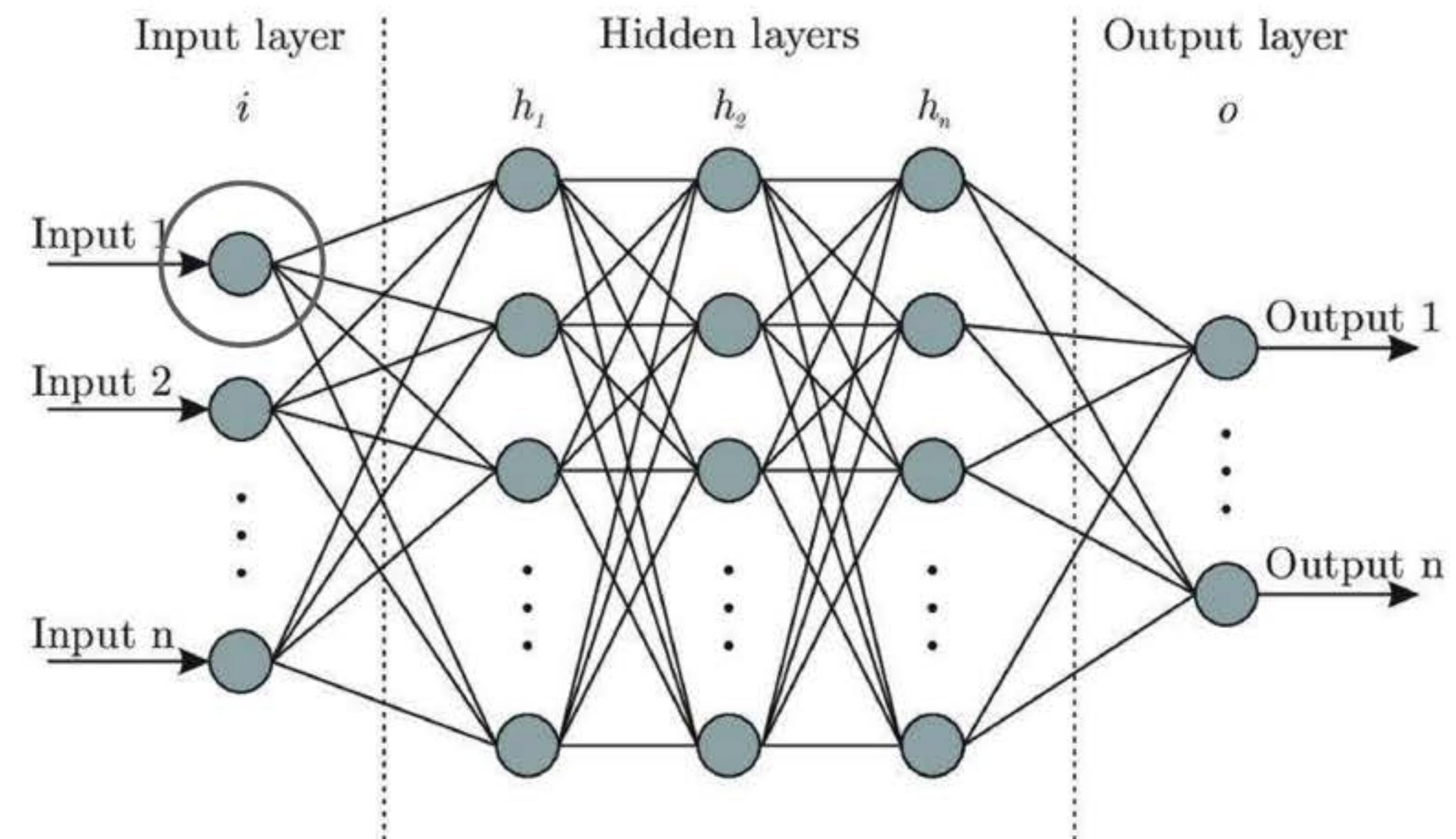
- Introduction
- Structure, Functioning
- Example and Applications
- Perceptron
- Weights and Bias
- Activation Functions
- Types of Activation Functions

WHAT?

- A Machine Learning model.
- Inspired by the human brain.
- A web of interconnected entities known as nodes.
- Also called SNN: Simulated Neural Networks.

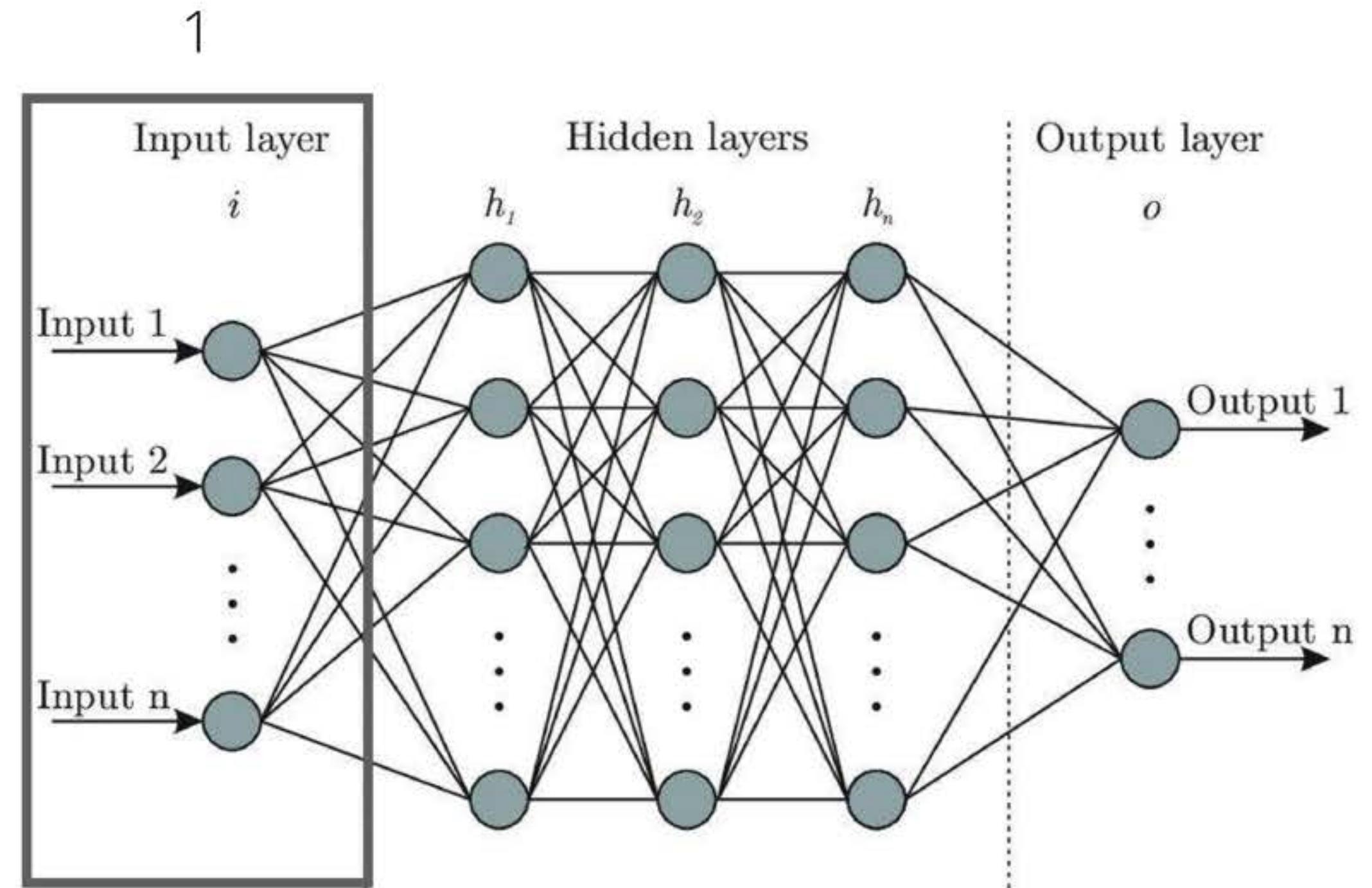
STRUCTURE

- **Artificial Neurons**
- Layers:
 - a. Input Layer
 - b. Hidden Layers
 - c. Output Layer



STRUCTURE

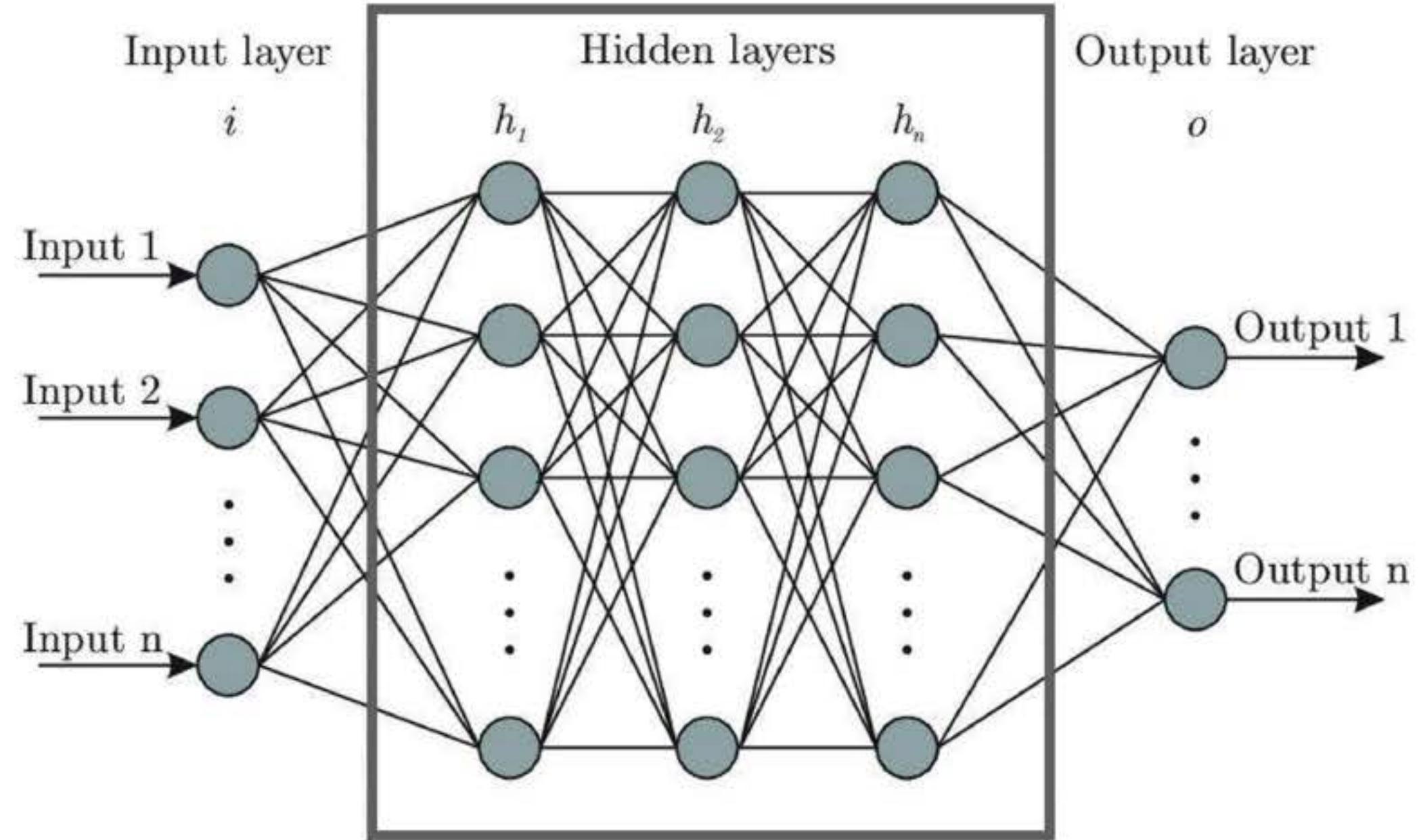
- Artificial Neurons
- Layers:
 - a. Input Layer
 - b. Hidden Layers
 - c. Output Layer



STRUCTURE

- Artificial Neurons
- Layers:
 - a. Input Layer
 - b. Hidden Layers
 - c. Output Layer

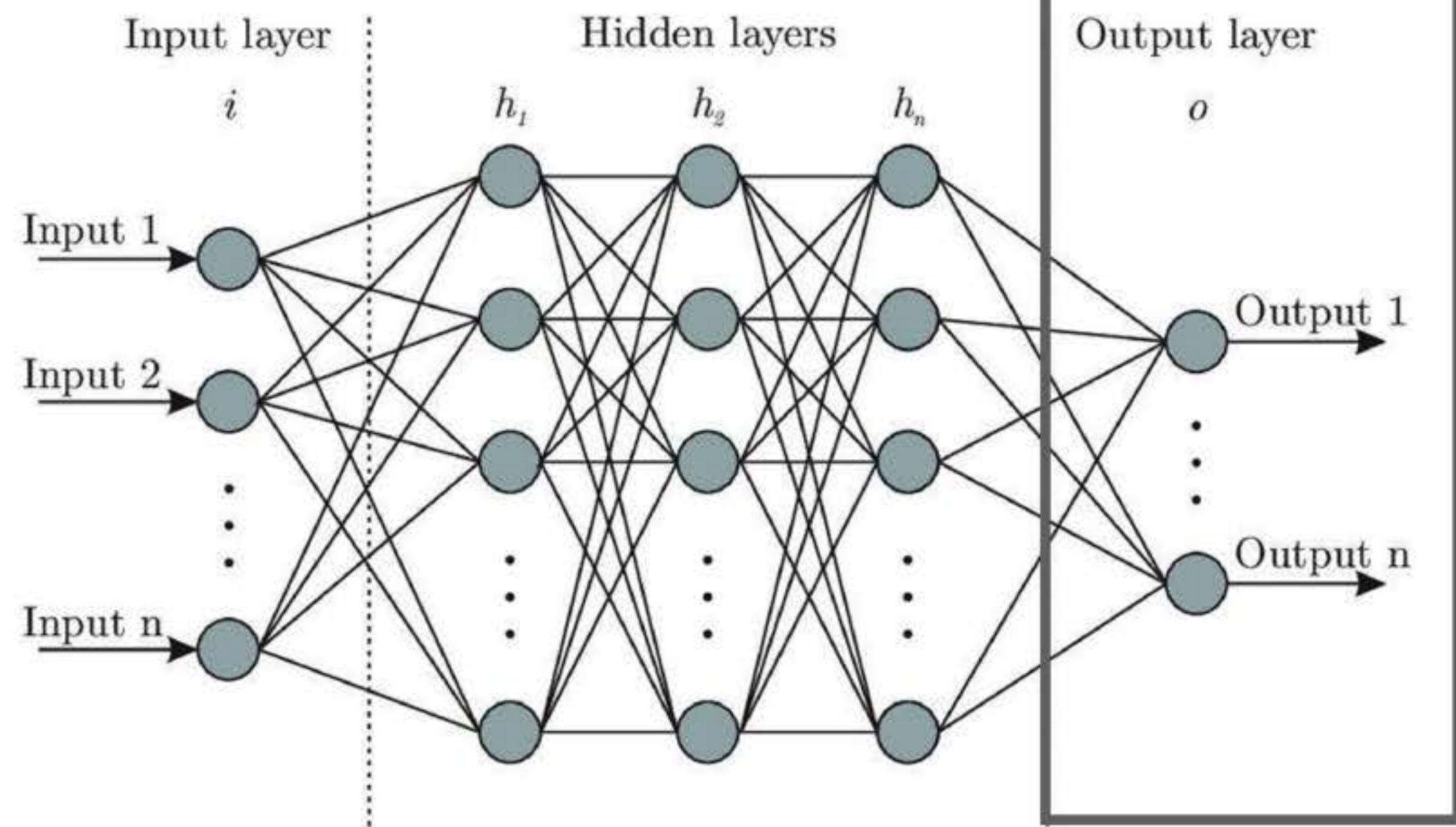
2



STRUCTURE

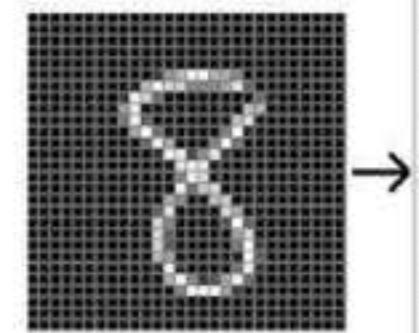
3

- Artificial Neurons
- Layers:
 - a. Input Layer
 - b. Hidden Layers
 - c. Output Layer

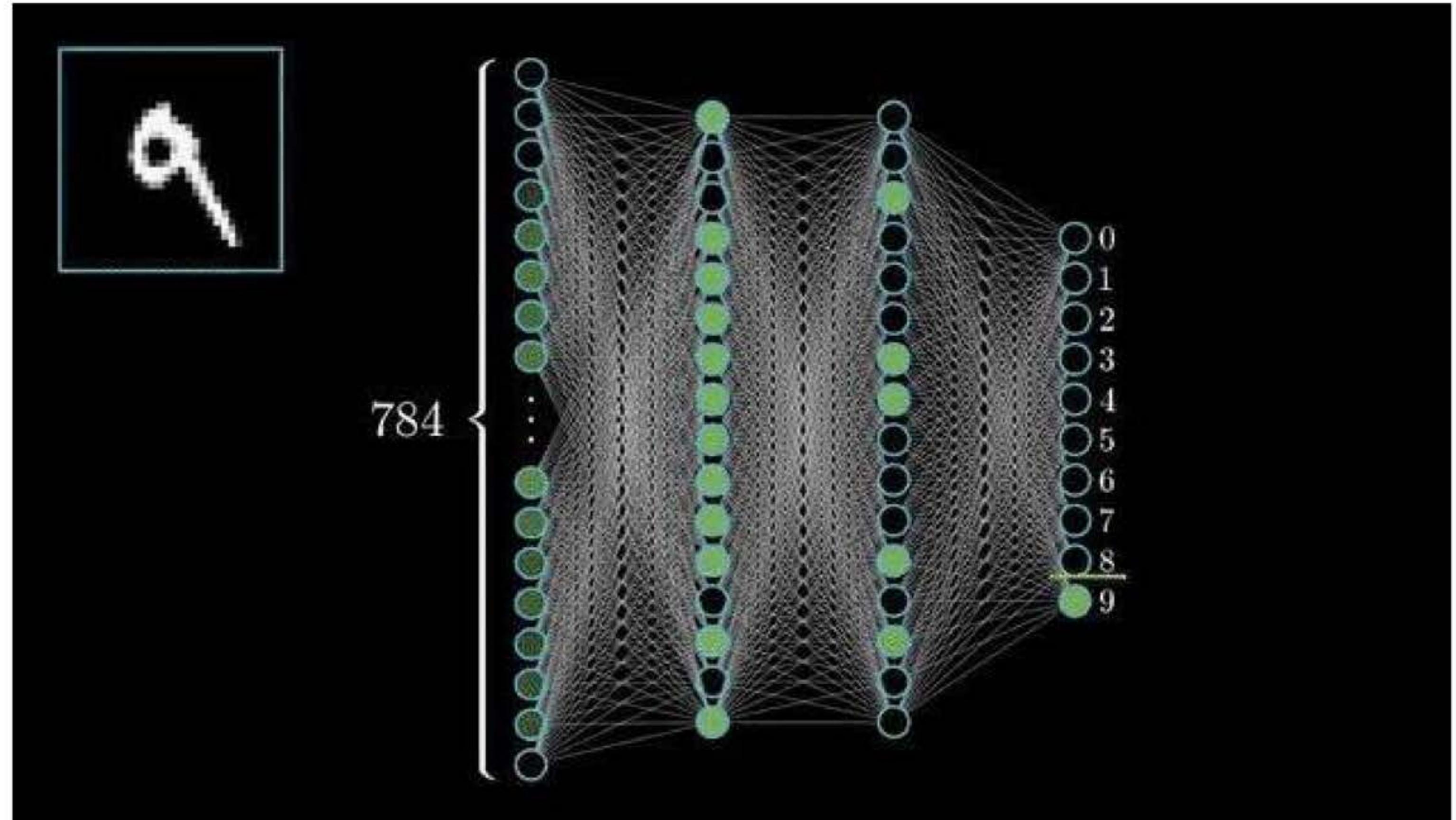


HOW?

0	4	1	9	2	1	3	1	4	3
5	3	6	1	7	2	8	6	9	4
0	9	1	1	2	4	3	2	7	3
8	6	9	0	5	6	0	7	6	1
8	1	9	3	9	8	5	9	3	3
0	7	4	9	8	0	9	4	1	4
4	6	0	4	5	6	1	0	0	1
7	1	6	3	0	2	1	1	7	9
0	2	6	7	8	3	9	0	4	6
7	4	6	8	0	7	8	3	1	5



**28 x 28
784 pixels**



TYPES OF NEURAL NETWORKS

- Feedforward Neural Network
- Recurrent Neural Network(RNN)
- Convolutional Neural Network(CNN)

WHEN TO AND WHEN NOT TO?

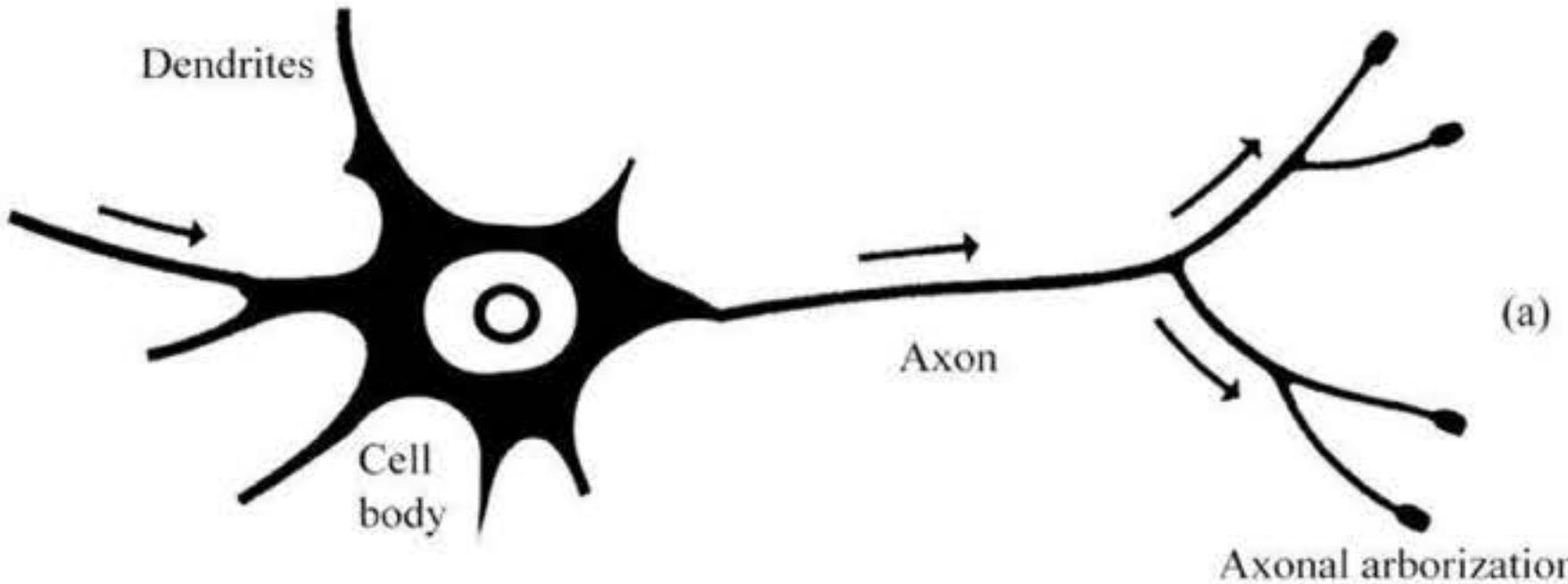
- When data is not well structured and is complex.
- For large amounts of data.



WHERE?

- Face Recognition
- Speech Recognition
- Fraud Detection
- Signature Verification
- Image Processing
- Forecasting

Comparison



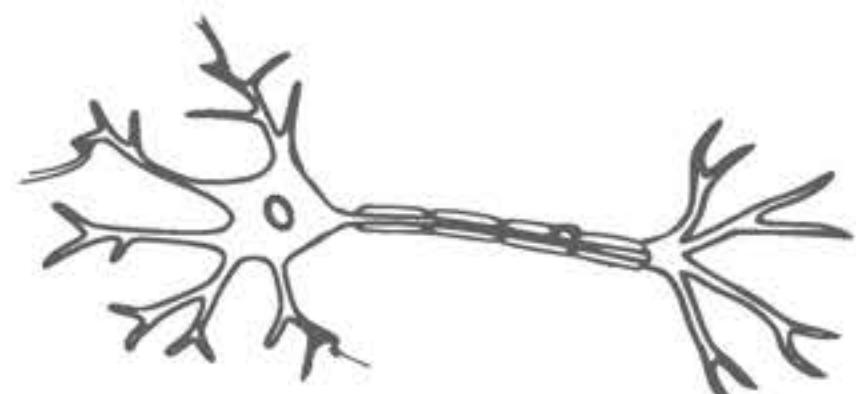
Neurons are nerve impulse-conducting cells that make up nerves, brain and spinal column

A neuron is a mathematical function that model the functioning of a biological neuron.

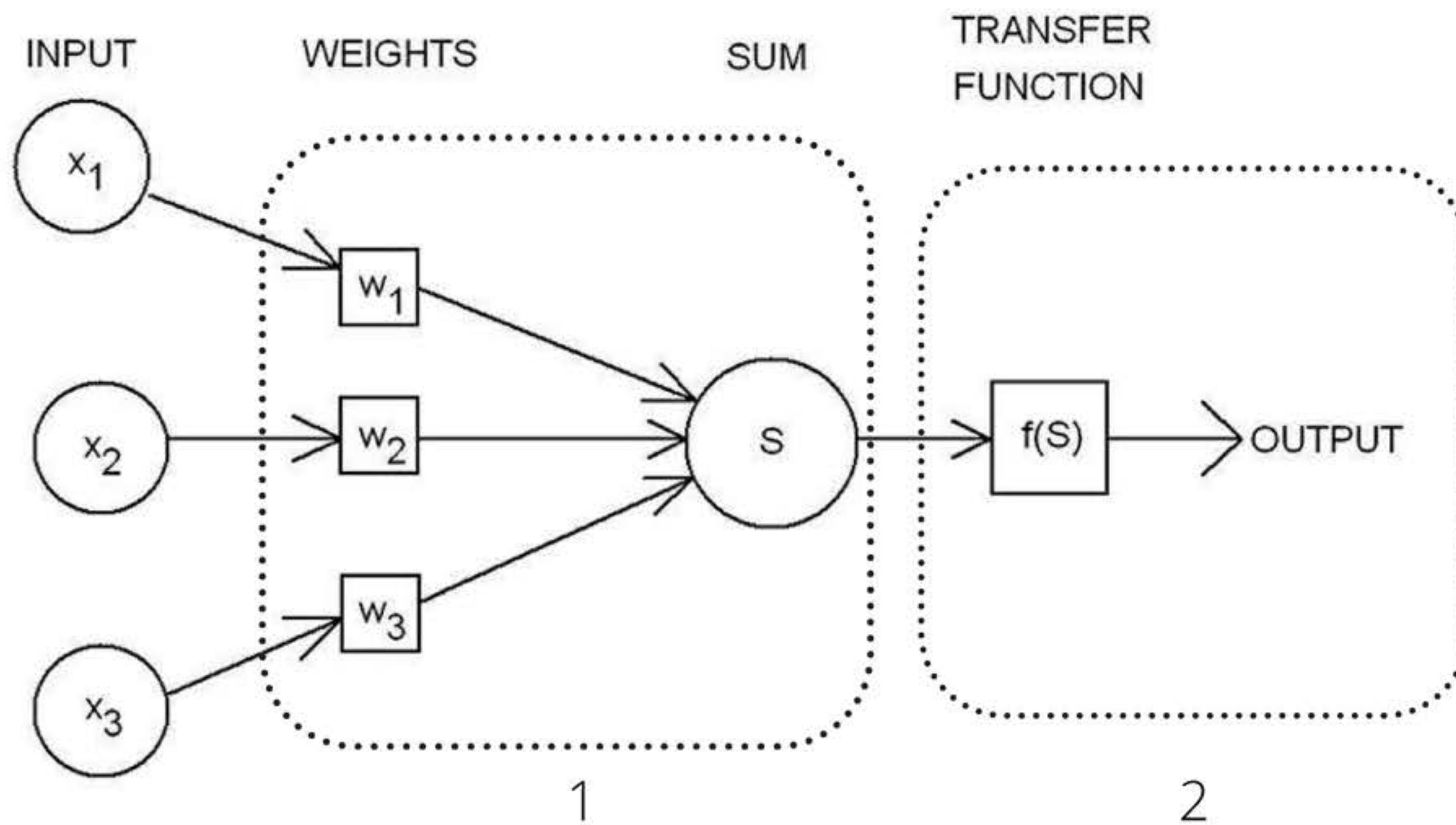
Perceptron

History

The perceptron was first introduced by American psychologist, Frank Rosenblatt in 1957 at Cornell Aeronautical Laboratory. Rosenblatt was heavily inspired by the biological neuron and its ability to learn.

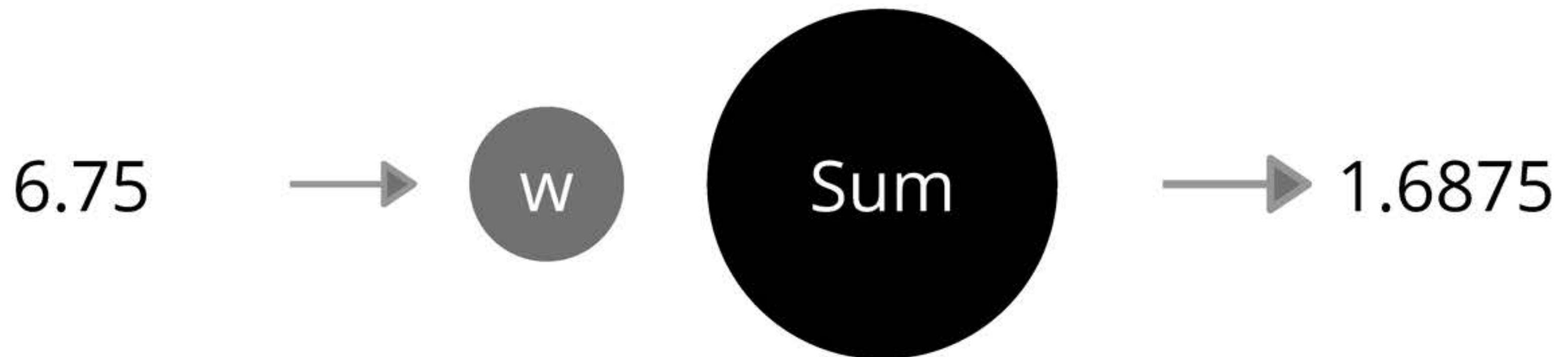


Rosenblatt's perceptron



How does a perceptron work?

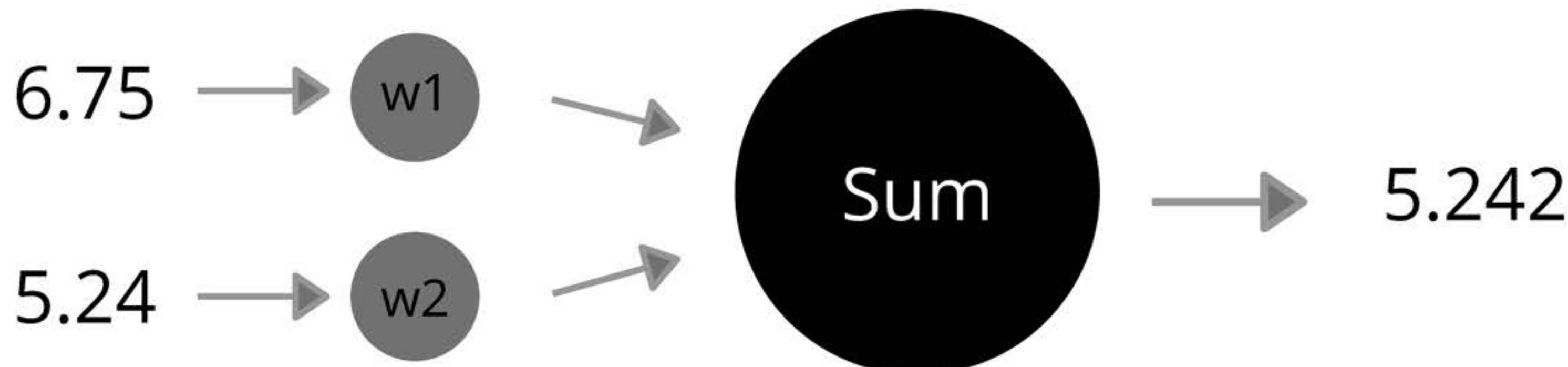
Let's take w as 0.25



Output: $w \cdot x$

Example:

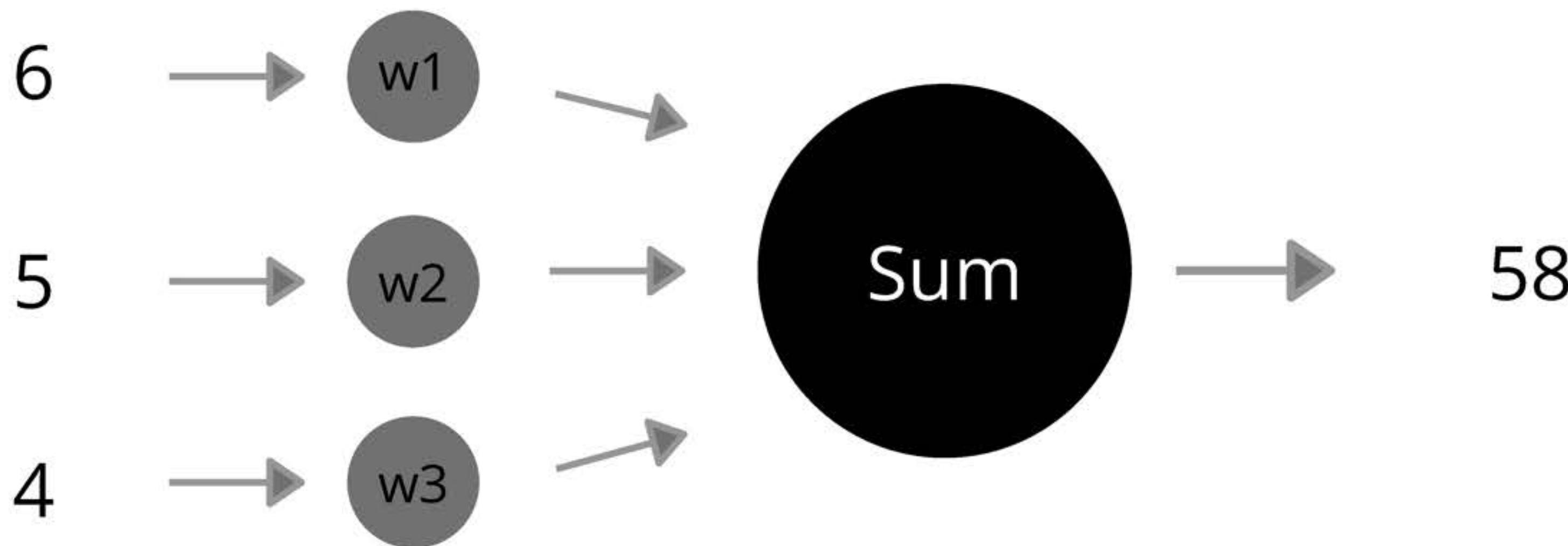
Let's take w_1, w_2 as 0.25, 0.65



Output: $(w_1 * x_1) + (w_2 * x_2)$

Example:

Let's take w_1, w_2, w_3 as 2, 6, 4



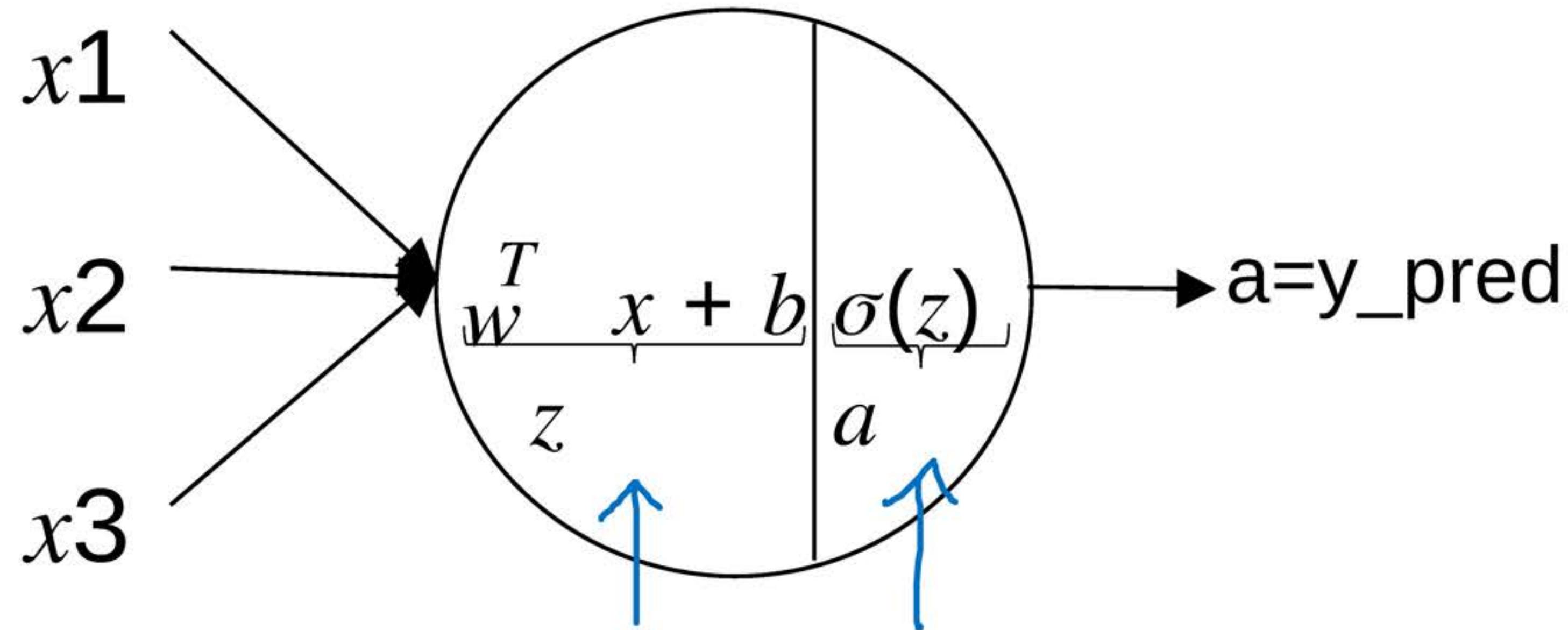
$$\text{Output: } (w_1 \cdot x_1) + (w_2 \cdot x_2) + (w_3 \cdot x_3) == (w^T \cdot x)$$

Activation function



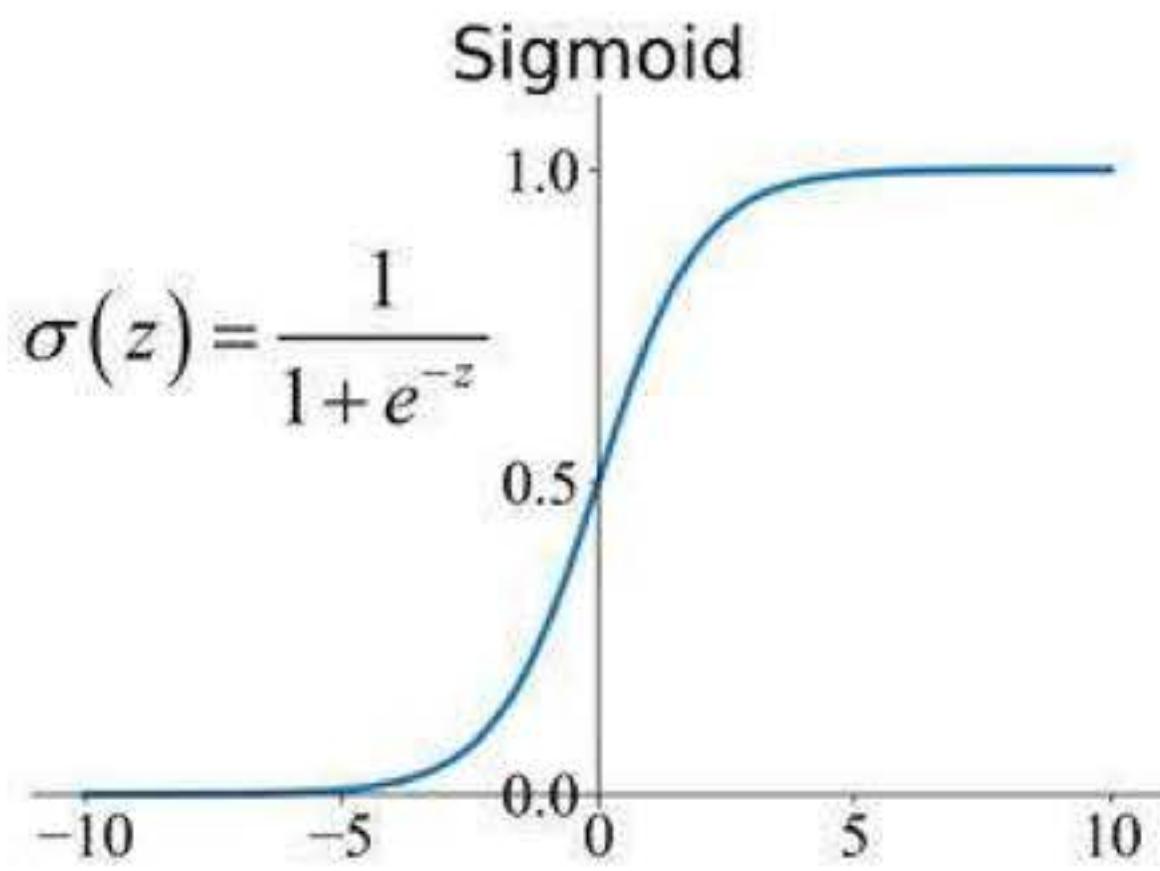
An activation function is a function used in artificial neural networks which outputs a small value for small inputs, and a larger value if its inputs exceed a threshold.

Activation functions are useful because they ~~add non-linearities into neural networks, allowing the neural networks to learn powerful operations.~~

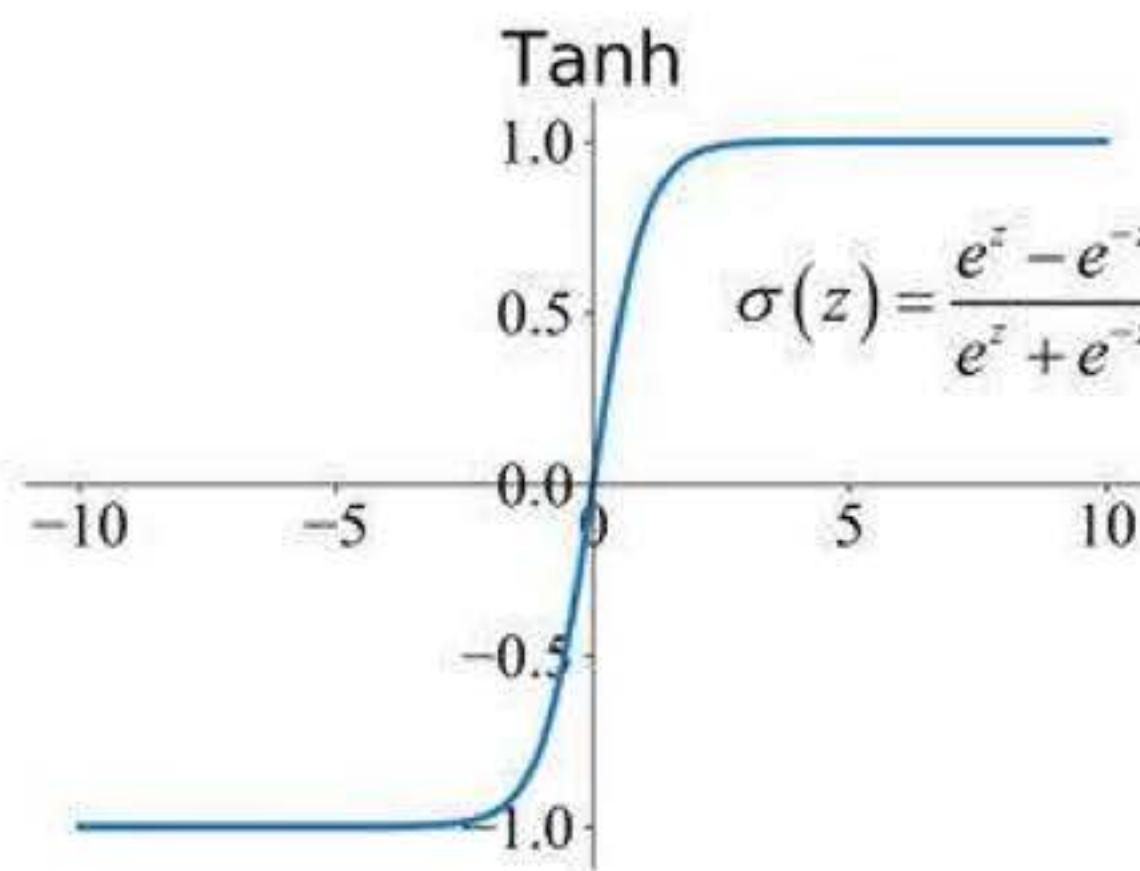


$$z = w^T \cdot x + b \quad a$$

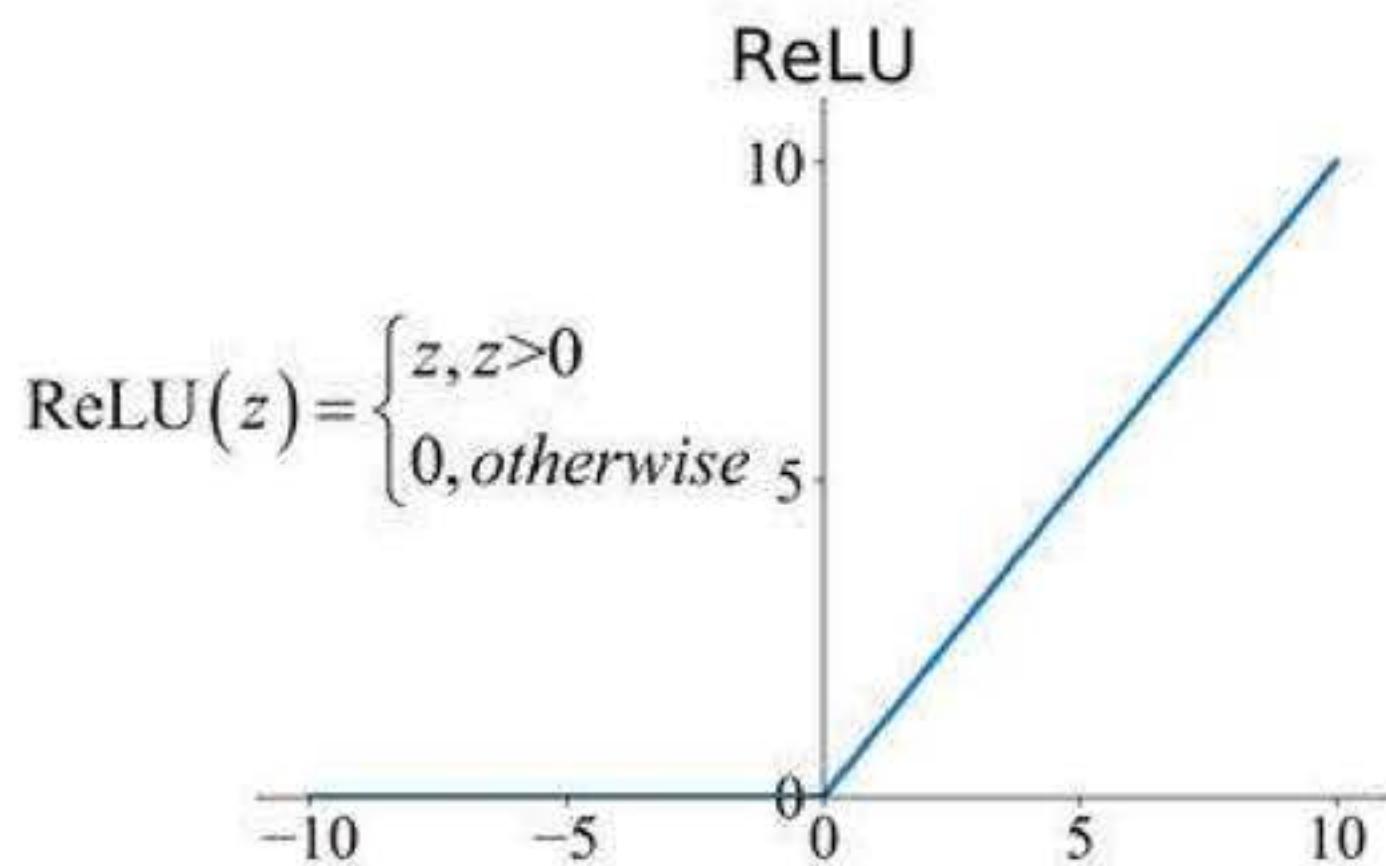
$$= \sigma(z)$$



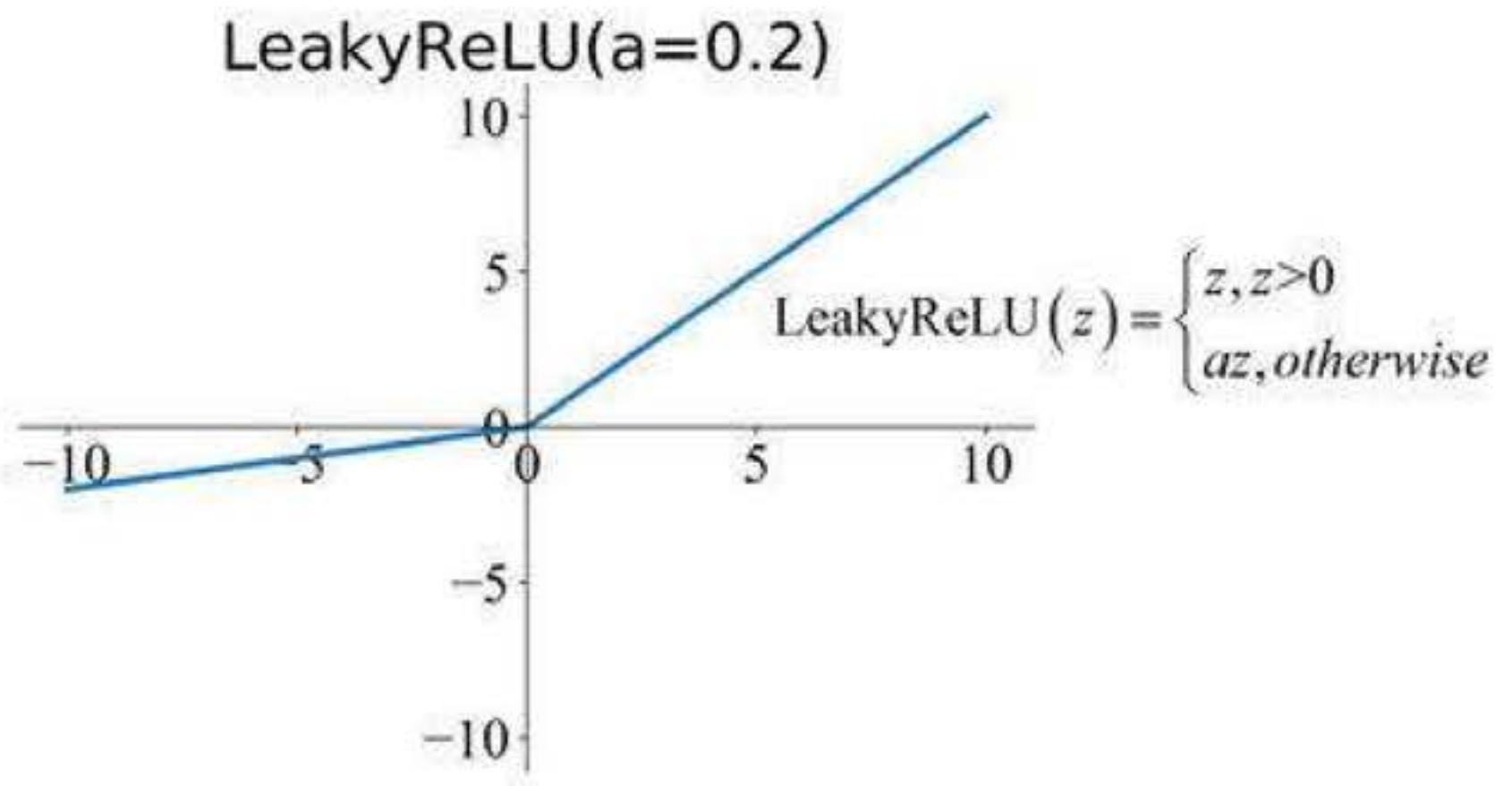
(a)



(b)

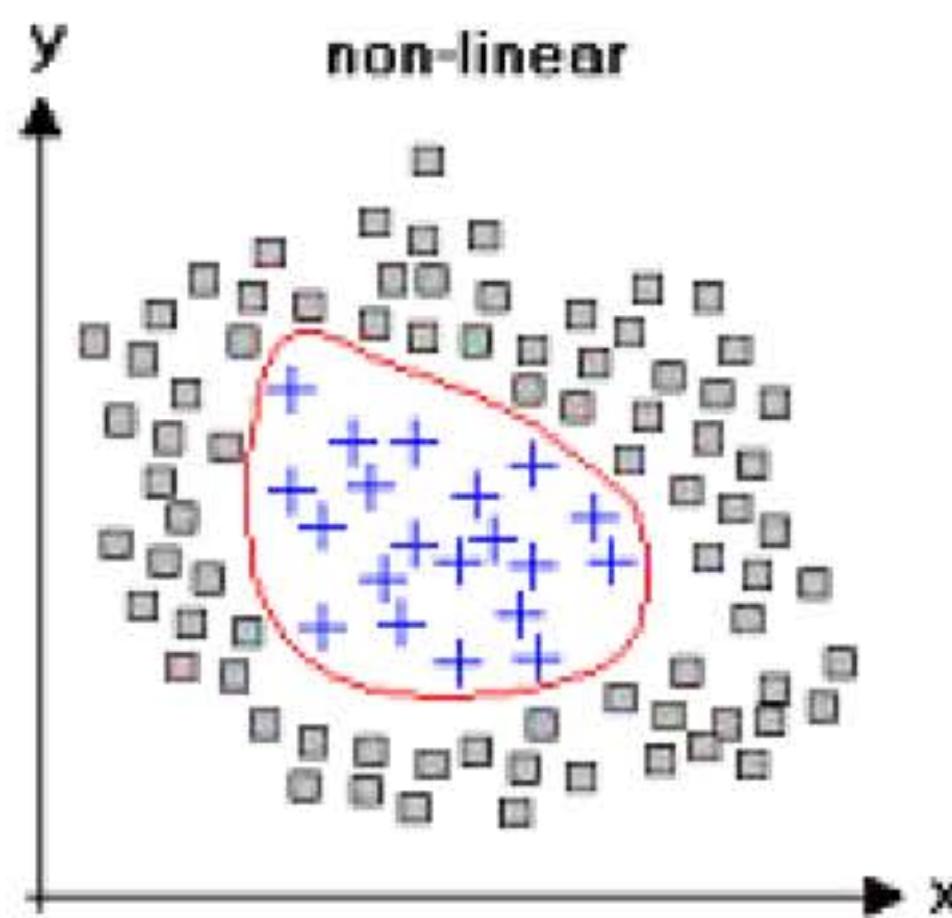
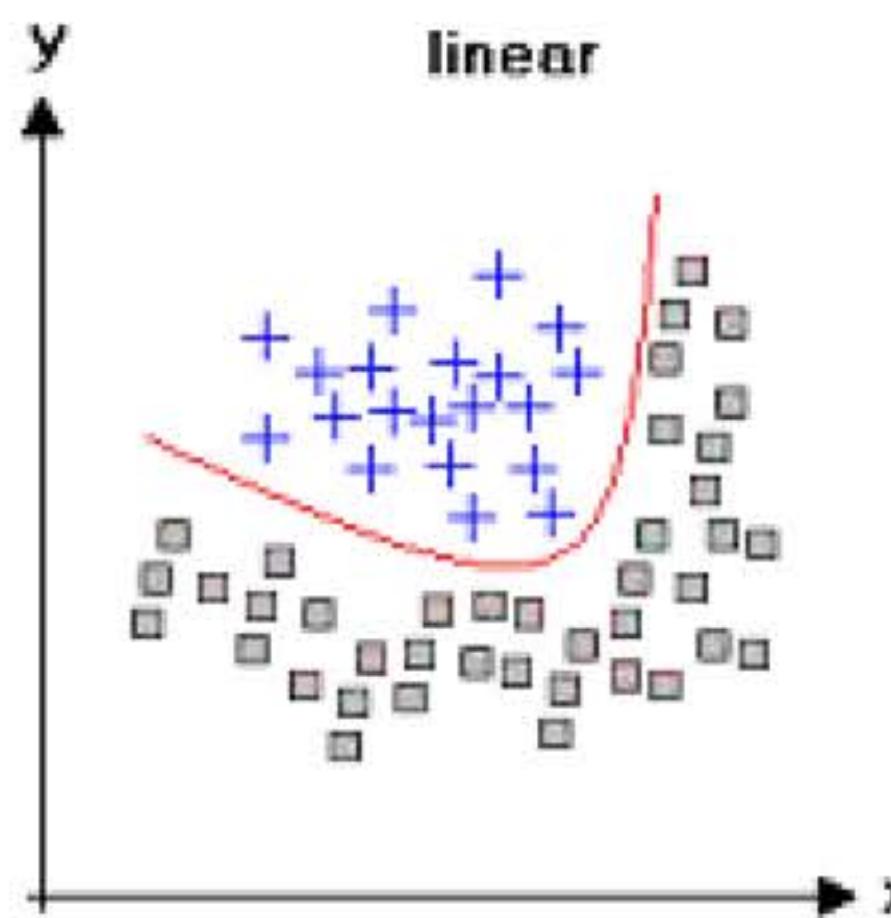
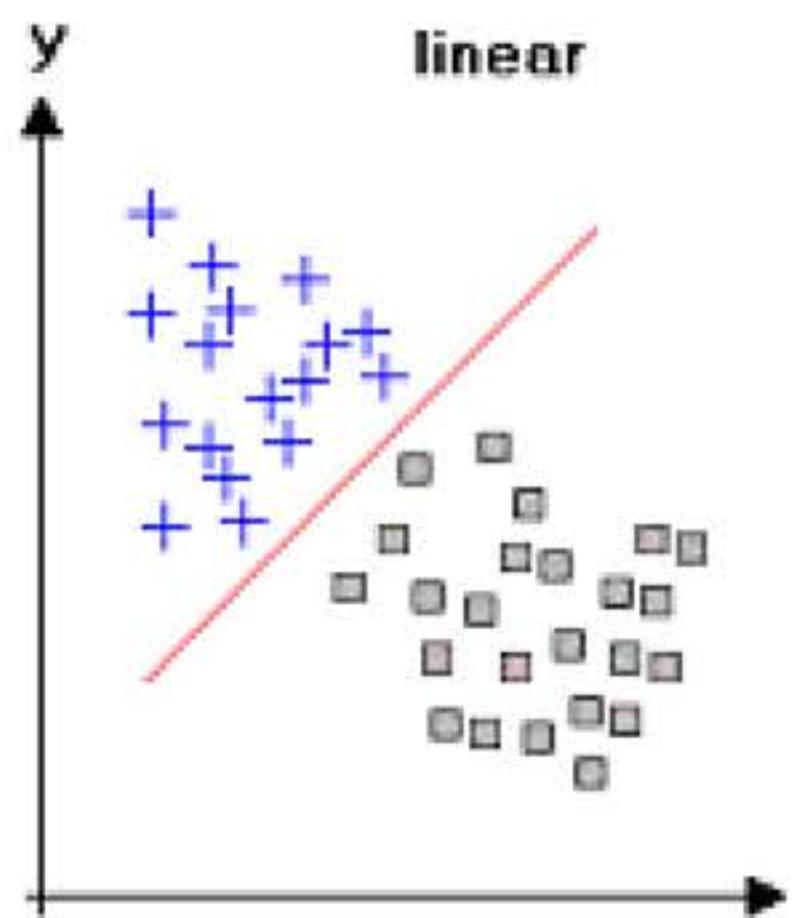


(c)

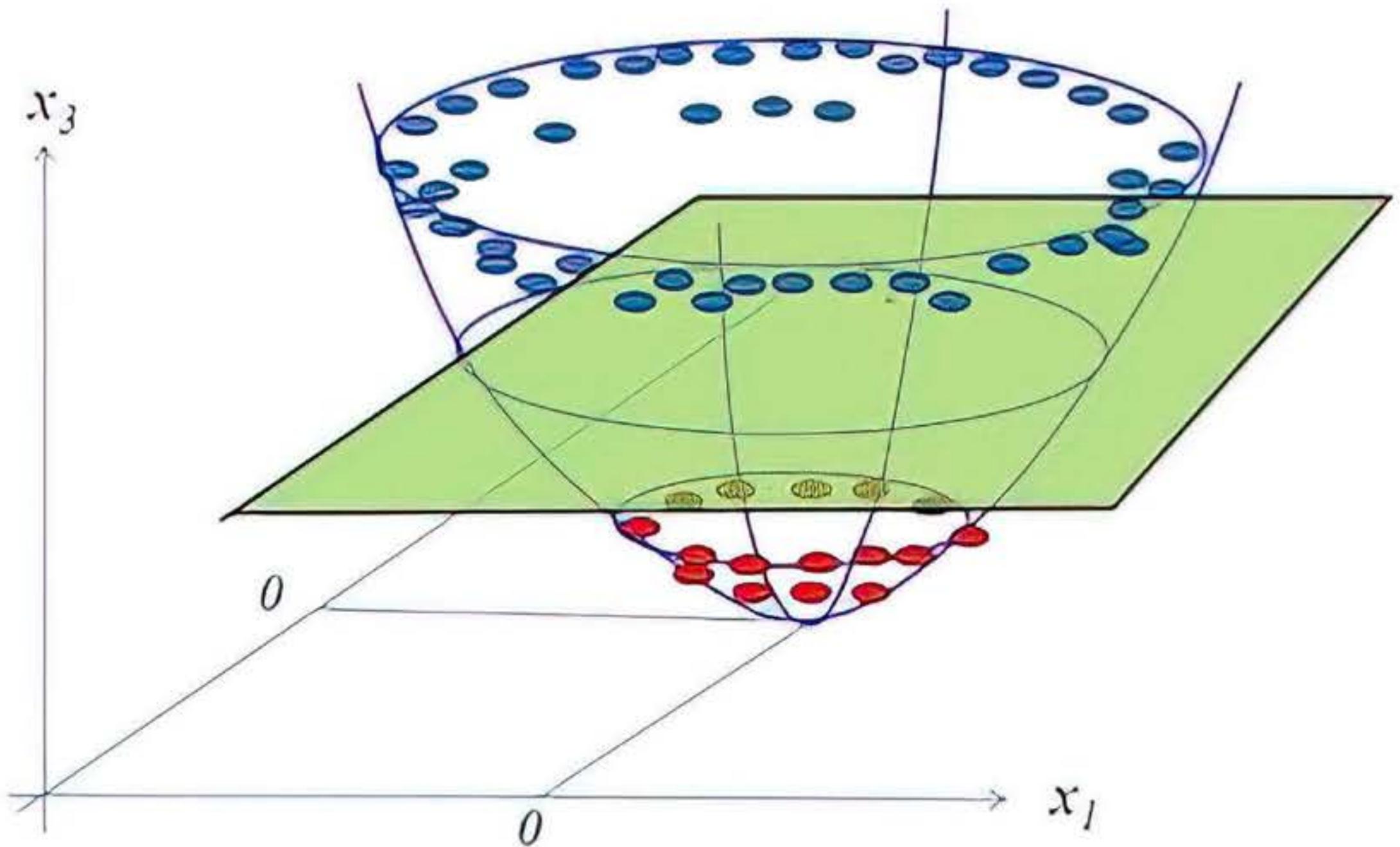
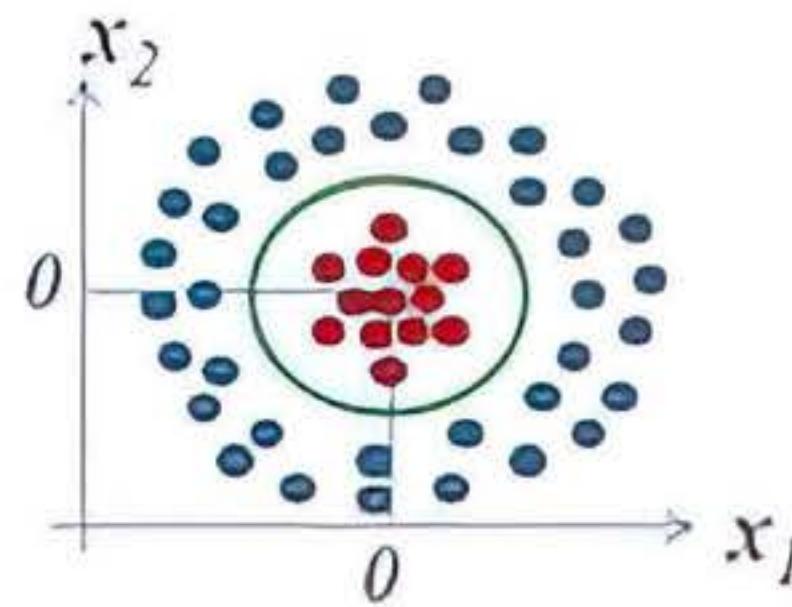


(d)

Non-linearity in neural networks simply mean that the output at any unit cannot be reproduced from a linear function of the input.

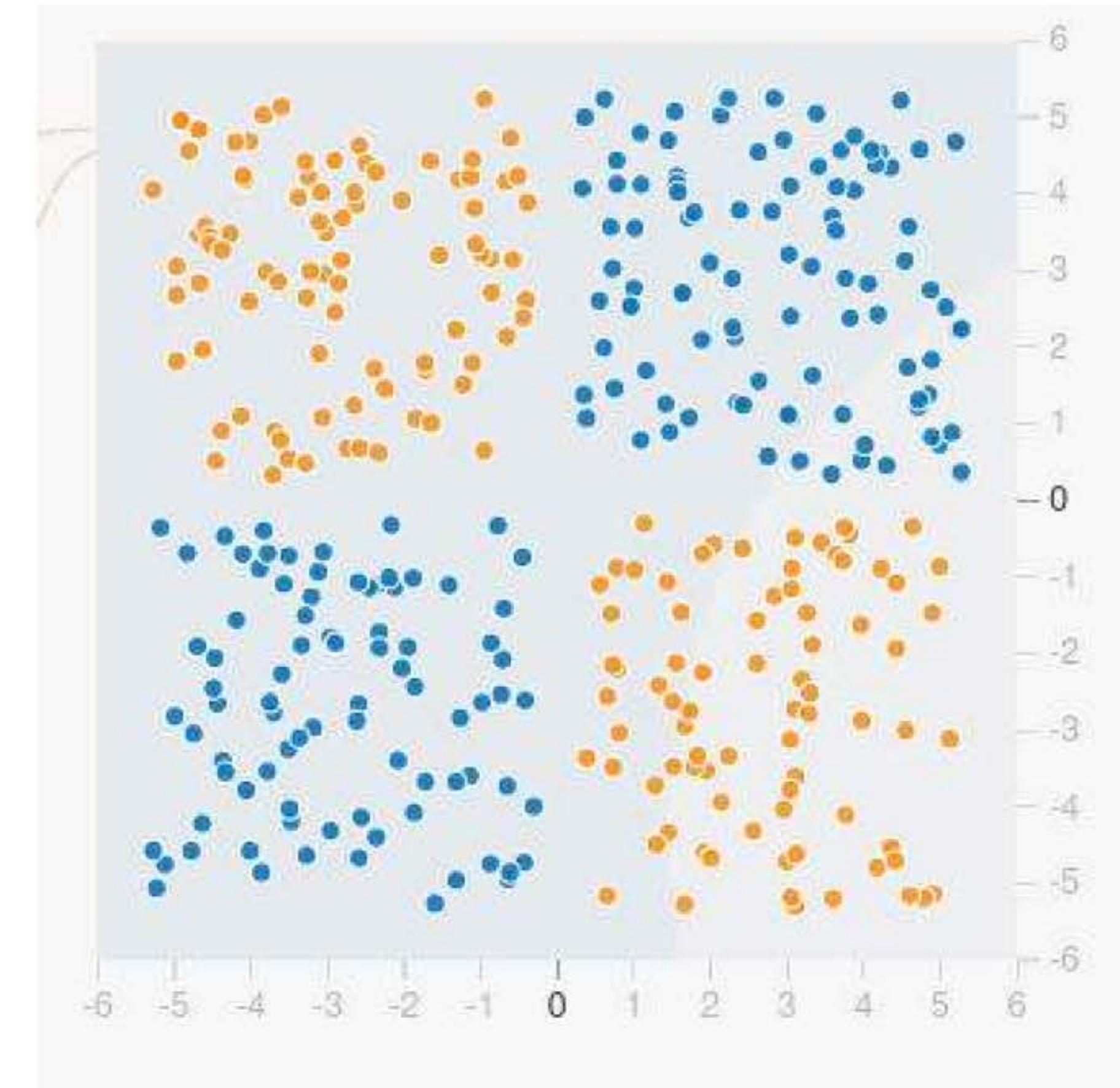


$$x_3 = x_1^2 + x_2^2$$



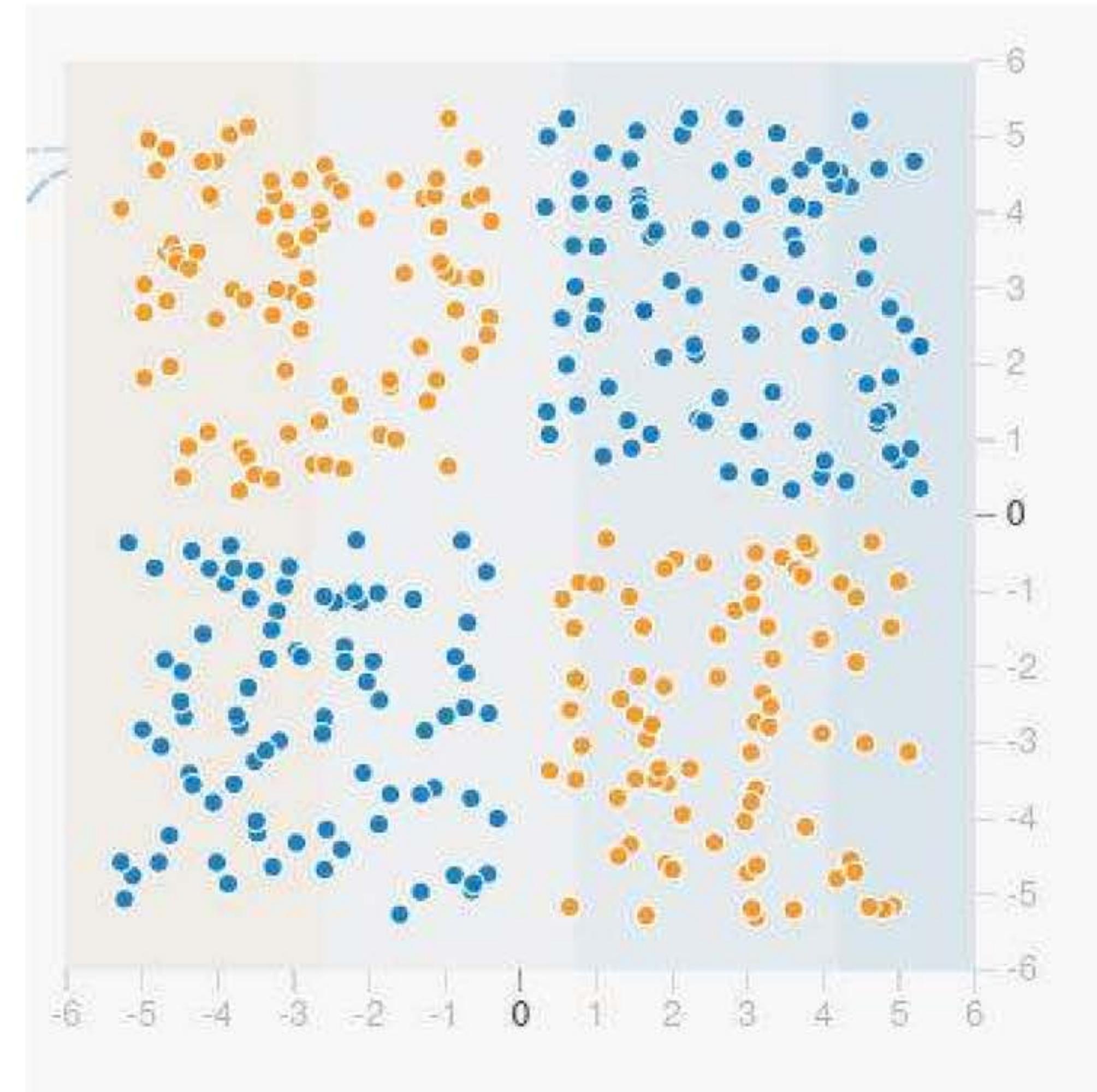
Class 1

Class 2

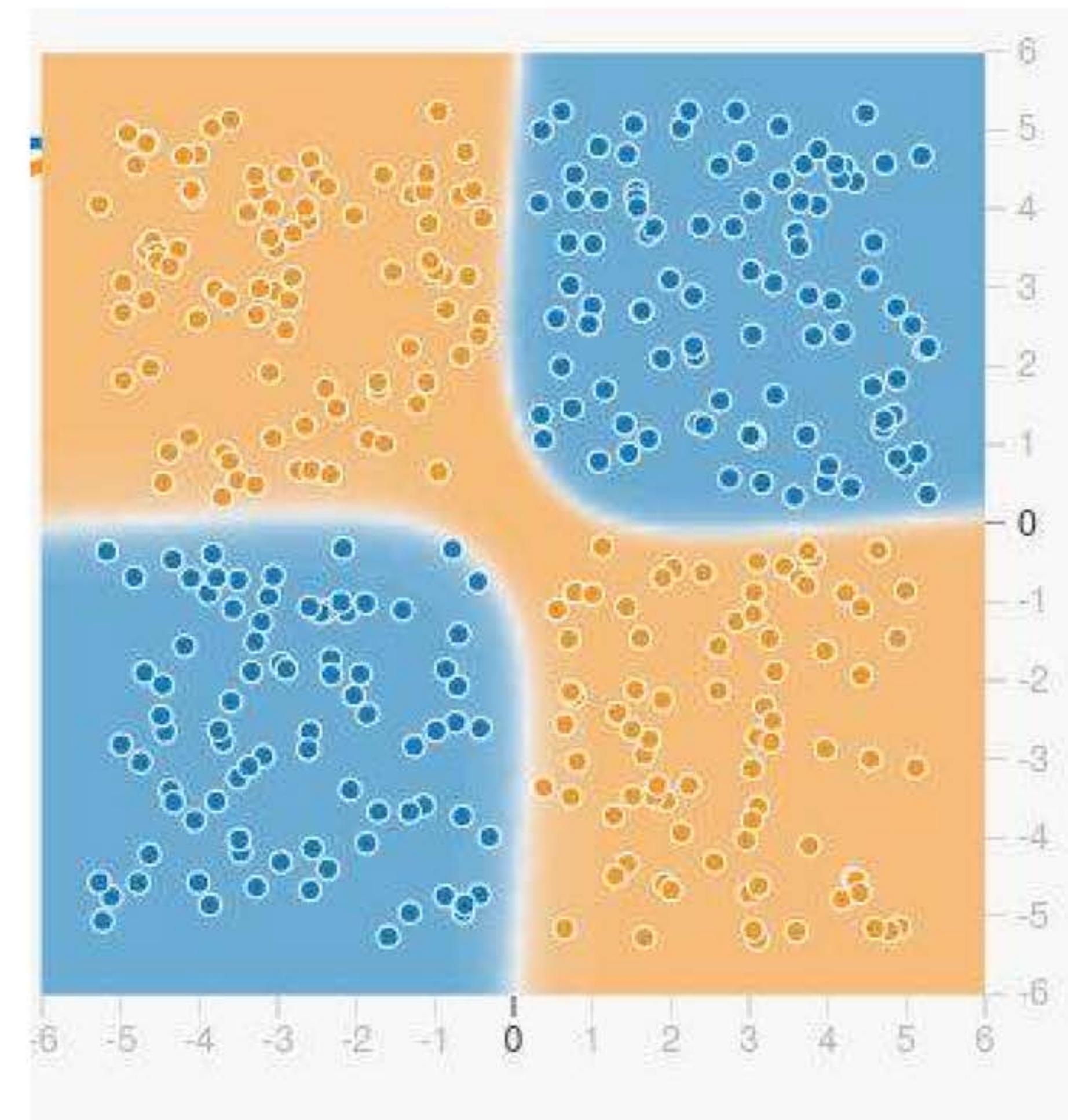




linear
activation
function, after
1000 epochs

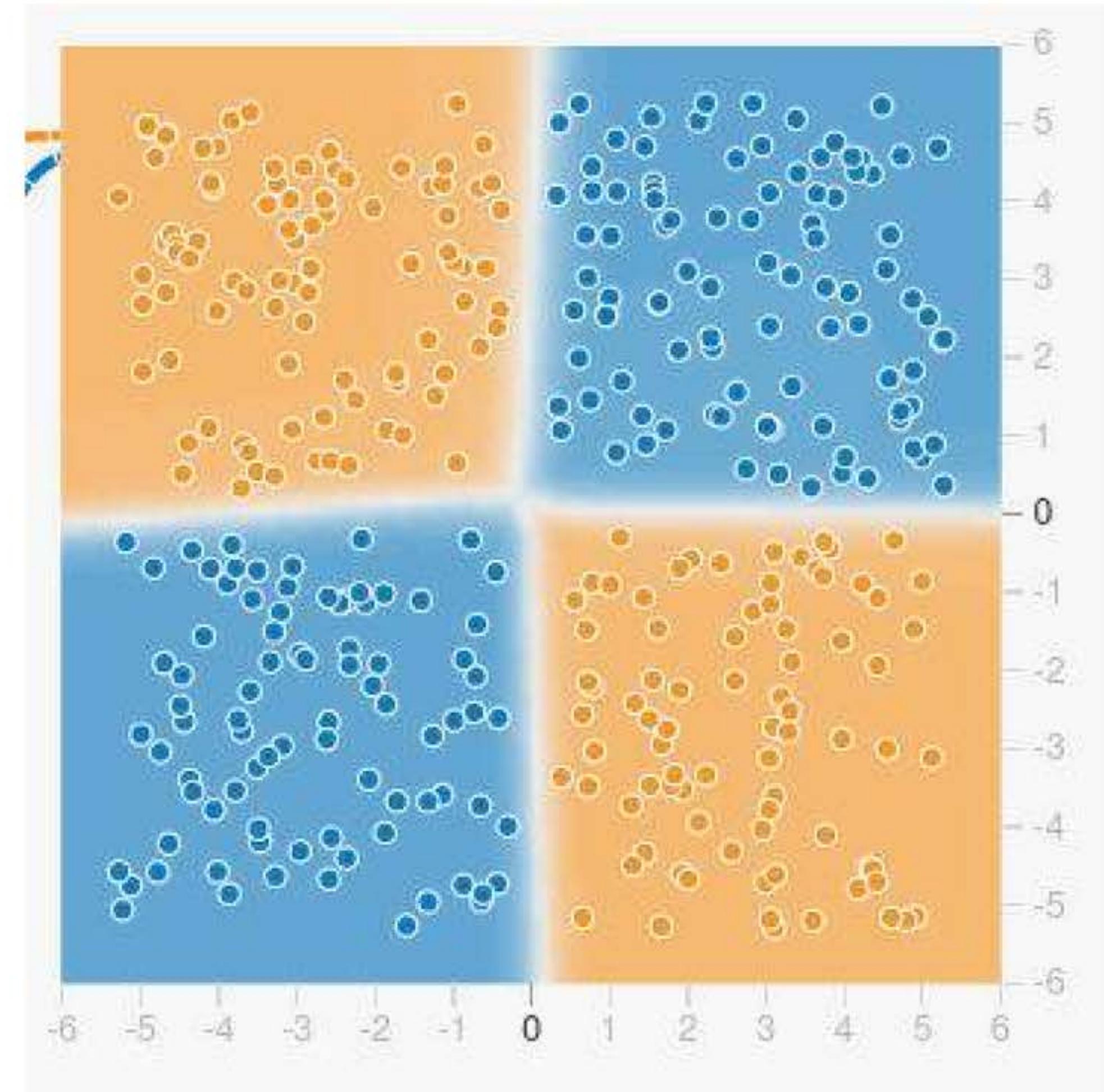


Tanh
activation
function, after 85
epochs



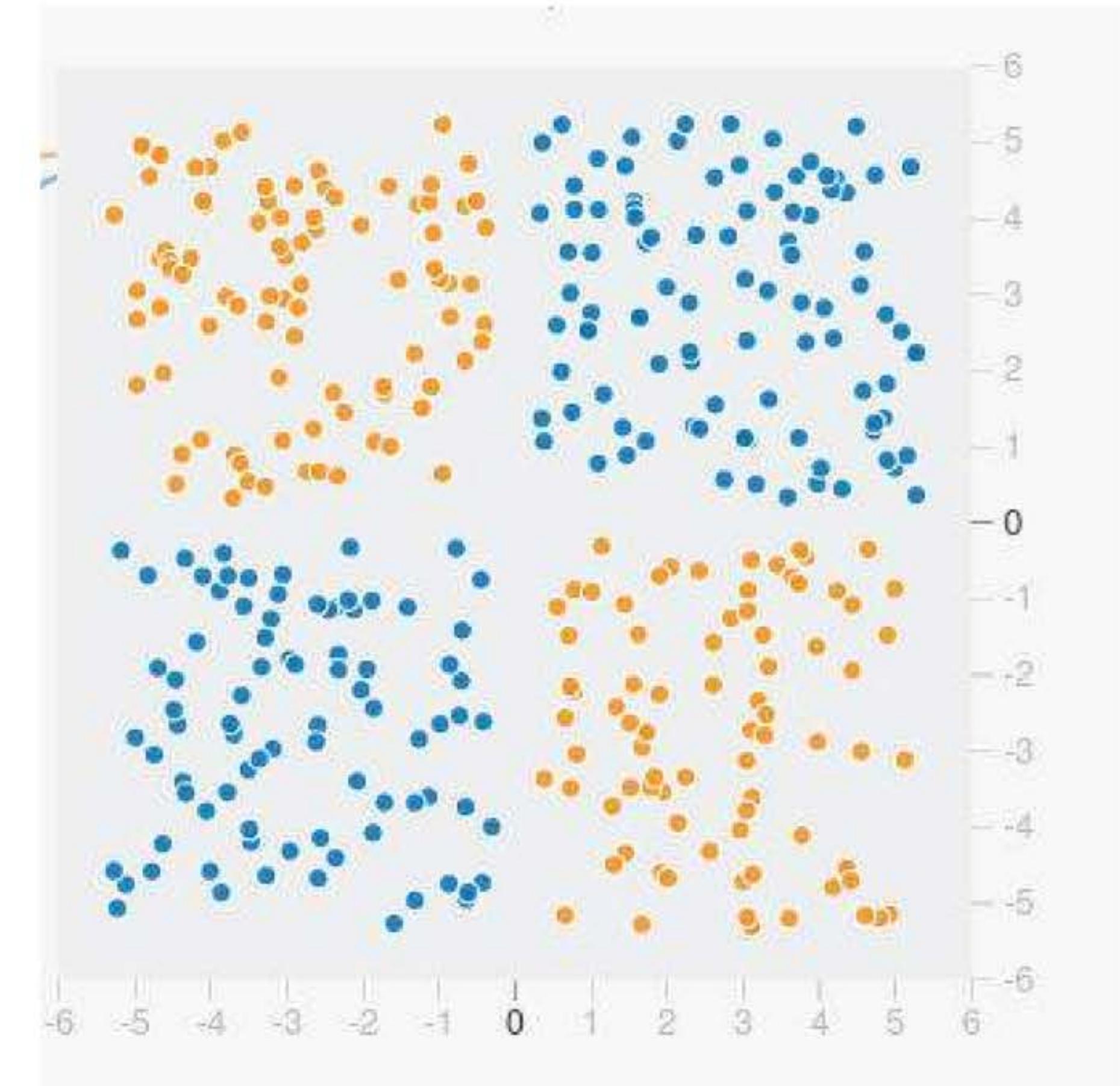


**Relu
activation
function, after
23 epochs**

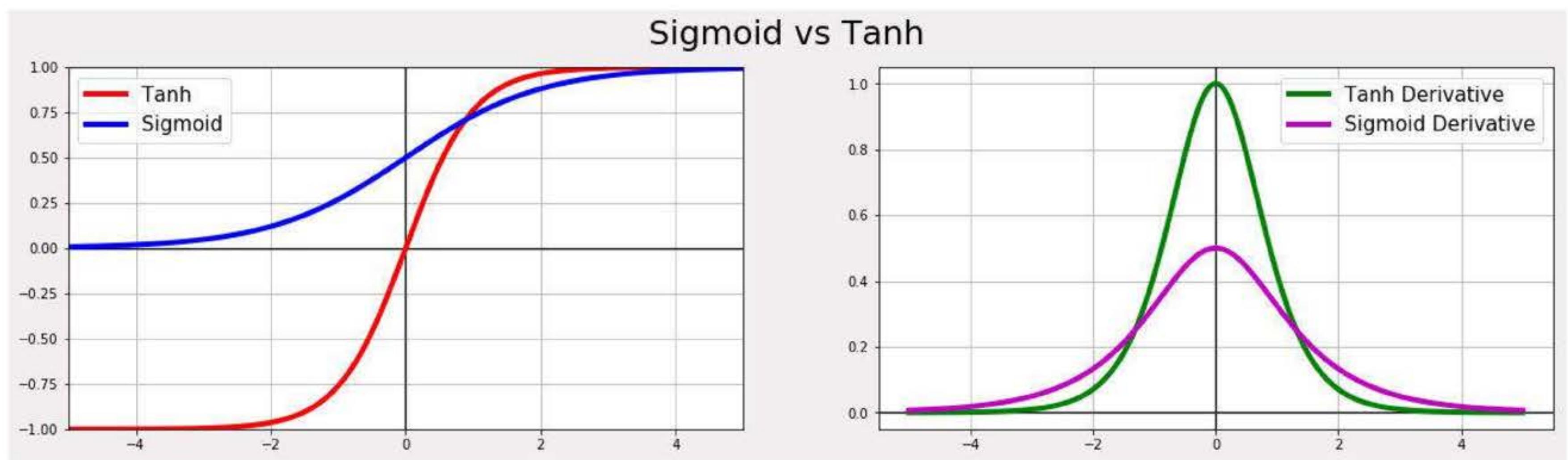


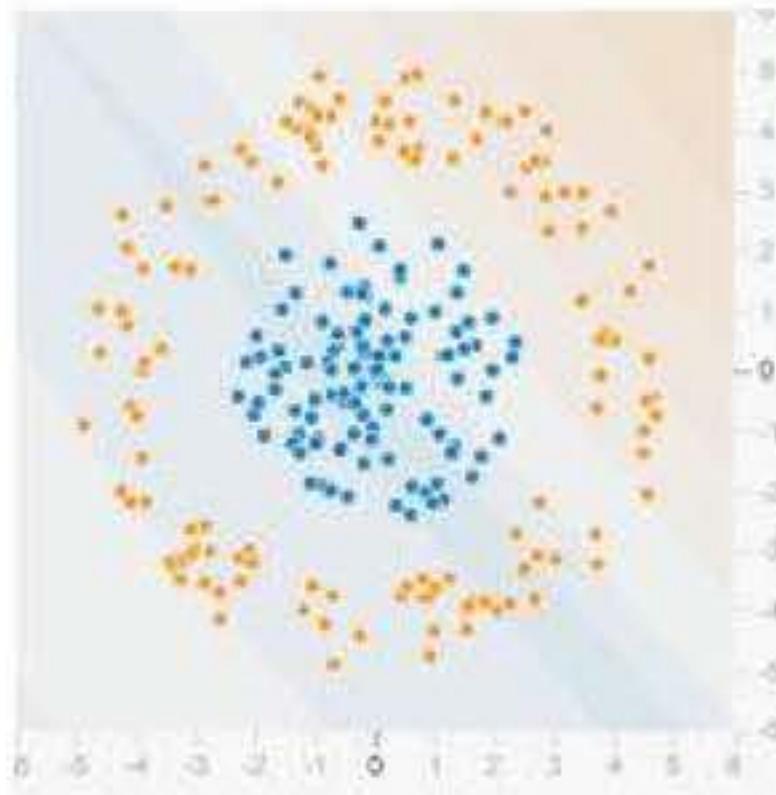
Sigmoid activation
function, after 500
epochs

sigmoid is a
nonlinear activation
function but doesn't
perform well , why ?

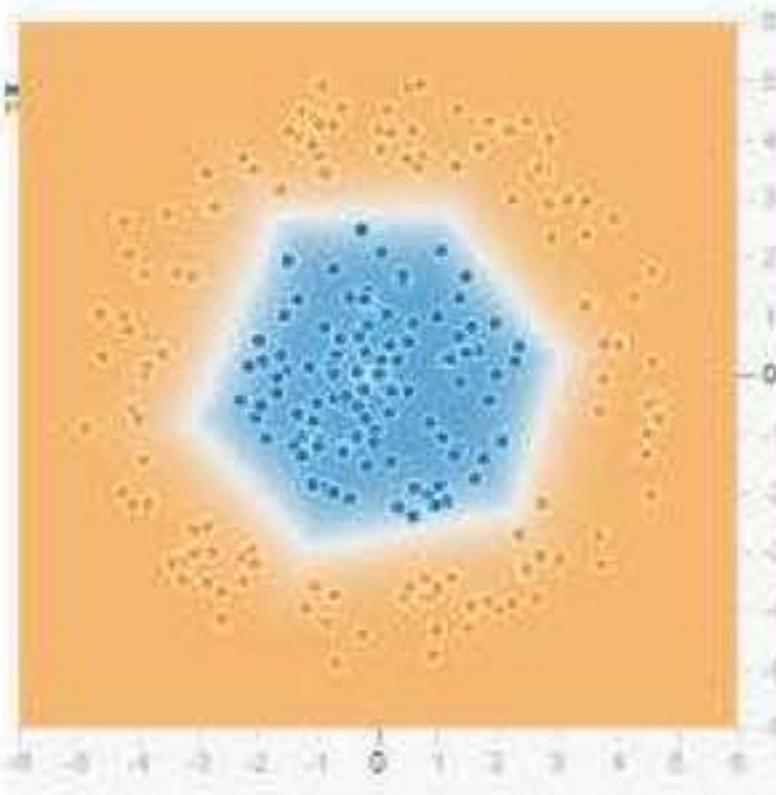


- Tanh is almost always preferable to using sigmoid
- reason is that the outputs using tanh centre around 0 rather than sigmoid's 0.5, and this makes learning for the next layer a little bit easier".

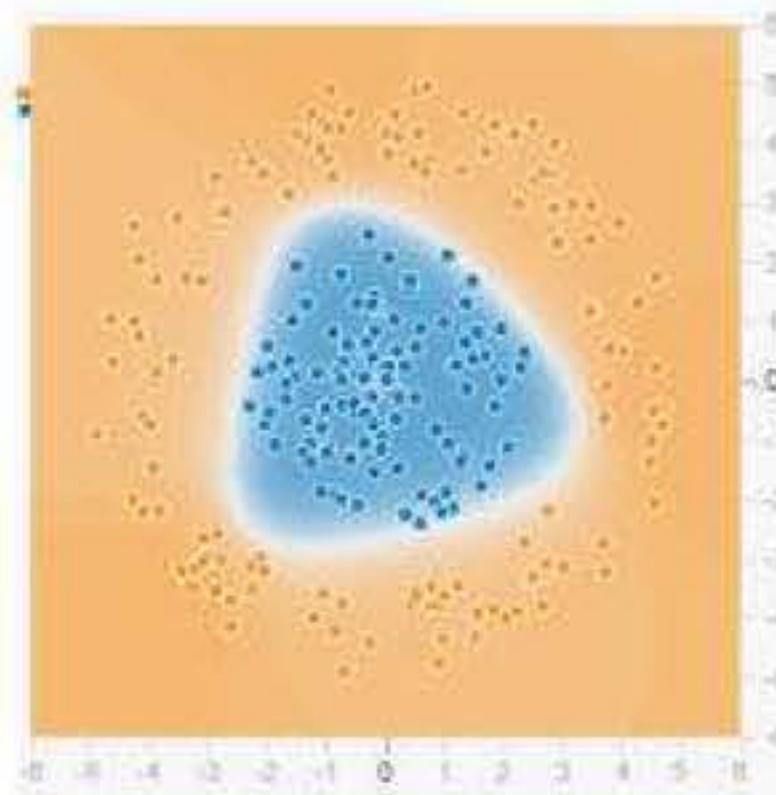




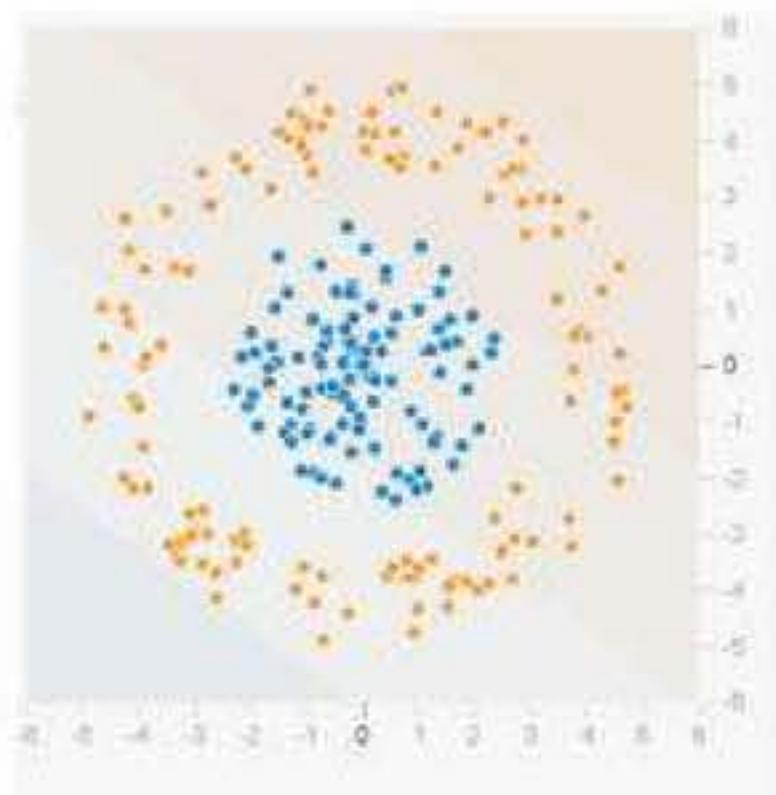
| Data for classification



Decision boundary
using RELU in 75 epochs



Decision boundary
using tanh in 90 epochs



Decision boundary
using linear activation
in 2000 epochs

Sigmoid



$$y = \frac{1}{1+e^{-x}}$$

Tanh



Step Function



$$y = \begin{cases} 0, & x < n \\ 1, & x \geq n \end{cases}$$

Softplus



$$y = \ln(1+e^x)$$

ReLU



$$y = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}$$

Softsign



$$y = \frac{x}{(1+|x|)}$$

ELU



$$y = \begin{cases} \alpha(e^x - 1), & x < 0 \\ x, & x \geq 0 \end{cases}$$

Log of Sigmoid



$$y = \ln\left(\frac{1}{1+e^{-x}}\right)$$

Swish



$$y = \frac{x}{1+e^{-x}}$$

Sinc



$$y = \frac{\sin(x)}{x}$$

Leaky ReLU



$$y = \max(0.1x, x)$$

Mish



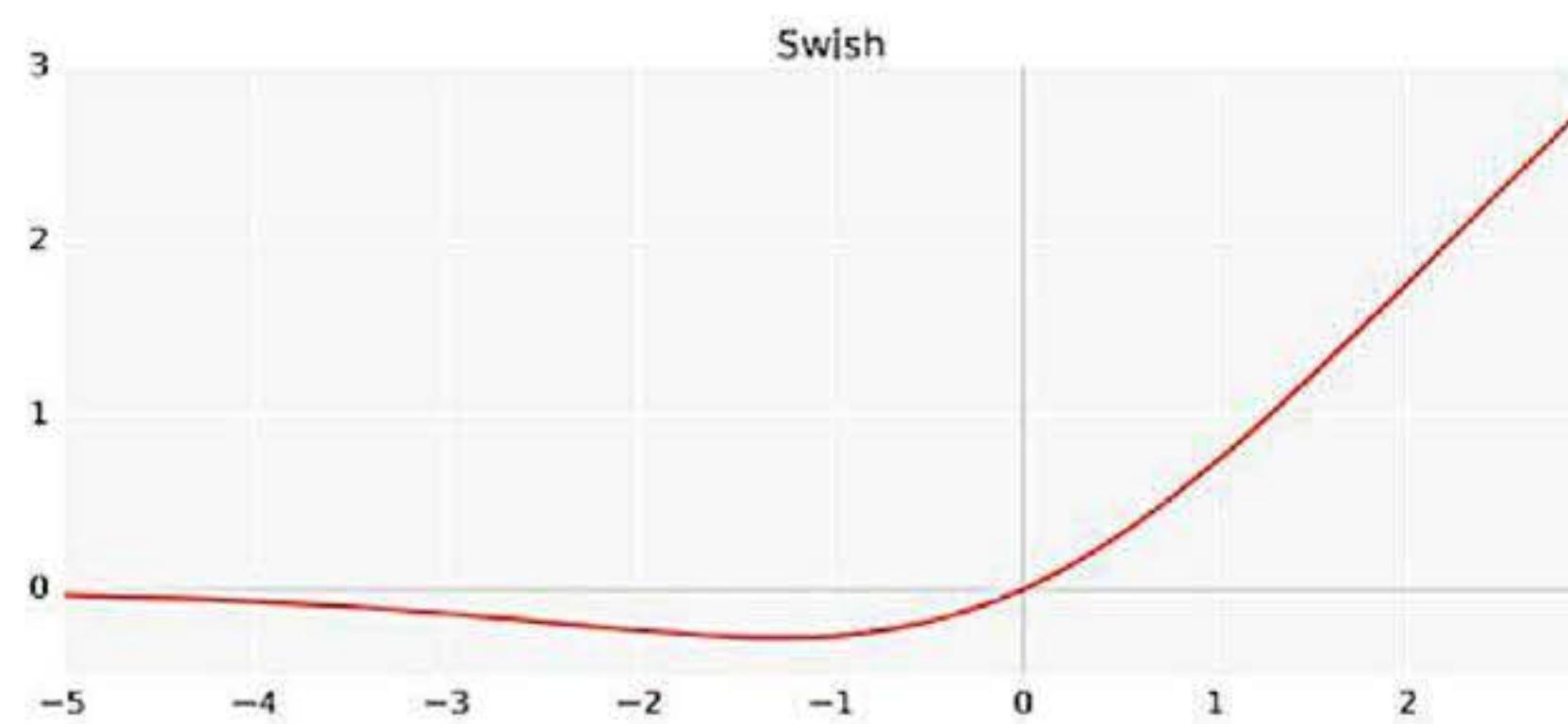
$$y = x(\tanh(\text{softplus}(x)))$$



Choosing the right Activation Function

- Sigmoid functions and their combinations generally work better in the case of classifiers.
- Sigmoids and tanh functions are sometimes avoided due to the vanishing gradient problem.
- ReLU function is a general activation function and is used in most cases these days.

Swish vs. ReLU. The authors find that by substituting the ReLU units for Swish units, there is significant improvement over ReLU as the number of layers increases from 42 (when optimization becomes more difficult). The authors also found that Swish outperforms ReLU with diverse sizes of batches.



**Swish Activation
Function**

$$f(x) = x * \text{sigmoid}(x)$$

Applying Activation function

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

Sigmoid Function

$$w_1=0.25, w_2=0.65$$



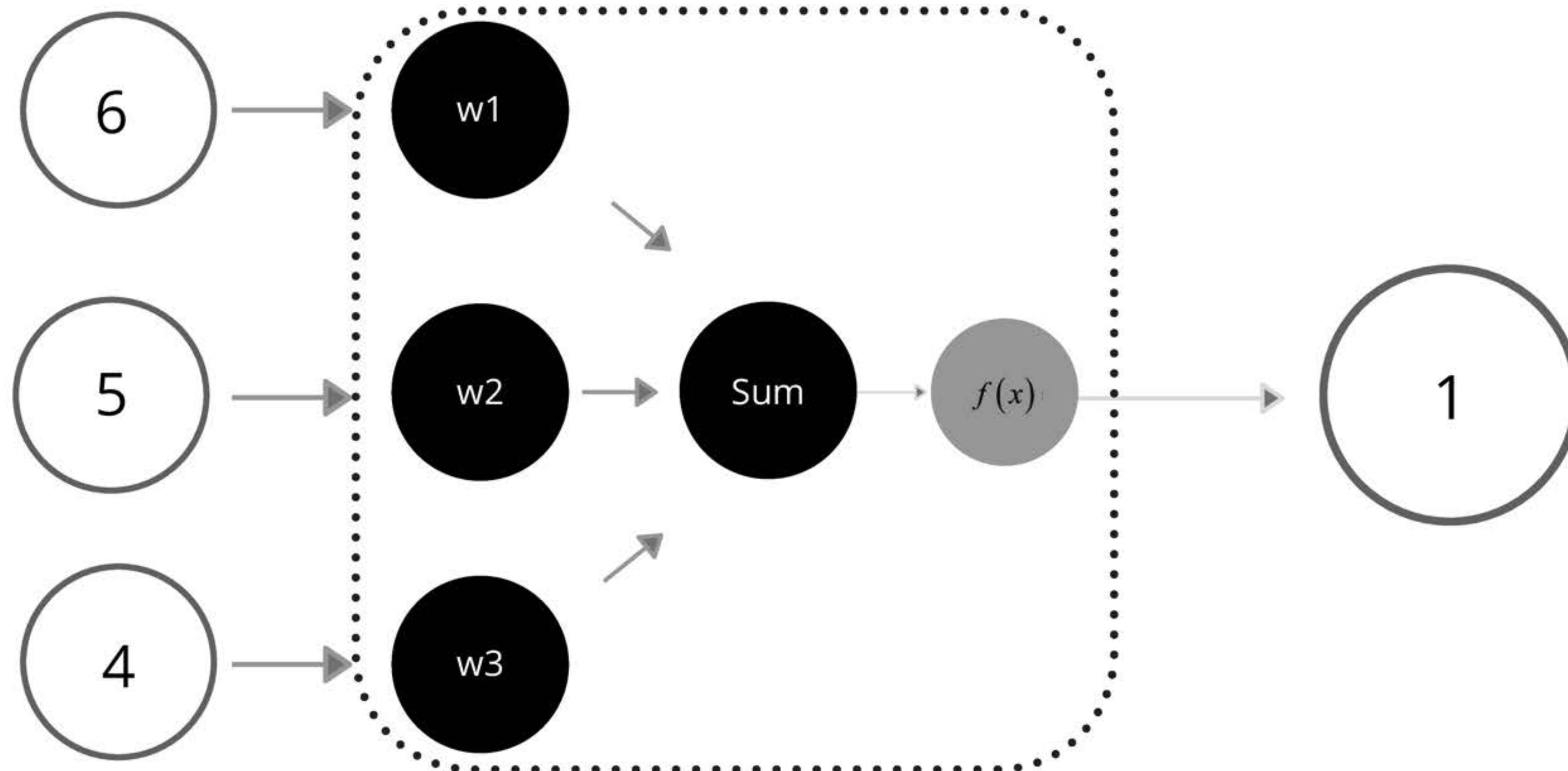
Applying Activation function

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

Sigmoid Function



An Overview



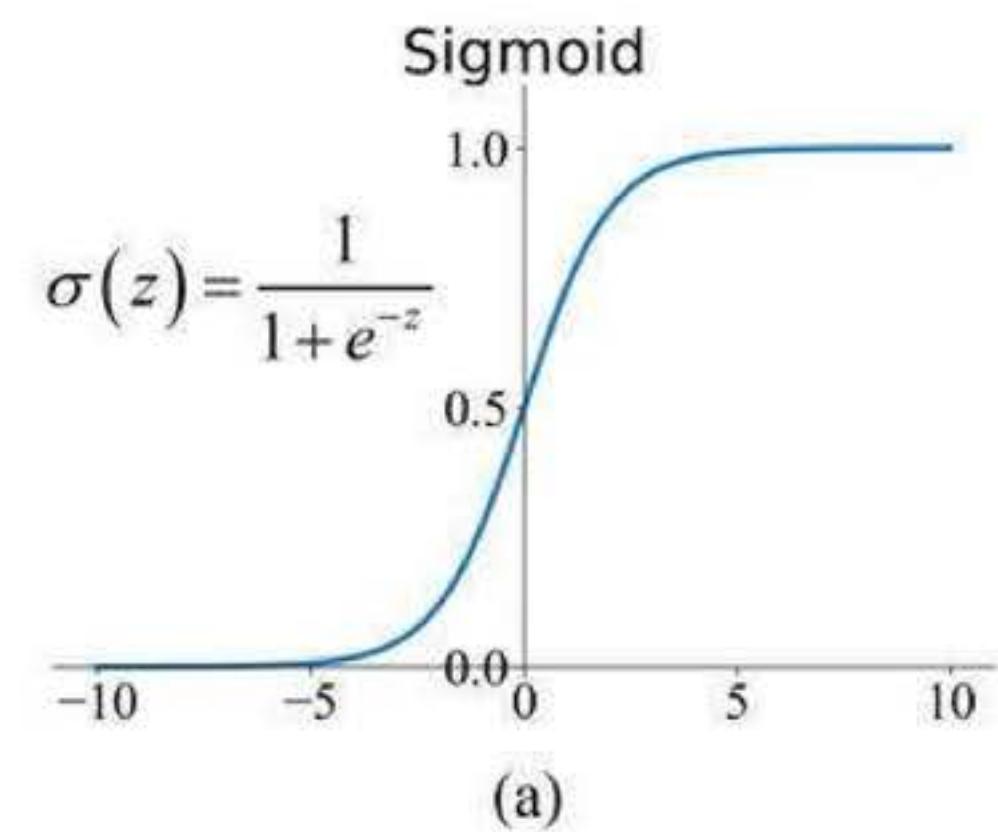
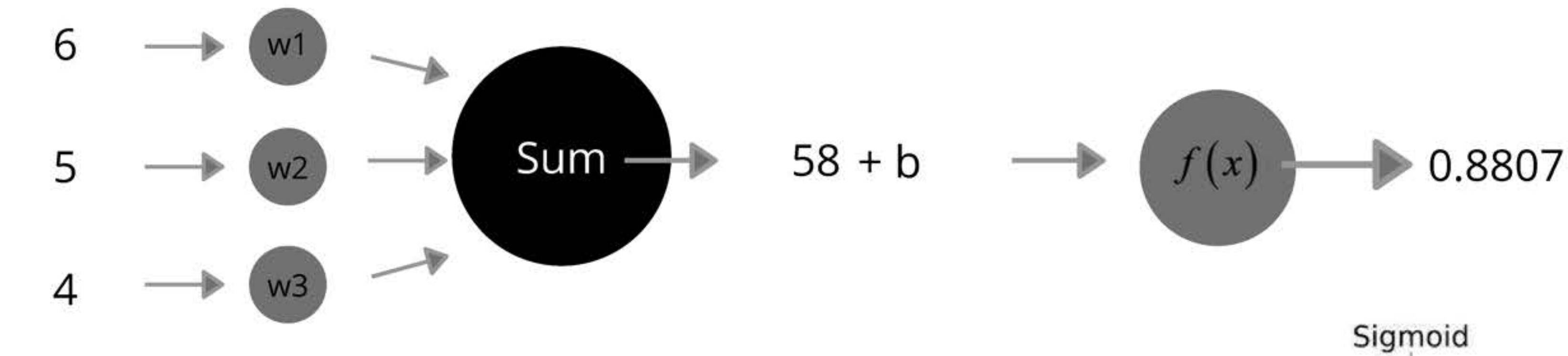
The need for Bias

Bias allows you to shift the activation function by adding a constant

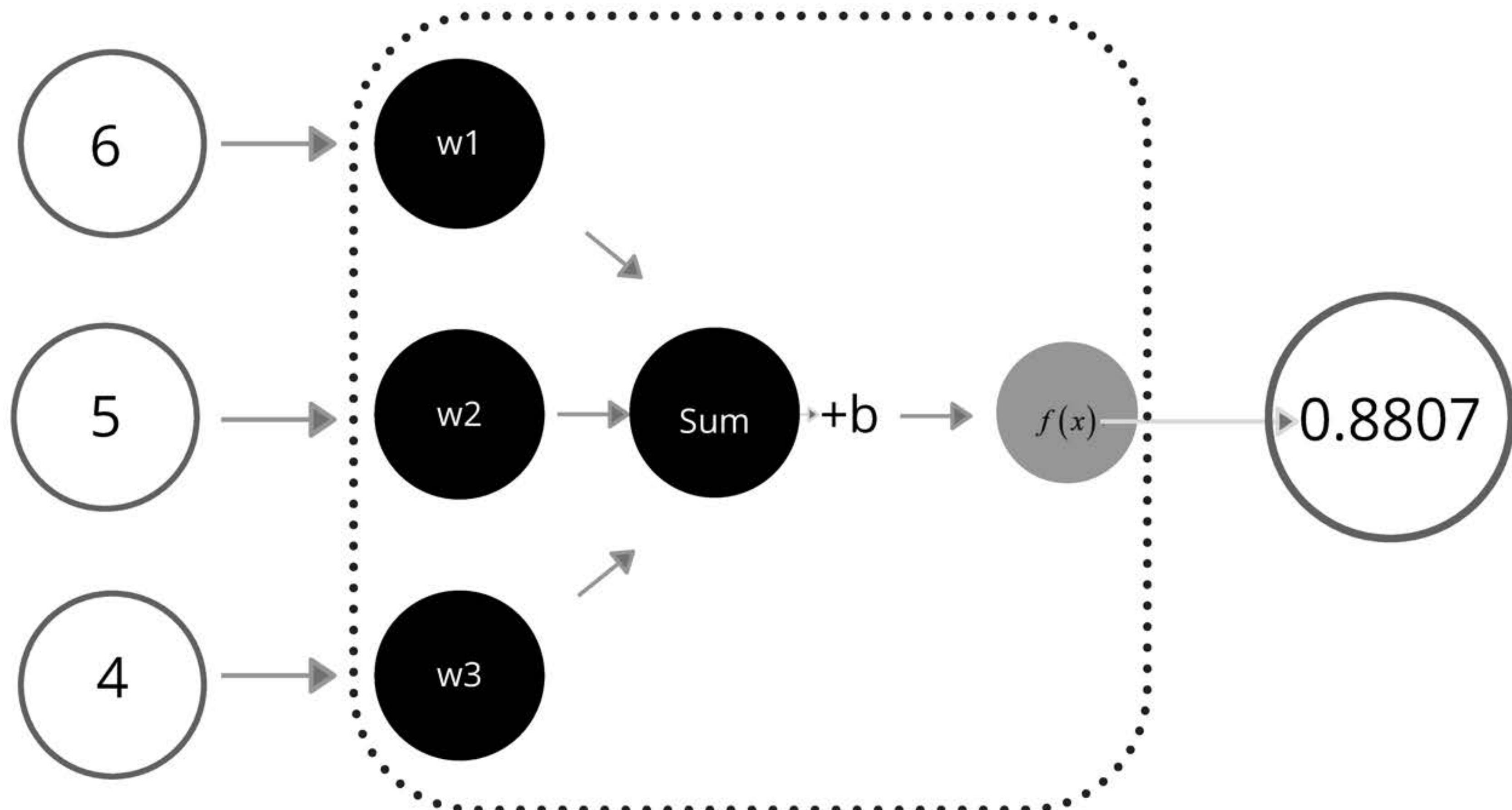
$$y = (m*x) + c$$

$$\text{Output} = (W*x) + b(\text{bias})$$

The need for Bias



An Overview



Thanks
you!

Machine Learning (19CSE305)

Artificial Neural Network



Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

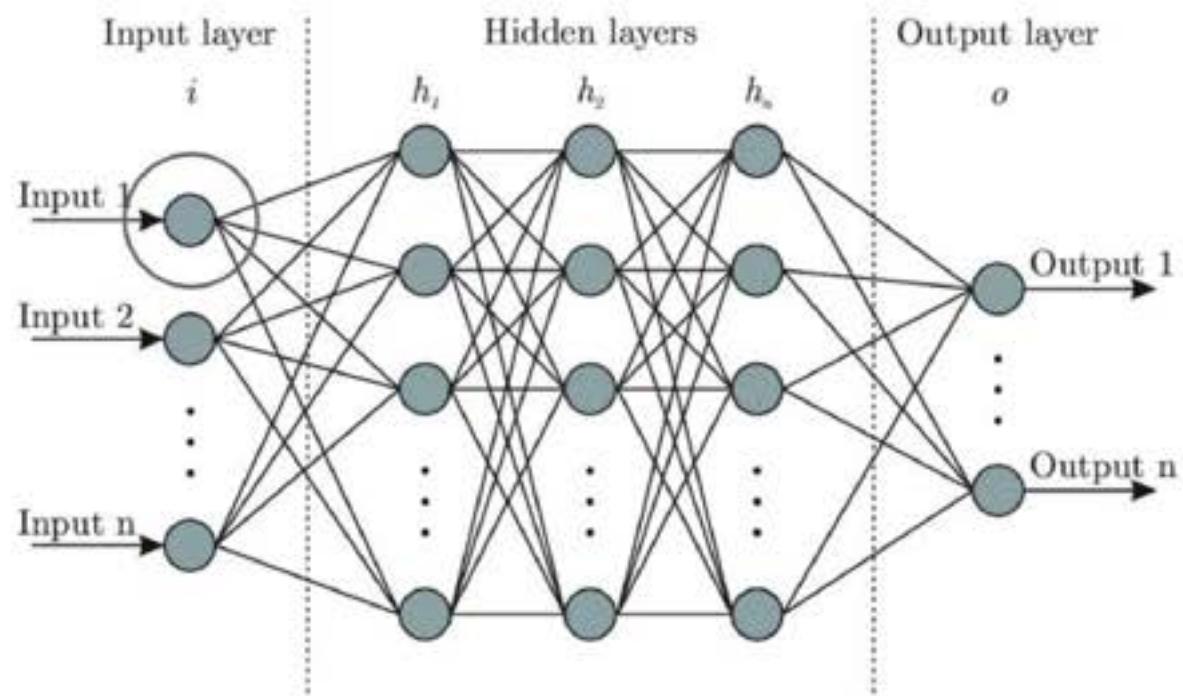
- Superkey
- Keys
- Schema
- Normalization

WHAT?

- A Machine Learning model.
- Inspired by the human brain.
- A web of interconnected entities known as nodes.
- Also called SNN: Simulated Neural Networks.

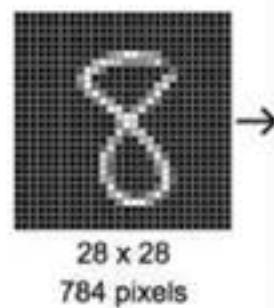
STRUCTURE

- Artificial Neurons
- Layers:
 - a. Input Layer
 - b. Hidden Layers
 - c. Output Layer

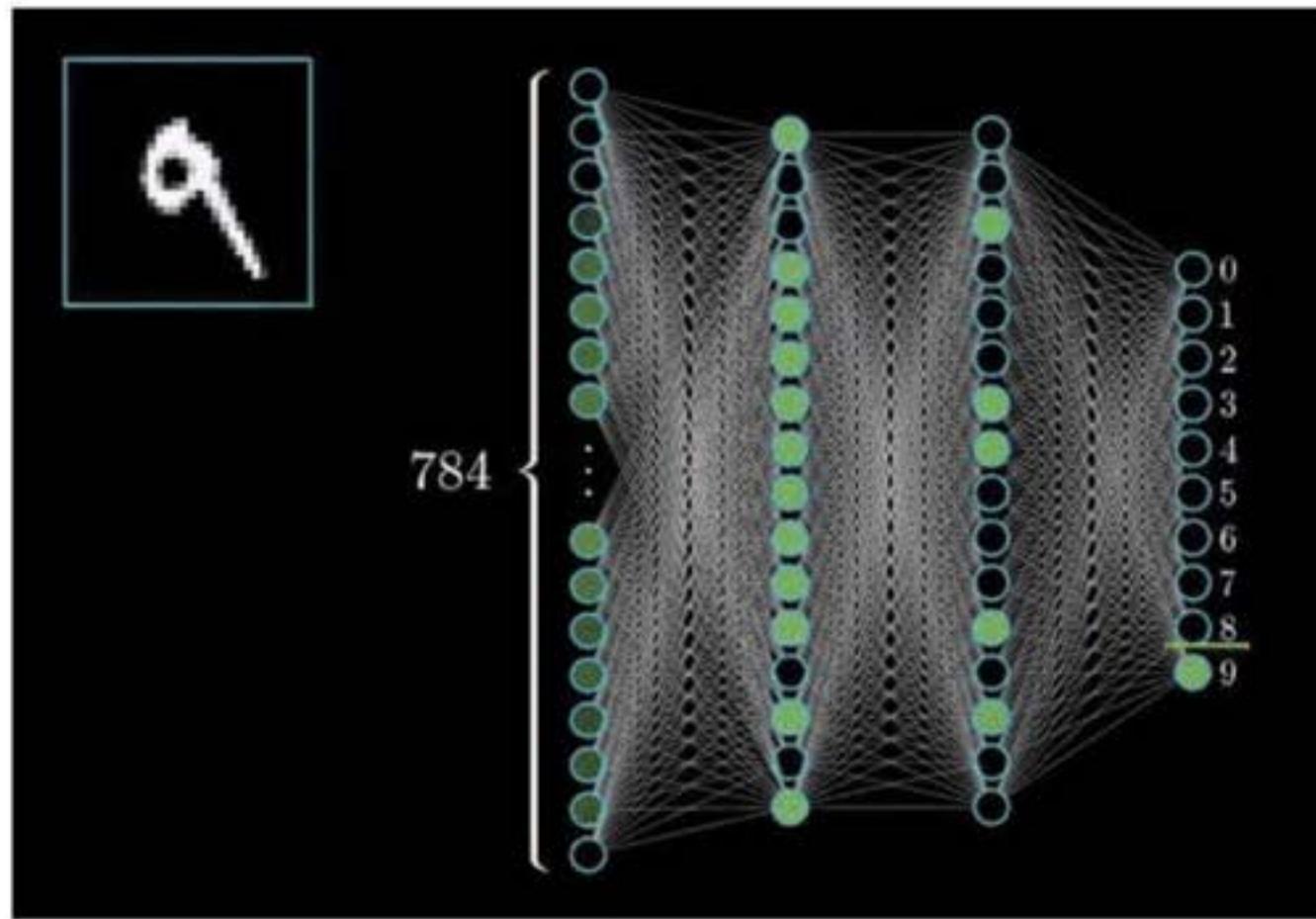


HOW?

0	4	1	9	2	1	3	1	4	3
5	3	6	1	7	2	8	6	9	4
0	9	1	1	2	4	3	2	7	3
8	6	9	0	5	6	0	7	6	1
8	7	9	3	9	8	5	9	3	3
0	7	4	9	8	0	9	4	1	4
4	6	0	4	5	6	1	0	0	1
7	1	6	3	0	2	1	1	7	9
0	2	6	7	8	3	9	0	4	6
7	4	6	8	0	7	8	3	1	5



28 x 28
784 pixels



WHEN TO AND WHEN NOT TO?

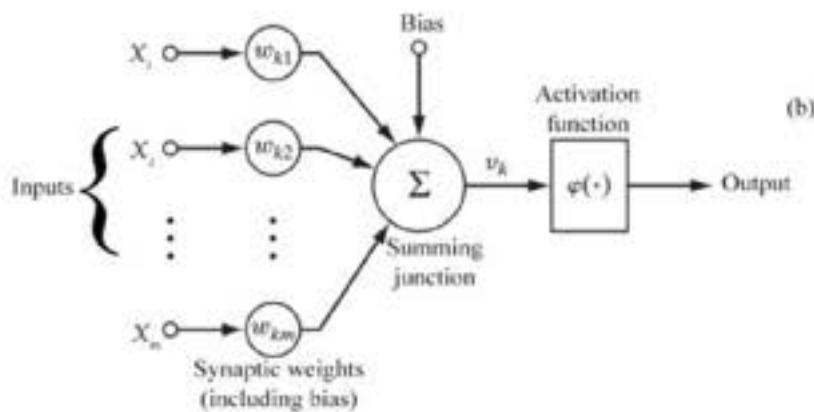
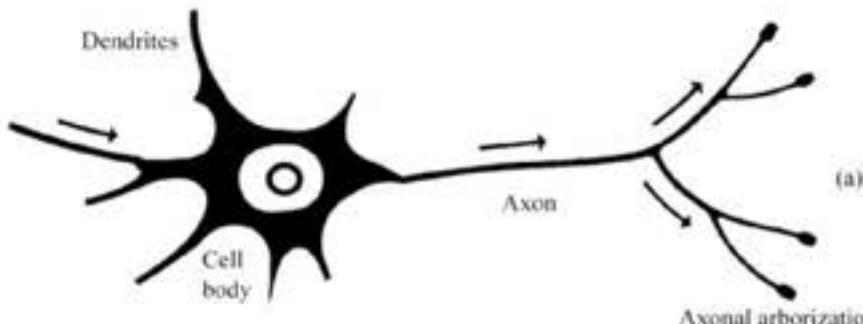
- When data is not well structured and is complex.
- For large amounts of data.



WHERE?

- Face Recognition
- Speech Recognition
- Fraud Detection
- Signature Verification
- Image Processing
- Forecasting

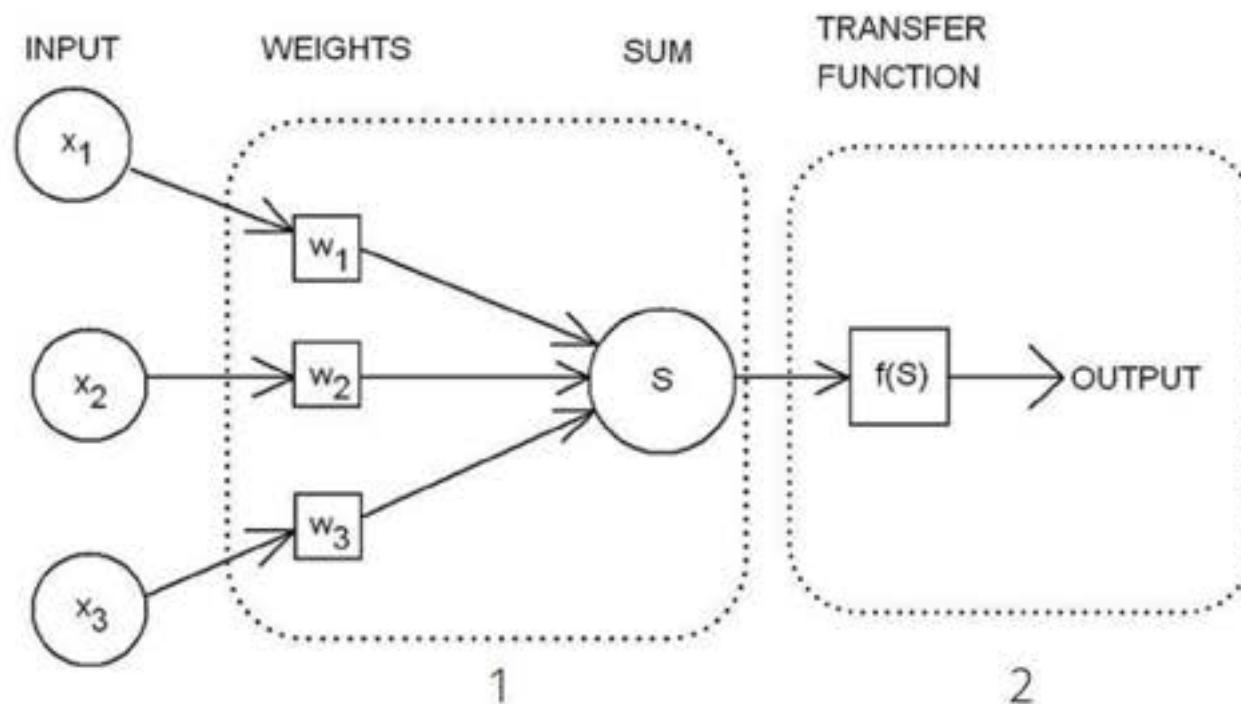
Comparison



Neurons are nerve impulse-conducting cells that make up nerves, brain and spinal column

A neuron is a mathematical function that model the functioning of a biological neuron.

Rosenblatt's perceptron



Neural Networks

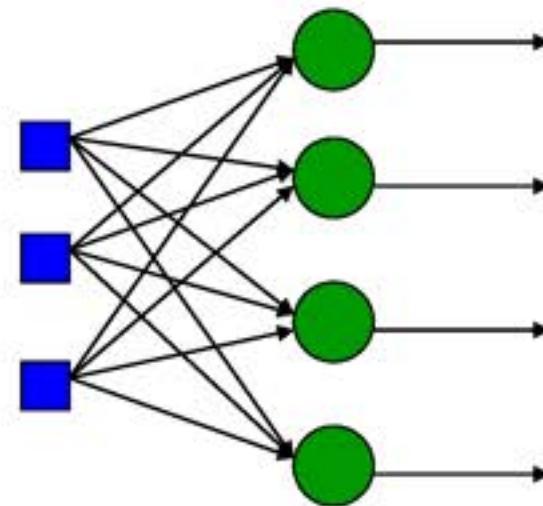
- Artificial neural network (ANN) is a machine learning approach that models human brain and consists of a number of artificial neurons.
- Each neuron in ANN receives a number of inputs.
- An activation function is applied to these inputs which results in activation level of neuron (output value of the neuron).
- Knowledge about the learning task is given in the form of examples called training examples.

Contd..

- An Artificial Neural Network is specified by:
 - **neuron model**: the information processing unit of the NN,
 - **an architecture**: a set of neurons and links connecting neurons. Each link has a weight,
 - **a learning algorithm**: used for training the NN by modifying the weights in order to model a particular learning task correctly on the training examples.
- The aim is to obtain a NN that is trained and generalizes well.
- It should behave correctly on new instances of the learning task.

Single Layer Feed-forward

*Input layer
of
source nodes*

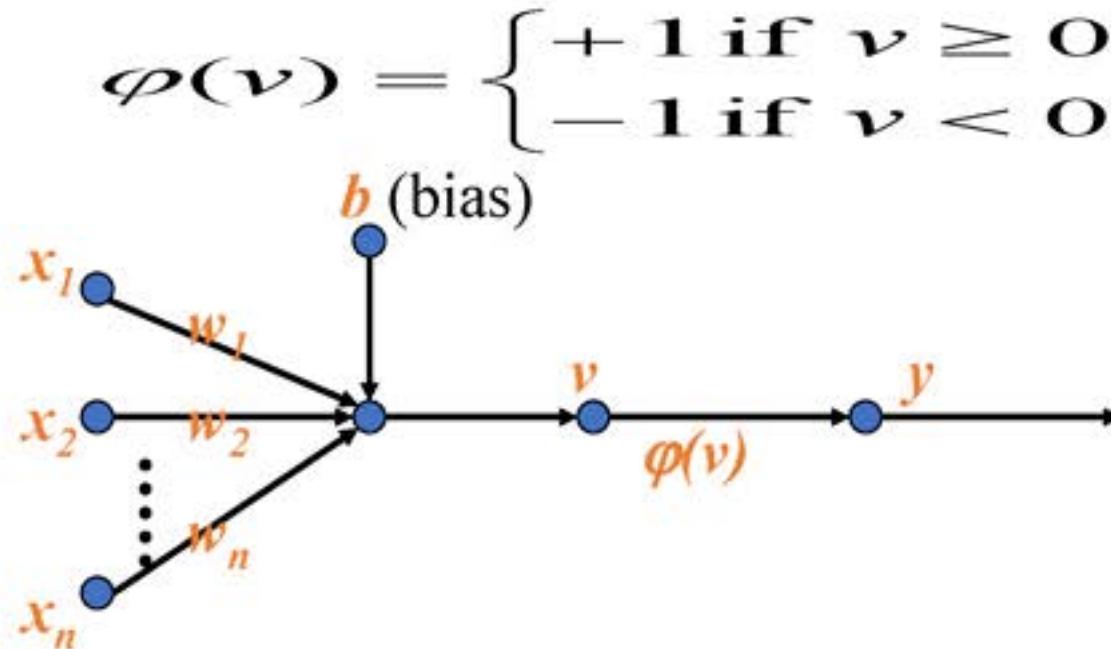


*Output layer
of
neurons*

Perceptron: Neuron Model

(Special form of single layer feed forward)

- The perceptron was first proposed by Rosenblatt (1958) is a simple neuron that is used to classify its input into one of two categories.
- A perceptron uses a **step function** that returns +1 if weighted sum of its input ≥ 0 and -1 otherwise



Perceptron for Classification

- used for binary classification.
- First train a perceptron for a classification task.
 - Find suitable weights in such a way that the training examples are correctly classified.
 - Geometrically try to find a hyper-plane that separates the examples of the two classes.
- can only model **linearly separable classes**.
- When the two classes are not linearly separable, it may be desirable to obtain a linear separator that minimizes the mean squared error.
- Given training examples of classes C_1, C_2 train the perceptron in such a way that :
 - *If the output of the perceptron is +1 then the input is assigned to class C_1*
 - *If the output is -1 then the input is assigned to C_2*

Learning Process for Perceptron

- Initially assign random weights to inputs between -0.5 and +0.5
- Training data is presented to perceptron and its output is observed.
- If output is incorrect, the weights are adjusted accordingly using following formula.

$$w_i \leftarrow w_i + (a * x_i * e), \text{ where 'e' is error produced}$$

and 'a' ($-1 < a < 1$) is learning rate

- 'a' is defined as 0 if output is correct, it is +ve, if output is too low and -ve, if output is too high.
- Once the modification to weights has taken place, the next piece of training data is used in the same way.
- Once all the training data have been applied, the process starts again until all the weights are correct and all errors are zero.
- Each iteration of this process is known as an epoch.

Perceptron: Limitations

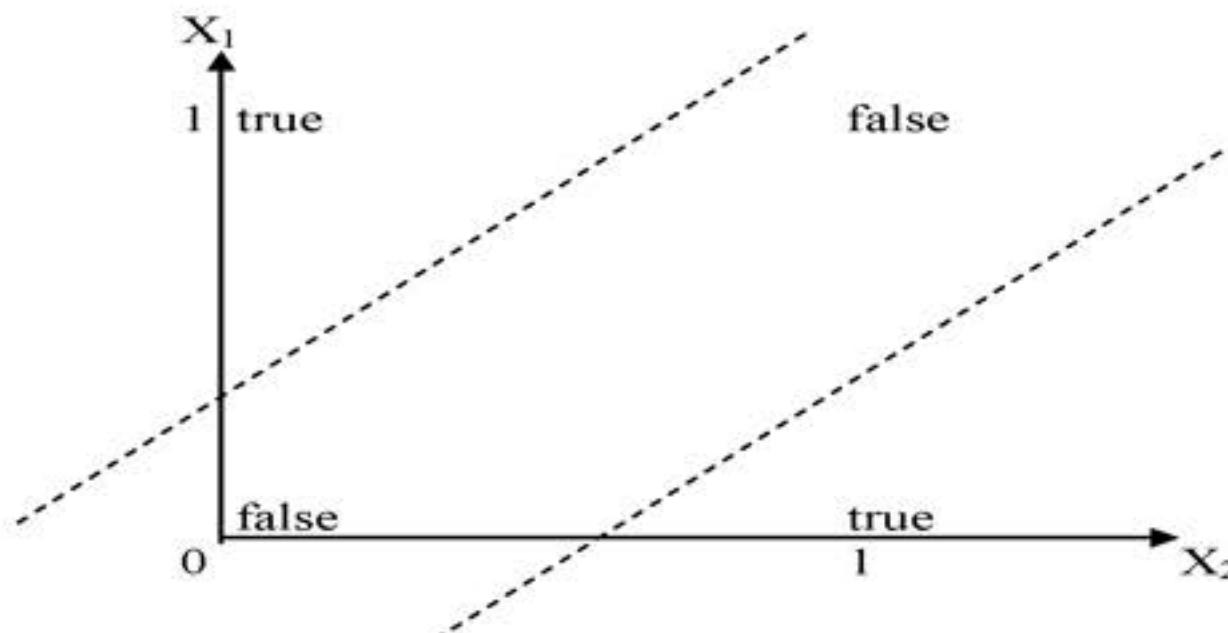
- The perceptron can only model linearly separable functions,
 - those functions which can be drawn in 2-dim graph and single straight line separates values in two part.
- Boolean functions given below are linearly separable:
 - AND
 - OR
 - COMPLEMENT
- It cannot model XOR function as it is non linearly separable.
 - When the two classes are not linearly separable, it may be desirable to obtain a linear separator that minimizes the mean squared error.

XOR – Non linearly separable function

- A typical example of non-linearly separable function is the XOR that computes the logical **exclusive or..**
- This function takes two input arguments with values in $\{0,1\}$ and returns one output in $\{0,1\}$,
- Here 0 and 1 are encoding of the truth values **false** and **true**,
- The output is **true** if and only if the two inputs have different truth values.
- XOR is non linearly separable function which can not be modeled by perceptron.
- For such functions we have to use multi layer feed-forward network.

Input		Output
X_1	X_2	$X_1 \text{ XOR } X_2$
0	0	0
0	1	1
1	0	1
1	1	0

These two classes (true and false) cannot be separated using a line. Hence XOR is non linearly separable.



Activation function

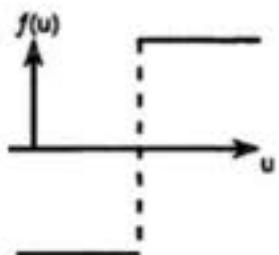
An activation function is a function used in artificial neural networks which outputs a small value for small inputs, and a larger value if its inputs exceed a threshold.

Activation functions are useful because they add non-linearities into neural networks, allowing the neural networks to learn powerful operations.

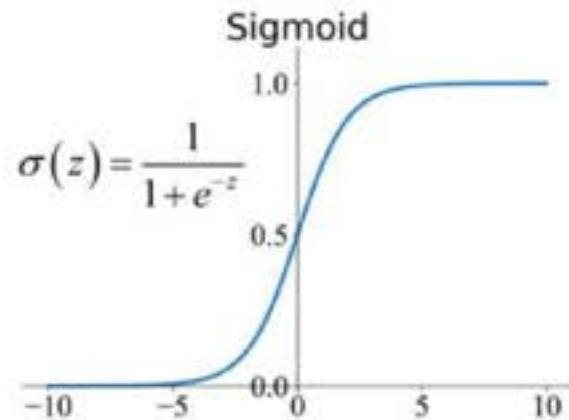
Activation functions

Name	Plot	Equation
Identity		$f(x) = x$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$

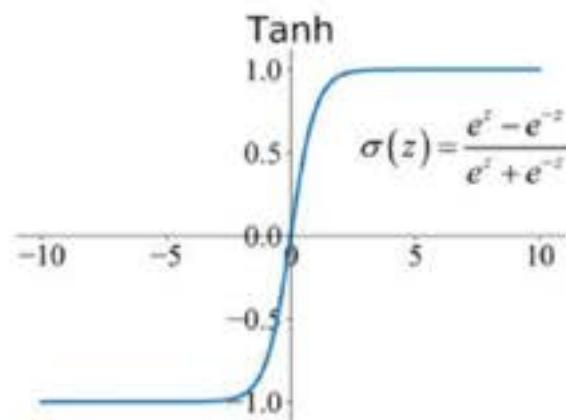
Bipolar step



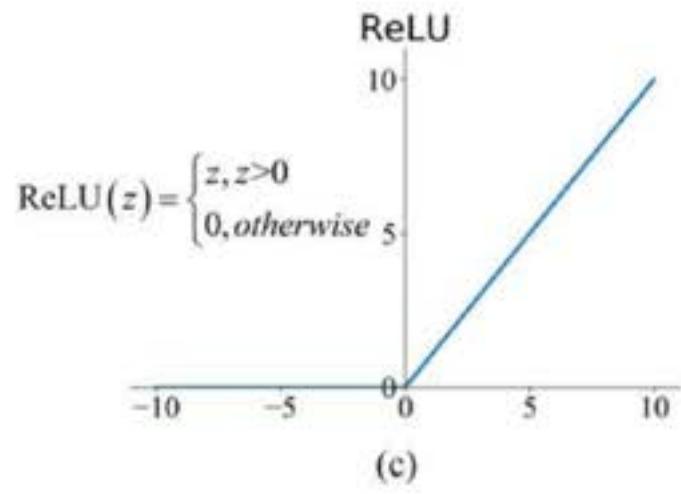
$$f(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{if } x < 0 \end{cases}$$



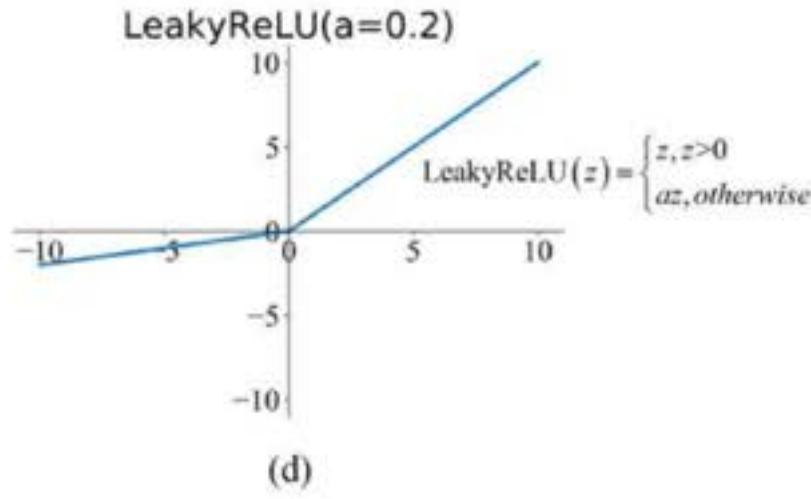
(a)



(b)



(c)



(d)

The need for Bias

Bias allows you to shift the activation function by adding a constant

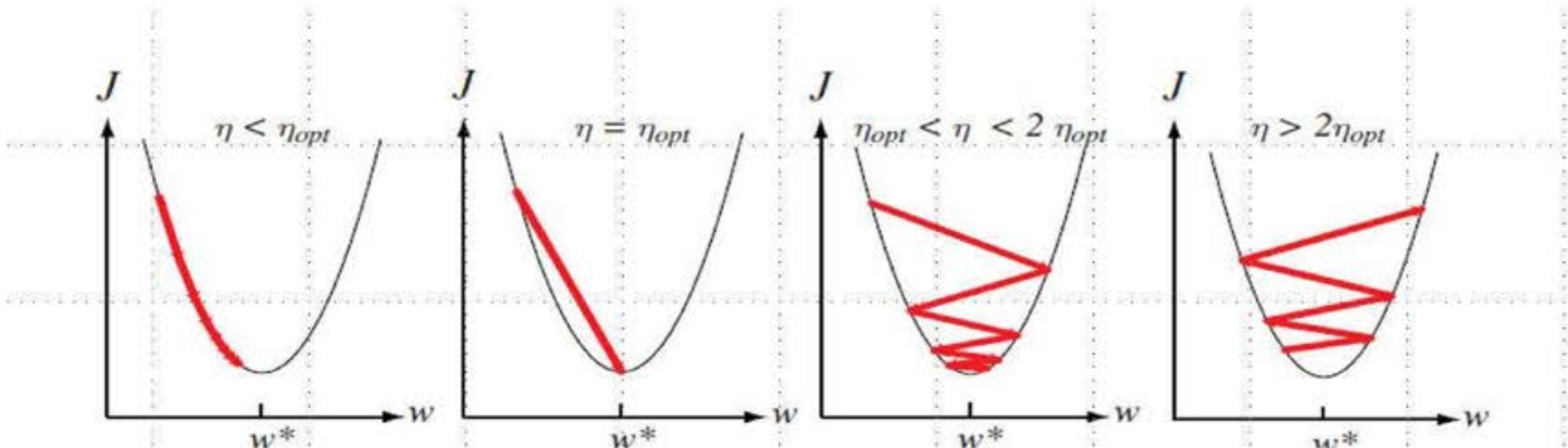
$$y = (m*x) + c$$

$$\text{Output} = (W*x) + b(\text{bias})$$

helps the model in a way that it can fit best for the given data

Bias is b sometime referred to as w_o

Learning Rate



A small learning rate requires many updates before reaching the minimum point

The optimal learning rate swiftly reaches the minimum point

Too large of a learning rate causes drastic updates which lead to divergent behaviors

Thank you !!!!!



Machine Learning (19CSE305)

Backpropagation



Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Backpropagation

Neural Networks

- Artificial neural network (ANN) is a machine learning approach that models human brain and consists of a number of artificial neurons.
- Each neuron in ANN receives a number of inputs.
- An activation function is applied to these inputs which results in activation level of neuron (output value of the neuron).
- Knowledge about the learning task is given in the form of examples called training examples.

Contd..

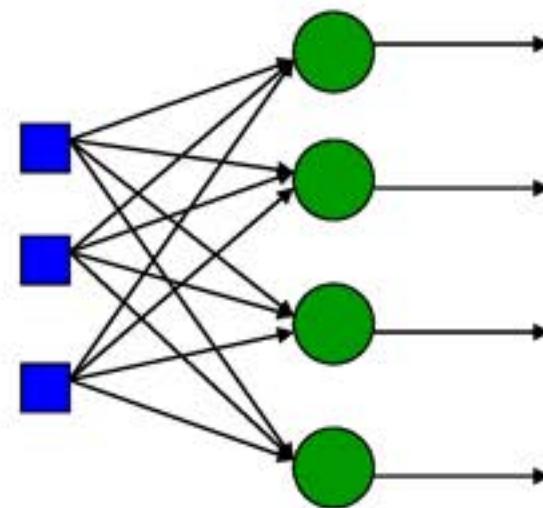
- An Artificial Neural Network is specified by:
 - **neuron model**: the information processing unit of the NN,
 - **an architecture**: a set of neurons and links connecting neurons. Each link has a weight,
 - **a learning algorithm**: used for training the NN by modifying the weights in order to model a particular learning task correctly on the training examples.
- The aim is to obtain a NN that is trained and generalizes well.
- It should behave correctly on new instances of the learning task.

Network Architectures

- Three different classes of network architectures
 - single-layer feed-forward
 - multi-layer feed-forward
 - recurrent
- The **architecture** of a neural network is linked with the learning algorithm used to train

Single Layer Feed-forward

*Input layer
of
source nodes*

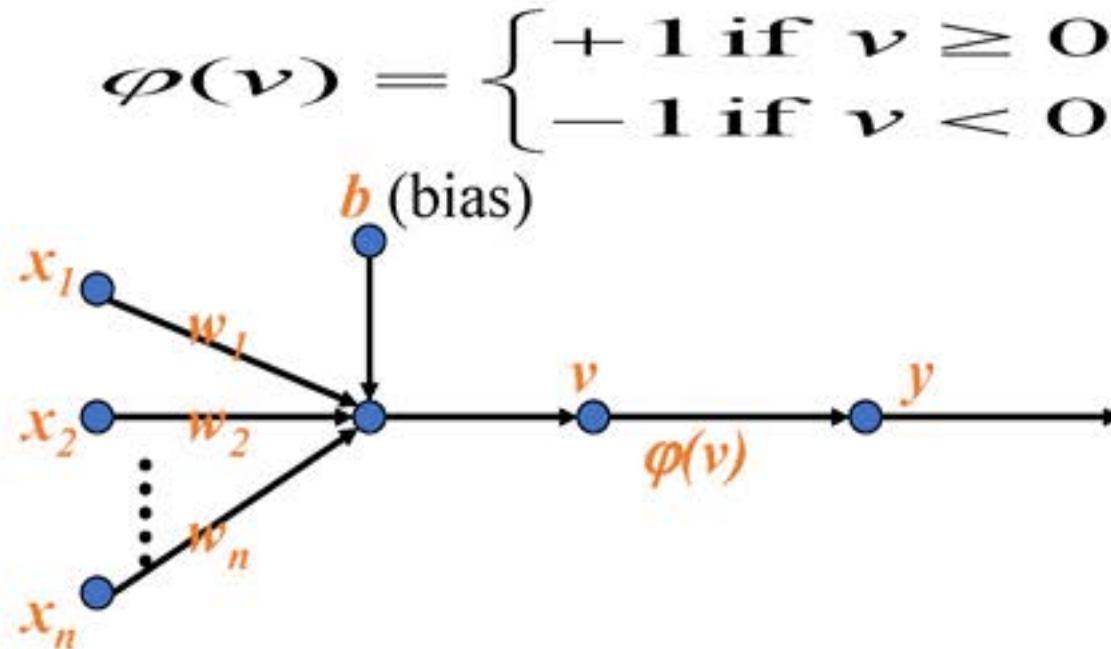


*Output layer
of
neurons*

Perceptron: Neuron Model

(Special form of single layer feed forward)

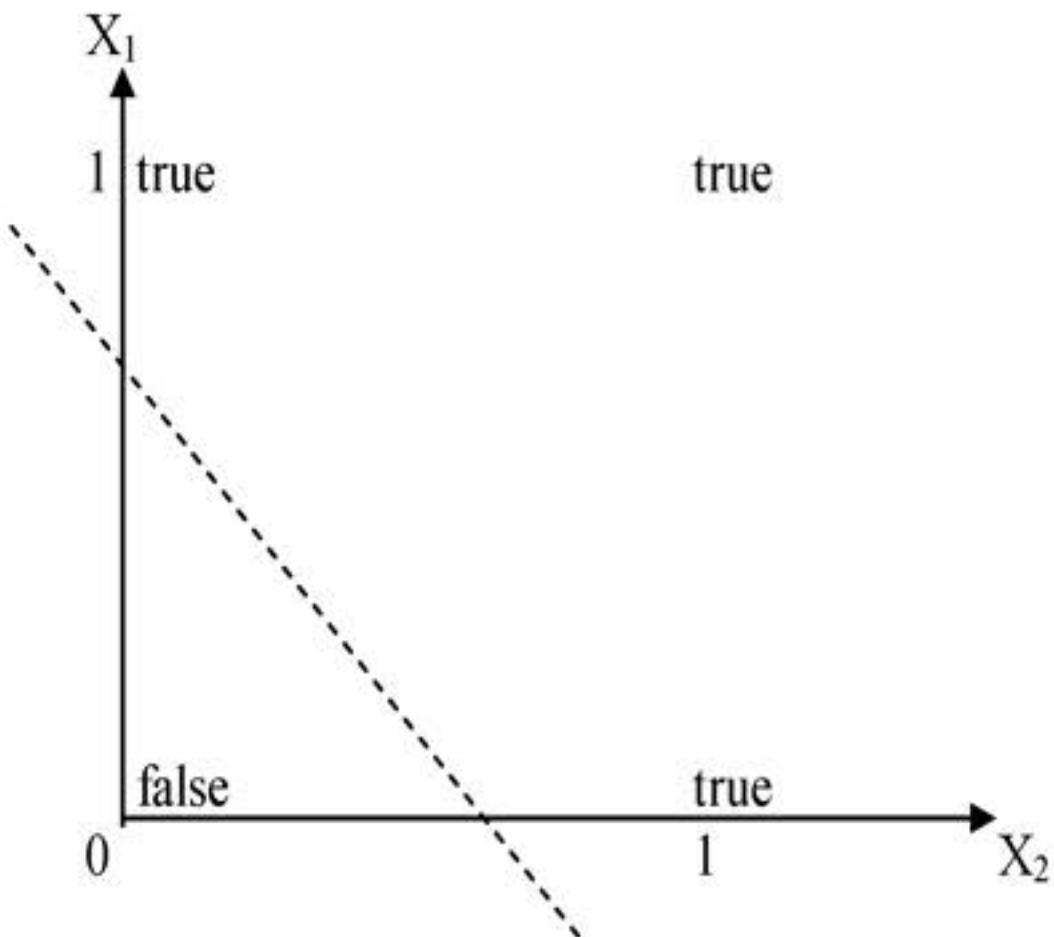
- The perceptron was first proposed by Rosenblatt (1958) is a simple neuron that is used to classify its input into one of two categories.
- A perceptron uses a **step function** that returns +1 if weighted sum of its input ≥ 0 and -1 otherwise



Perceptron for Classification

- used for binary classification.
- First train a perceptron for a classification task.
 - Find suitable weights in such a way that the training examples are correctly classified.
 - Geometrically try to find a hyper-plane that separates the examples of the two classes.
- can only model **linearly separable classes**.
- When the two classes are not linearly separable, it may be desirable to obtain a linear separator that minimizes the mean squared error.
- Given training examples of classes C_1, C_2 train the perceptron in such a way that :
 - *If the output of the perceptron is +1 then the input is assigned to class C_1*
 - *If the output is -1 then the input is assigned to C_2*

Boolean function OR – Linearly separable



Perceptron: Limitations

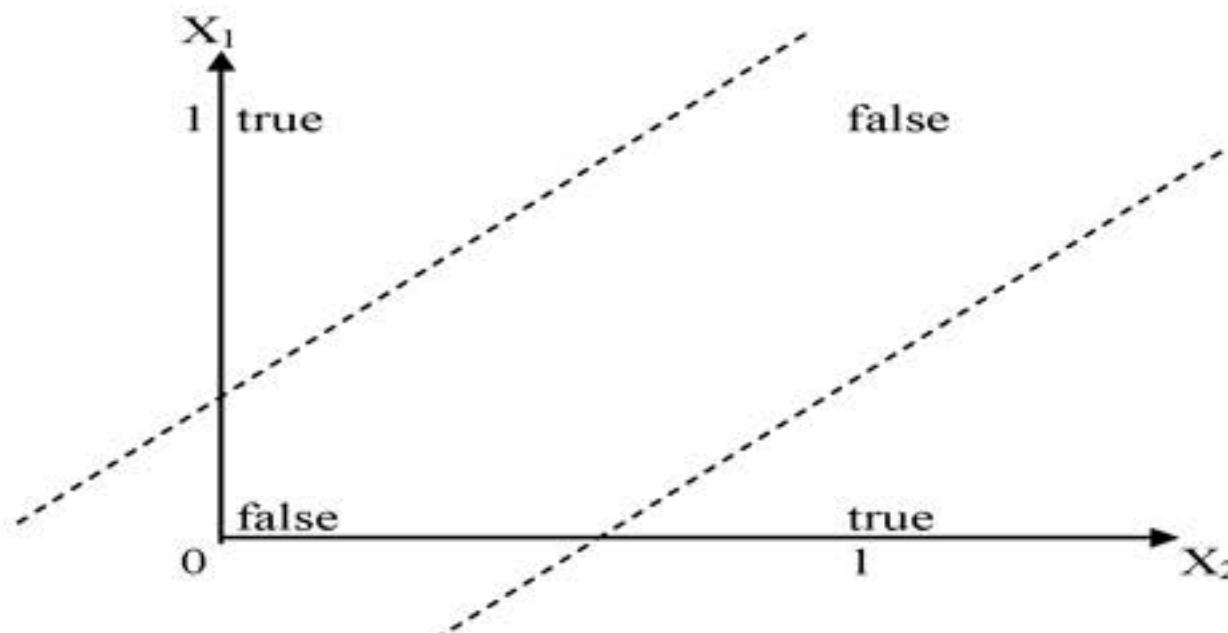
- The perceptron can only model linearly separable functions,
 - those functions which can be drawn in 2-dim graph and single straight line separates values in two part.
- Boolean functions given below are linearly separable:
 - AND
 - OR
 - COMPLEMENT
- It cannot model XOR function as it is non linearly separable.
 - When the two classes are not linearly separable, it may be desirable to obtain a linear separator that minimizes the mean squared error.

XOR – Non linearly separable function

- A typical example of non-linearly separable function is the XOR that computes the logical **exclusive or..**
- This function takes two input arguments with values in $\{0,1\}$ and returns one output in $\{0,1\}$,
- Here 0 and 1 are encoding of the truth values **false** and **true**,
- The output is **true** if and only if the two inputs have different truth values.
- XOR is non linearly separable function which can not be modeled by perceptron.
- For such functions we have to use multi layer feed-forward network.

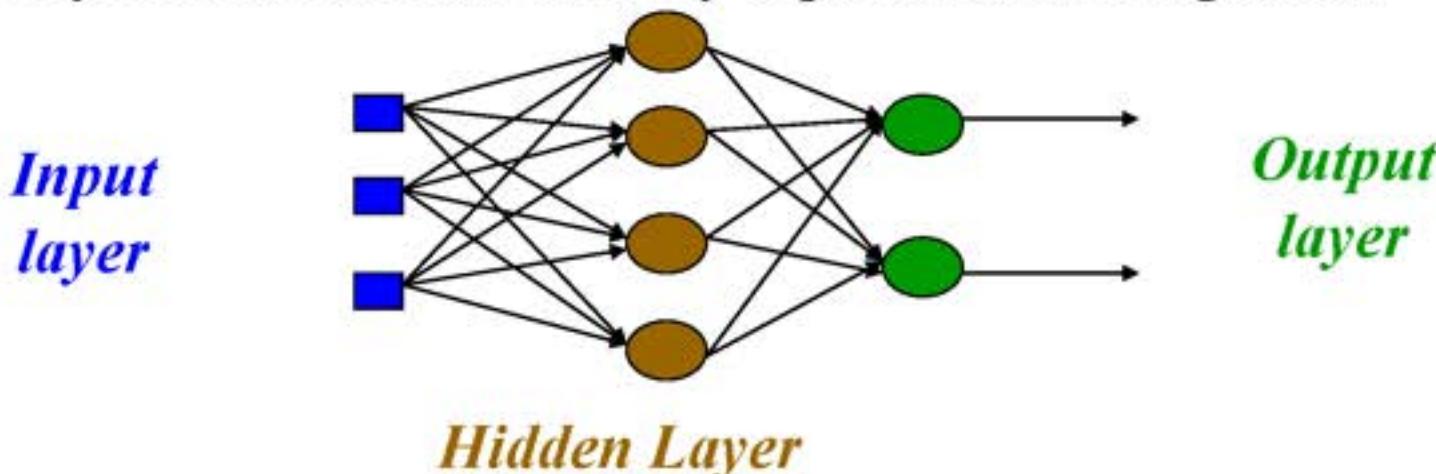
Input		Output
X_1	X_2	$X_1 \text{ XOR } X_2$
0	0	0
0	1	1
1	0	1
1	1	0

These two classes (true and false) cannot be separated using a line. Hence XOR is non linearly separable.



Multi layer feed-forward NN (FFNN)

- FFNN is a more general network architecture, where there are hidden layers between input and output layers.
- Hidden nodes do not directly receive inputs nor send outputs to the external environment.
- FFNNs overcome the limitation of single-layer NN.
- They can handle non-linearly separable learning tasks.



FFNN NEURON MODEL

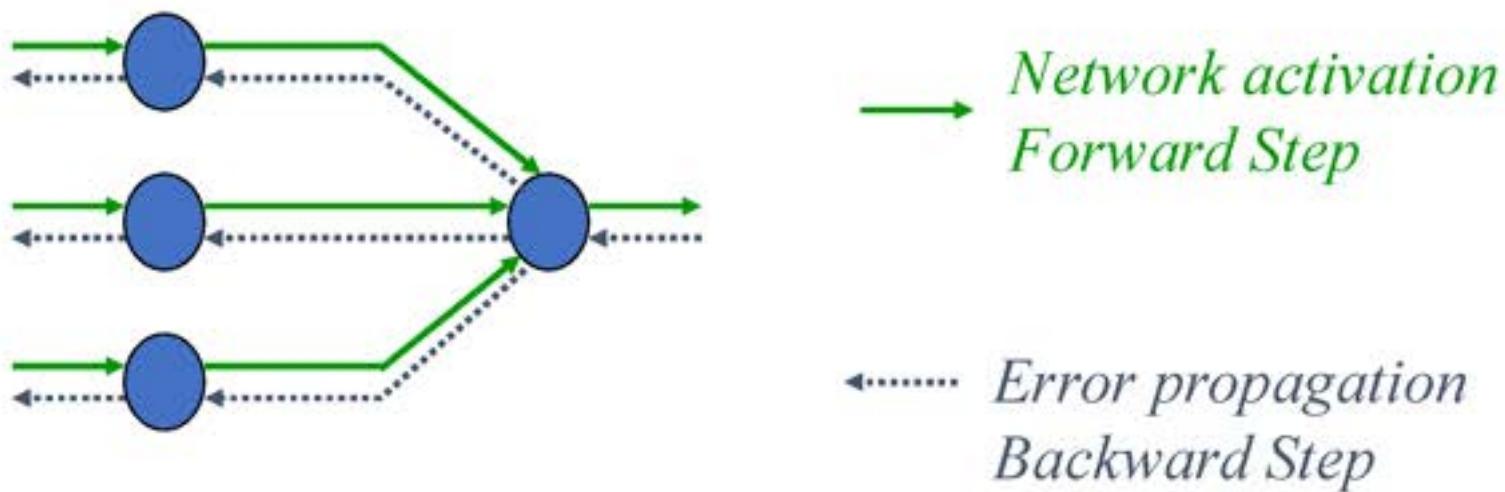
- The classical learning algorithm of FFNN is based on the gradient descent method.
- For this reason the activation function used in FFNN are continuous functions of the weights, differentiable everywhere.
- Eg. **sigmoid function**

Training Algorithm: Backpropagation

- The Backpropagation algorithm learns in the same way as single perceptron.
- It searches for weight values that minimize the total error of the network over the set of training examples (training set).
- Backpropagation consists of the repeated application of the following two passes:
 - **Forward pass:** In this step, the network is activated on one example and the error of (each neuron of) the output layer is computed.
 - **Backward pass:** in this step the network error is used for updating the weights. The error is propagated backwards from the output layer through the network layer by layer. This is done by recursively computing the local gradient of each neuron.

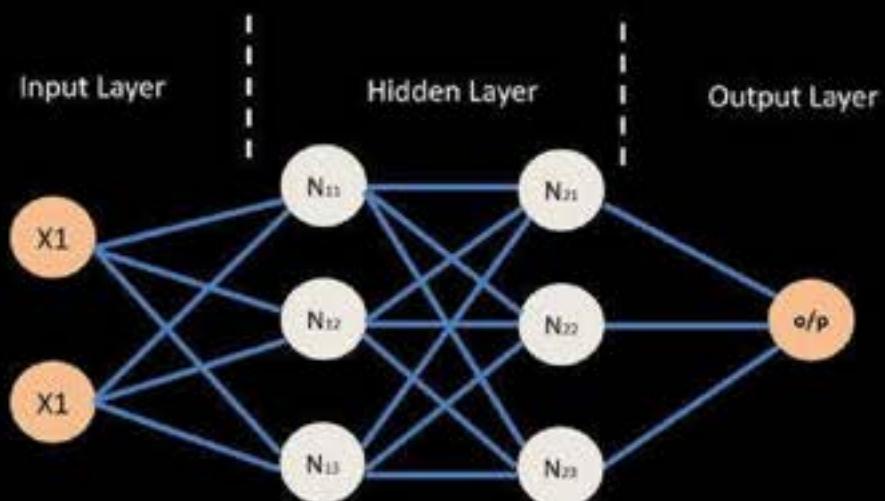
Backpropagation

- Back-propagation training algorithm



- Backpropagation adjusts the weights of the NN in order to minimize the network total mean squared error.

Neural Network – Backpropagation

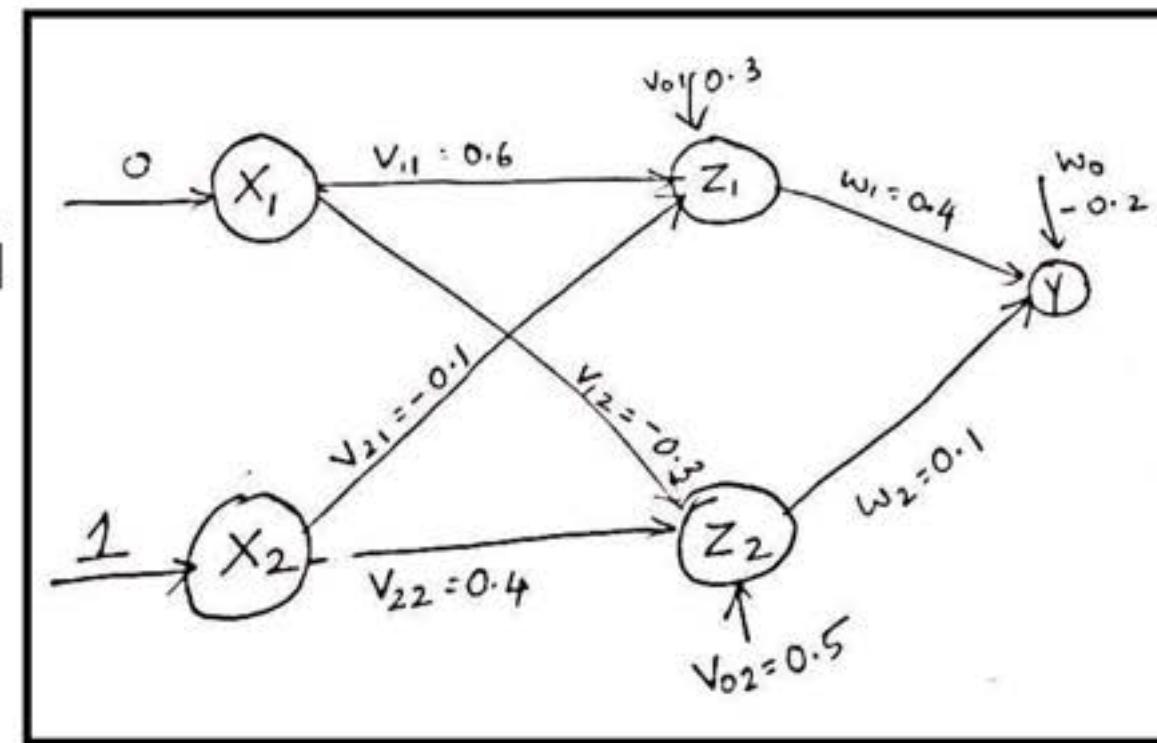


© machinelearningknowledge.ai

- Using back-propagation network, find the weights for the net. It is presented with the input pattern $[0, 1]$ and the target output is 1. Use a learning rate $\alpha = 0.25$ and sigmoidal activation function. initial weights are $[v_{11} \ v_{21} \ v_{01}] = [0.6 \ -0.1 \ 0.3]$

$[v_{12} \ v_{22} \ v_{02}] = [-0.3 \ 0.4 \ 0.5]$ and $[w_1 \ w_2 \ w_0] = [0.4 \ 0.1 \ -0.2]$, and the learning rate is $\alpha = 0.25$. Activation function used is binary sigmoidal activation function and is given by

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



Calculate the net input: For z_1 layer

$$\begin{aligned} z_{in1} &= v_{01} + x_1 v_{11} + x_2 v_{21} \\ &= 0.3 + 0 \times 0.6 + 1 \times -0.1 = 0.2 \end{aligned}$$

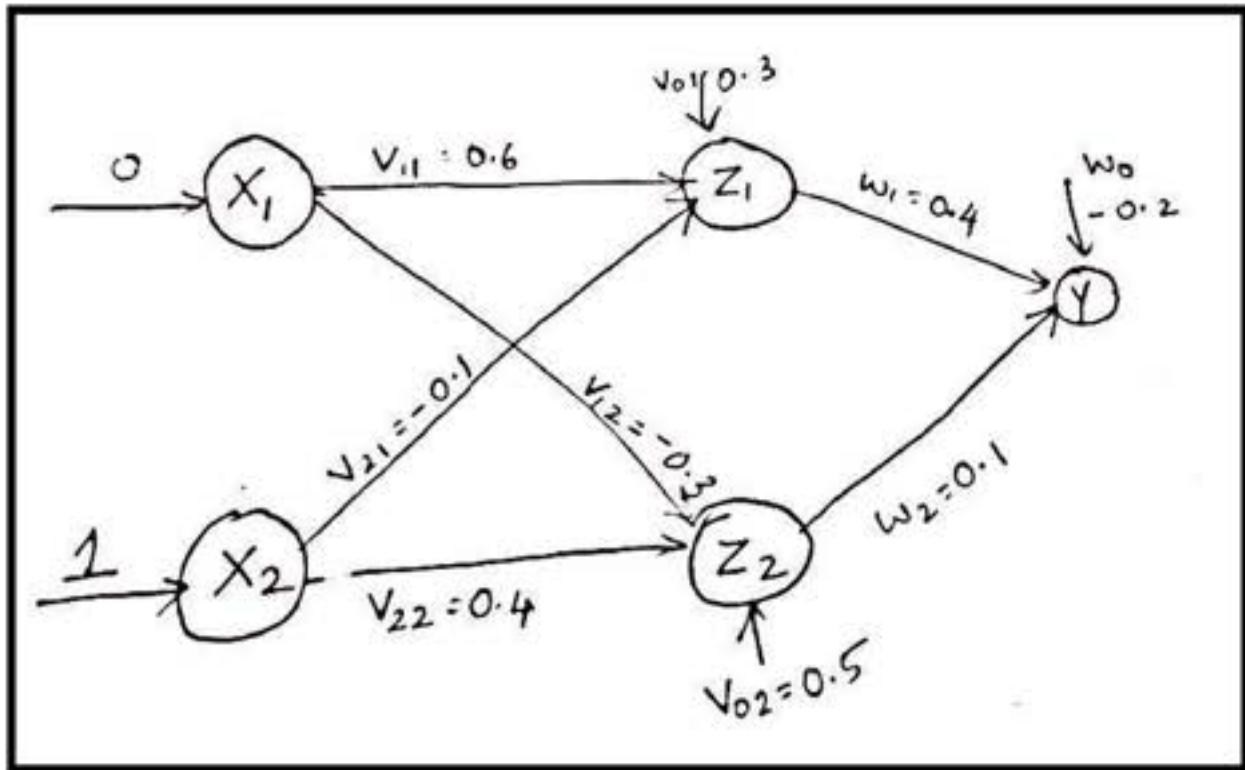
For z_2 layer

$$\begin{aligned} z_{in2} &= v_{02} + x_1 v_{12} + x_2 v_{22} \\ &= 0.5 + 0 \times -0.3 + 1 \times 0.4 = 0.9 \end{aligned}$$

Applying activation to calculate the output, we obtain

$$z_1 = f(z_{in1}) = \frac{1}{1 + e^{-z_{in1}}} = \frac{1}{1 + e^{-0.2}} = 0.5498$$

$$z_2 = f(z_{in2}) = \frac{1}{1 + e^{-z_{in2}}} = \frac{1}{1 + e^{-0.9}} = 0.7109$$



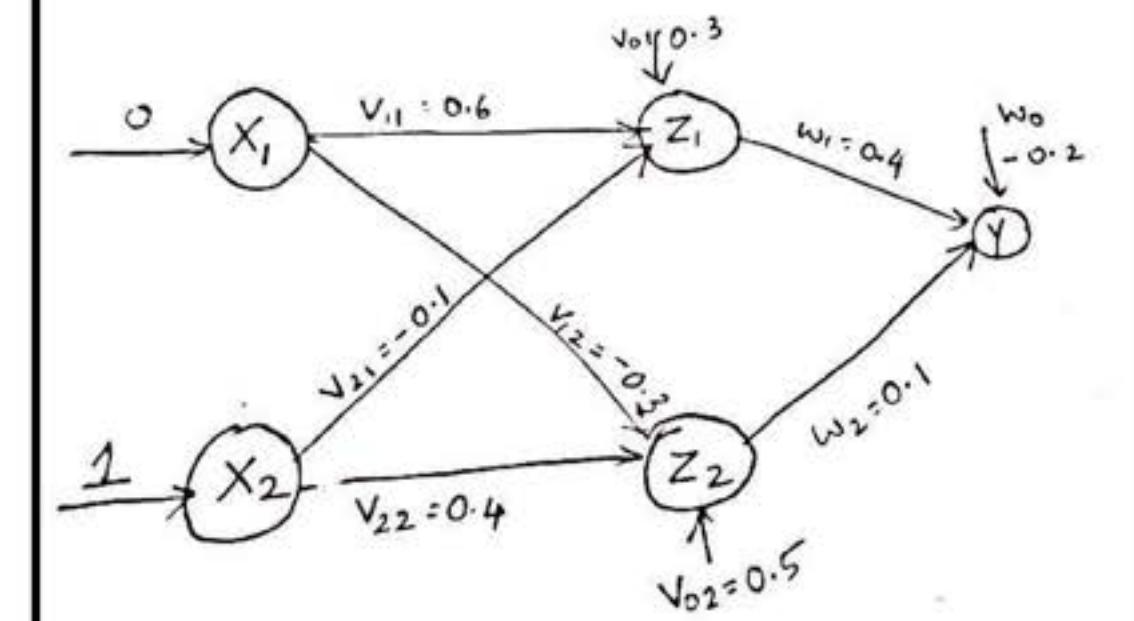
Calculate the net input entering the output layer.

For y layer

$$\begin{aligned}y_{in} &= w_0 + z_1 w_1 + z_2 w_2 \\&= -0.2 + 0.5498 \times 0.4 + 0.7109 \times 0.1 \\&= 0.09101\end{aligned}$$

Applying activations to calculate the output, we obtain

$$y = f(y_{in}) = \frac{1}{1 + e^{-y_{in}}} = \frac{1}{1 + e^{-0.09101}} = 0.5227$$



- Compute the error portion δ_k :

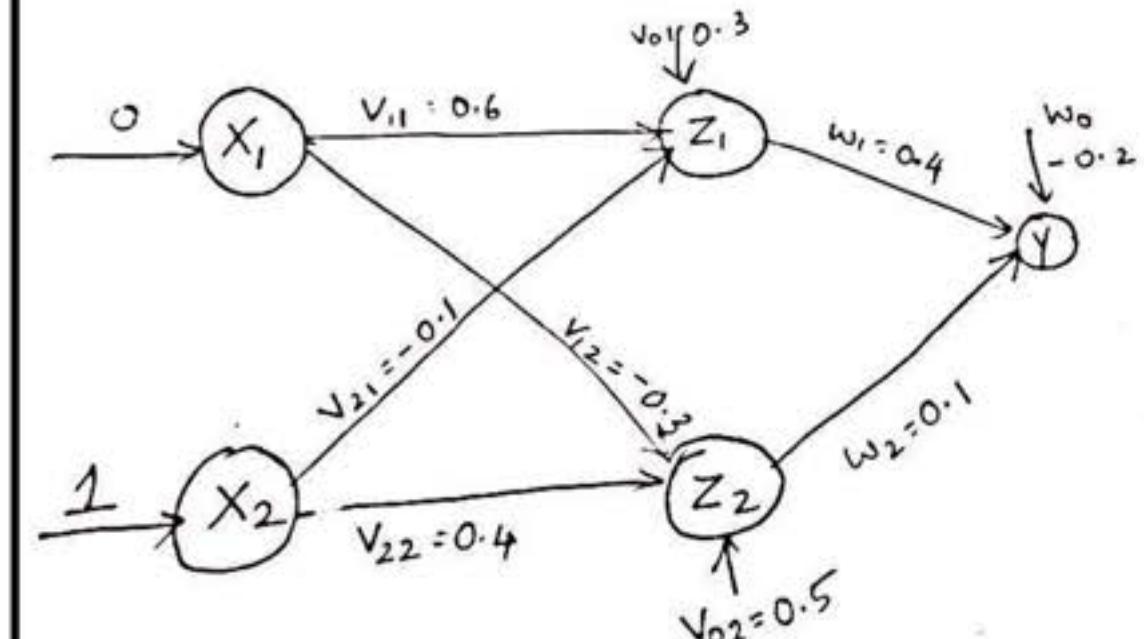
$$\delta_k = (t_k - y_k)f'(y_{ink})$$

Now

$$f'(y_{in}) = f(y_{in})[1 - f(y_{in})] = 0.5227[1 - 0.5227]$$

$$f'(y_{in}) = 0.2495$$

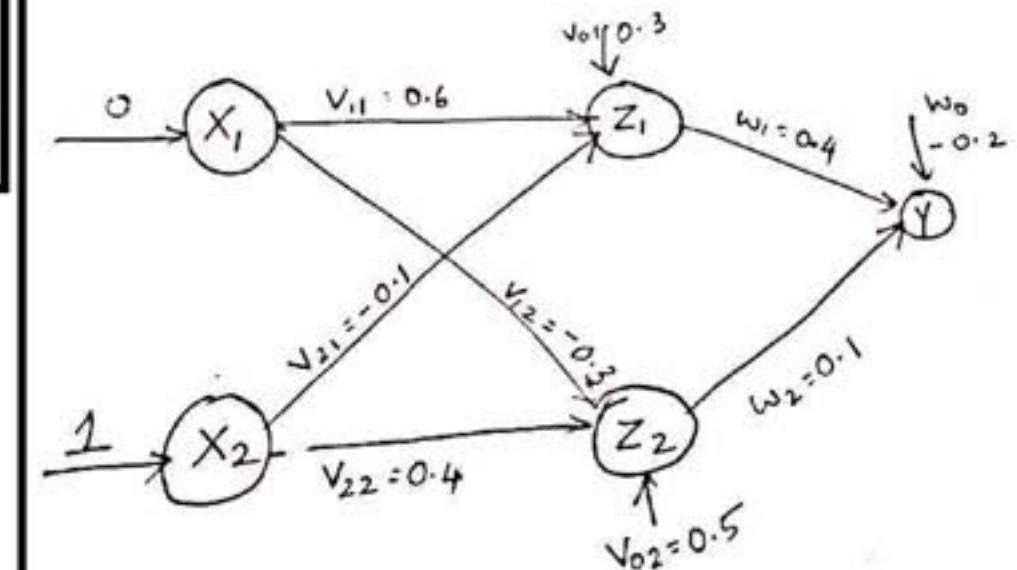
$$\delta_1 = (1 - 0.5227)(0.2495) = 0.1191$$



$$\Delta w_1 = \alpha \delta_1 z_1 = 0.25 \times 0.1191 \times 0.54\% \\ = 0.0164$$

$$\Delta w_2 = \alpha \delta_1 z_2 = 0.25 \times 0.1191 \times 0.71\% \\ = 0.02117$$

$$\Delta w_0 = \alpha \delta_1 = 0.25 \times 0.1191 = 0.02978$$



- Compute the error portion δ_j between input and hidden layer ($j = 1$ to 2):

$$\delta_j = \delta_{inj} f'(z_{inj})$$

$$\delta_{inj} = \sum_{k=1}^m \delta_k w_{jk}$$

$$\delta_{inj} = \delta_1 w_{j1} \quad [\because \text{only one output neuron}]$$

$$\Rightarrow \delta_{in1} = \delta_1 w_{11} = 0.1191 \times 0.4 = 0.04764$$

$$\Rightarrow \delta_{in2} = \delta_1 w_{21} = 0.1191 \times 0.1 = 0.01191$$

Error, $\delta_1 = \delta_{in1} f'(z_{in1})$

$$\begin{aligned}f'(z_{in1}) &= f(z_{in1}) [1 - f(z_{in1})] \\&= 0.5498[1 - 0.5498] = 0.2475\end{aligned}$$

$$\begin{aligned}\delta_1 &= \delta_{in1} f'(z_{in1}) \\&= 0.04764 \times 0.2475 = 0.0118\end{aligned}$$

Error, $\delta_2 = \delta_{in2} f'(z_{in2})$

$$\begin{aligned}f'(z_{in2}) &= f(z_{in2}) [1 - f(z_{in2})] \\&= 0.7109[1 - 0.7109] = 0.2055\end{aligned}$$

$$\begin{aligned}\delta_2 &= \delta_{in2} f'(z_{in2}) \\&= 0.01191 \times 0.2055 = 0.00245\end{aligned}$$

Now find the changes in weights between input and hidden layer:

$$\Delta v_{11} = \alpha \delta_1 x_1 = 0.25 \times 0.0118 \times 0 = 0$$

$$\Delta v_{21} = \alpha \delta_1 x_2 = 0.25 \times 0.0118 \times 1 = 0.00295$$

$$\Delta v_{01} = \alpha \delta_1 = 0.25 \times 0.0118 = 0.00295$$

$$\Delta v_{12} = \alpha \delta_2 x_1 = 0.25 \times 0.00245 \times 0 = 0$$

$$\Delta v_{22} = \alpha \delta_2 x_2 = 0.25 \times 0.00245 \times 1 = 0.0006125$$

$$\Delta v_{02} = \alpha \delta_2 = 0.25 \times 0.00245 = 0.0006125$$

- Compute the final weights of the network:

$$v_{11}(\text{new}) = v_{11}(\text{old}) + \Delta v_{11} = 0.6 + 0 = 0.6$$

$$v_{12}(\text{new}) = v_{12}(\text{old}) + \Delta v_{12} = -0.3 + 0 = -0.3$$

$$\begin{aligned}v_{21}(\text{new}) &= v_{21}(\text{old}) + \Delta v_{21} \\&= -0.1 + 0.00295 = -0.09705\end{aligned}$$

$$\begin{aligned}v_{22}(\text{new}) &= v_{22}(\text{old}) + \Delta v_{22} \\&= 0.4 + 0.0006125 = 0.4006125\end{aligned}$$

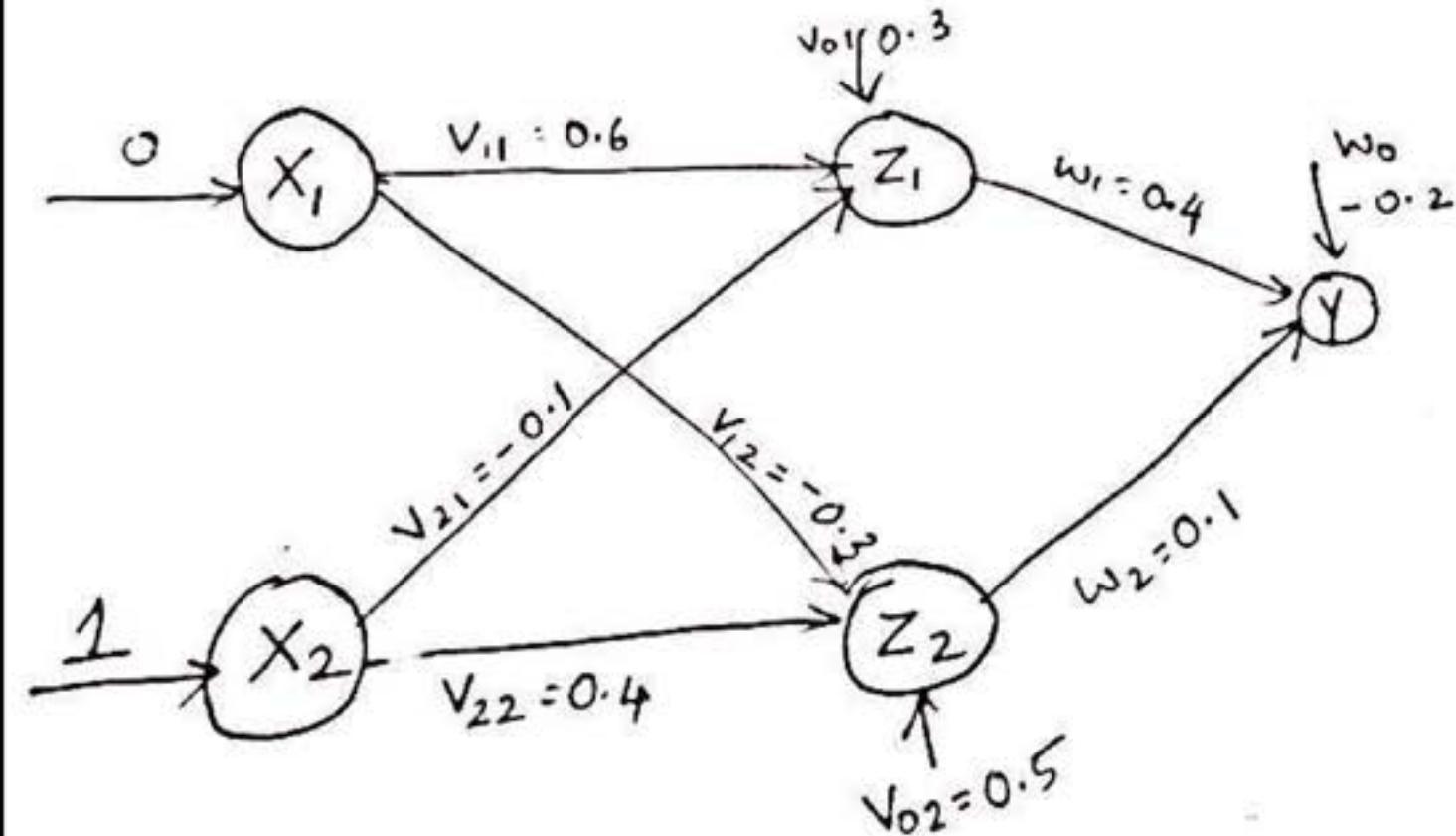
$$\begin{aligned}w_1(\text{new}) &= w_1(\text{old}) + \Delta w_1 = 0.4 + 0.0164 \\&= 0.4164\end{aligned}$$

$$\begin{aligned}w_2(\text{new}) &= w_2(\text{old}) + \Delta w_2 = 0.1 + 0.02117 \\&= 0.12117\end{aligned}$$

$$\begin{aligned}v_{01}(\text{new}) &= v_{01}(\text{old}) + \Delta v_{01} = 0.3 + 0.00295 \\&= 0.30295\end{aligned}$$

$$\begin{aligned}v_{02}(\text{new}) &= v_{02}(\text{old}) + \Delta v_{02} \\&= 0.5 + 0.0006125 = 0.5006125\end{aligned}$$

$$\begin{aligned}w_0(\text{new}) &= w_0(\text{old}) + \Delta w_0 = -0.2 + 0.02978 \\&= -0.17022\end{aligned}$$



Thank you !!!!!



Machine Learning (19CSE305)

Perceptron -Training



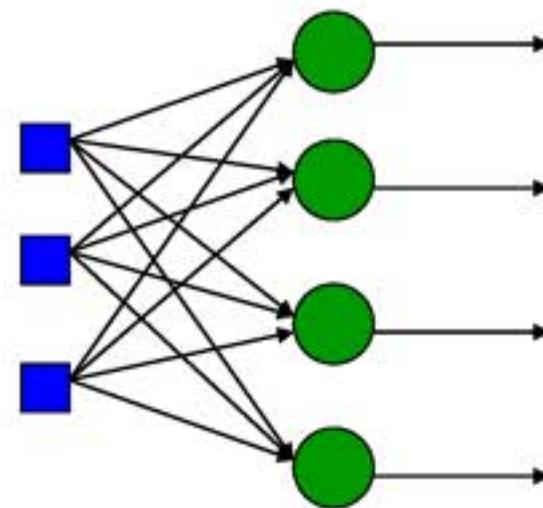
Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Perceptron

Single Layer Feed-forward

*Input layer
of
source nodes*

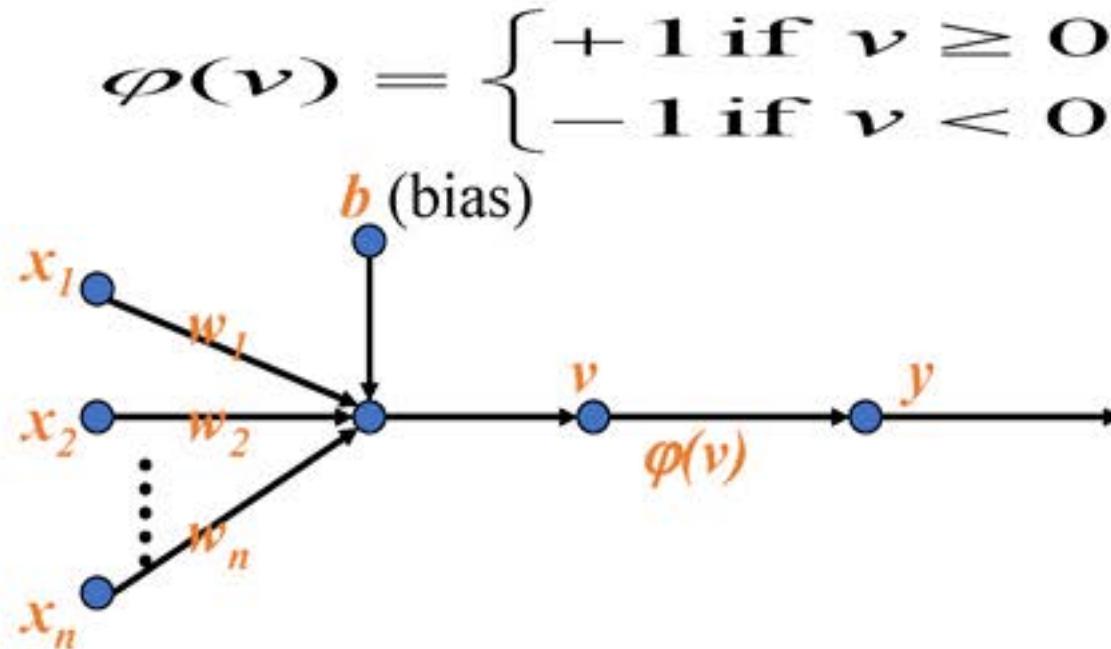


*Output layer
of
neurons*

Perceptron: Neuron Model

(Special form of single layer feed forward)

- The perceptron was first proposed by Rosenblatt (1958) is a simple neuron that is used to classify its input into one of two categories.
- A perceptron uses a **step function** that returns +1 if weighted sum of its input ≥ 0 and -1 otherwise



Perceptron for Classification

- used for binary classification.
- First train a perceptron for a classification task.
 - Find suitable weights in such a way that the training examples are correctly classified.
 - Geometrically try to find a hyper-plane that separates the examples of the two classes.
- can only model **linearly separable classes**.
- When the two classes are not linearly separable, it may be desirable to obtain a linear separator that minimizes the mean squared error.
- Given training examples of classes C_1, C_2 train the perceptron in such a way that :
 - *If the output of the perceptron is +1 then the input is assigned to class C_1*
 - *If the output is -1 then the input is assigned to C_2*

Perceptron training

Step 0: Initialize the weights and the bias (for easy calculation they can be set to zero). Also initialize the learning rate α ($0 < \alpha \leq 1$). For simplicity α is set to 1.

Step 1: Perform Steps 2–6 until the final stopping condition is false.

Step 2: Perform Steps 3–5 for each training pair indicated by $i:t$.

Step 3: The input layer containing input units is applied with identity activation functions:

$$x_i = z_i$$

Step 4: Calculate the output of the network. To do so, first obtain the net input:

$$y_{in} = b + \sum_{i=1}^n x_i w_i$$

where "n" is the number of input neurons in the input layer. Then apply activations over the net input calculated to obtain the output:

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

Step 5: *Weight and bias adjustment:* Compare the value of the actual (calculated) output and desired (target) output.

If $y \neq t$, then

$$w_i(\text{new}) = w_i(\text{old}) + \alpha x_i$$

$$b(\text{new}) = b(\text{old}) + \alpha z$$

else, we have

$$w_i(\text{new}) = w_i(\text{old})$$

$$b(\text{new}) = b(\text{old})$$

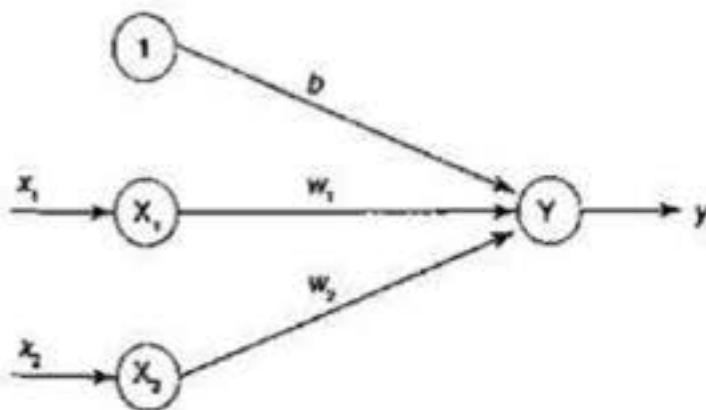
Step 6: Train the network until there is no weight change. This is the stopping condition for the network.

If this condition is not met, then start again from Step 2.

- Implement AND function using perceptron networks for bipolar inputs & targets.
- The initial weights and threshold are set to zero, $w_1 = w_2 = b = 0$ and $\theta = 0$.
- The learning rate is set equal to 1.

Table 1

x_1	x_2	t
1	1	1
1	-1	-1
-1	1	-1
-1	-1	-1



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

Epoch 1

Input 1

$$x_1 = 1, x_2 = 1, t = 1 \quad w_1 = 0, w_2 = 0, b = 0.$$

$$\alpha = 1$$

$$y_{in} = b + x_1 w_1 + x_2 w_2$$

$$= 0 + 1 \times 0 + 1 \times 0 = 0$$

$$y - f(y_{in}) = 0$$

$$t = 1, y = 0 \quad \therefore t \neq y$$

So update weights

$$b_{new} = b_{old} + \alpha t$$

$$= 0 + 1 \times 1 = 1$$

$$w_1^{new} = w_1^{old} + \alpha t x_1$$

$$= 0 + 1 \times 1 \times 1 = 1$$

$$w_2^{new} = w_2^{old} + \alpha t x_2$$

$$= 0 + 1 \times 1 \times 1 = 1$$

2nd i/p
 $x_1 = 1 \quad x_2 = -1, \quad t = -1$

$$y_{in} = b \cdot w = [1 \quad 1 \quad 1]$$

$$b = 1$$

$$\begin{aligned}y_{in} &= b + w_1 x_1 + w_2 x_2 \\&= 1 + 1 * 1 + 1 * -1 \\&= 1\end{aligned}$$

$$y = f(y_{in})$$

$$= 1$$

$$y = 1, t = -1 \quad \therefore t \neq y$$

$$\begin{aligned}w_1(\text{new}) &= 1 + 1 * (-1)(1) \\&= 0\end{aligned}$$

$$\begin{aligned}w_2(\text{new}) &= 1 + 1 * (-1)(-1) \\&= 2\end{aligned}$$

$$\begin{aligned}b(\text{new}) &= 1 + 1 * (-1) \\&= 0\end{aligned}$$

$$w = \begin{bmatrix} w_1 & w_2 & b \\ 0 & 2 & 0 \end{bmatrix}$$

3rd if

$$x_1 = -1, x_2 = 1 \quad t = -1$$

$$w_1 = 0 \quad w_2 = 2, b = 0$$

$$\begin{aligned}y_{in} &= 0 + 0 * (-1) + 2(1) \\&= 2\end{aligned}$$

$$y = f(y_{in}) = 1$$

$$y = 1, t = -1 \quad \therefore t \neq y$$

update weights

$$\begin{aligned}w_1(\text{new}) &= 0 + 1 * (-1)(-1) \\&= 1\end{aligned}$$

$$\begin{aligned}w_2(\text{new}) &= 2 + 1 * (-1)(1) \\&= 1\end{aligned}$$

$$\begin{aligned}b_{\text{new}} &= 0 + 1 * (-1) \\&= -1\end{aligned}$$

4th i/p

$$x_1 = -1, x_2 = -1, b = 1 \quad t = -1$$

$$\omega_1 = 1, \omega_2 = 1, b = -1$$

$$y_{in} = -1 + 1(-1) + 1(-1) \\ = -3$$

$$y = f(y_{in}) = 0 - 1$$

$$y = -1, t = -1$$

no wt. updation

$$\omega_1 = 1, \omega_2 = 1, b = -1$$

Input			Target (t)	Net input (y_{in})	Calculated output (y)	Weight changes			Weights		
x_1	x_2	1				Δw_1	Δw_2	Δb	w_1 (0)	w_2 (0)	b (0)
EPOCH-1											
1	1	1	1	0	0	1	1	1	1	1	1
1	-1	1	-1	1	1	-1	1	-1	0	2	0
-1	1	1	-1	2	1	+1	-1	-1	1	1	-1
-1	-1	1	-1	-3	-1	0	0	0	1	1	-1
EPOCH-2											
1	1	1	1	1	1	0	0	0	1	1	-1
1	-1	1	-1	-1	-1	0	0	0	1	1	-1
-1	1	1	-1	-1	-1	0	0	0	1	1	-1
-1	-1	1	-1	-3	-1	0	0	0	1	1	-1

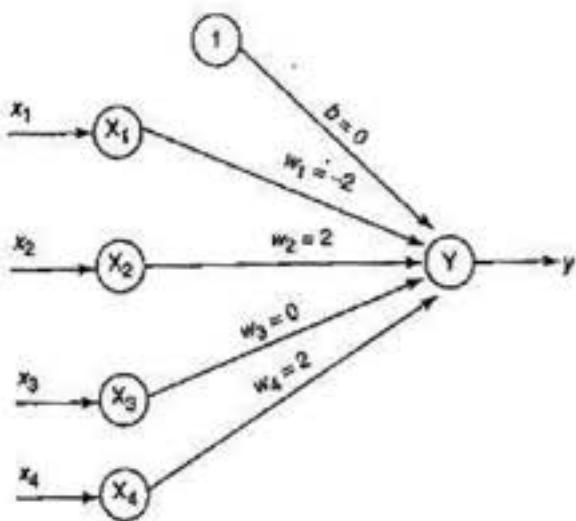
Find the weights required to perform the following classification using perceptron network. The vectors $(1, 1, 1, 1)$ and $(-1, 1, -1, -1)$ are belonging to the class (so have target value 1), vectors $(1, 1, 1, -1)$ and $(1, -1, -1, 1)$ are not belonging to the class (so have target value -1). Assume learning rate as 1 and initial weights as 0.

Input

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

$$y = \begin{cases} 1 & \text{if } y_{in} > 0.2 \\ 0 & \text{if } -0.2 \leq y_{in} \leq 0.2 \\ -1 & \text{if } y_{in} < -0.2 \end{cases}$$

Inputs					Target	Net input (y_{in})	Output (y)	Weight changes					Weights					
$(x_1$	x_2	x_3	x_4	$b)$	(t)			$(\Delta w_1$	Δw_2	Δw_3	Δw_4	$\Delta b)$	$(w_1$	w_2	w_3	w_4	$b)$	
EPOCH-1																		
(1	1	1	1	1)	1	0	0	1	1	1	1	1	1	1	1	1	1)	
(-1	1	-1	-1	1)	1	-1	-1	-1	1	-1	-1	1	0	2	0	0	2)	
(1	1	1	-1	1)	-1	4	1	-1	-1	-1	-1	1	-1	1	-1	1	1)	
(1	-1	-1	1	1)	-1	1	1	-1	1	1	-1	-1	-2	2	0	0	0)	
EPOCH-2																		
(1	1	1	1	1)	1	0	0	1	1	1	1	1	-1	3	1	1	1)	
(-1	1	-1	-1	1)	1	3	1	0	0	0	0	0	-1	3	1	1	1)	
(1	1	1	-1	1)	-1	4	1	-1	-1	-1	1	-1	-2	2	0	2	0)	
(1	-1	-1	1	1)	-1	-2	-1	0	0	0	0	0	-2	2	0	2	0)	
EPOCH-3																		
(1	1	1	1	1)	1	2	1	0	0	0	0	0	-2	2	0	2	0)	
(-1	1	-1	-1	1)	1	2	1	0	0	0	0	0	-2	2	0	2	0)	
(1	1	1	-1	1)	-1	-2	-1	0	0	0	0	0	-2	2	0	2	0)	
(1	-1	-1	1	1)	-1	-2	-1	0	0	0	0	0	-2	2	0	2	0)	



Thank you !!!!!



Machine Learning (19CSE305)

Error Surface, Parameter Optimization & SVM

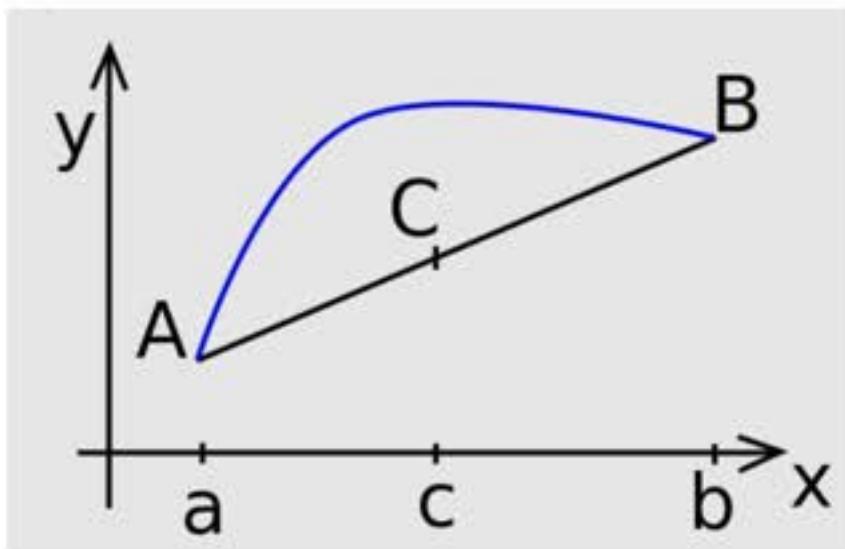
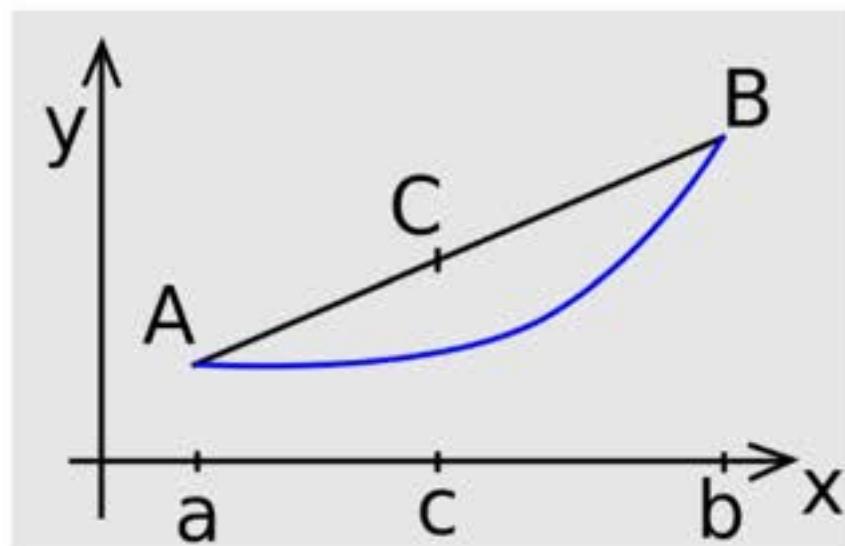
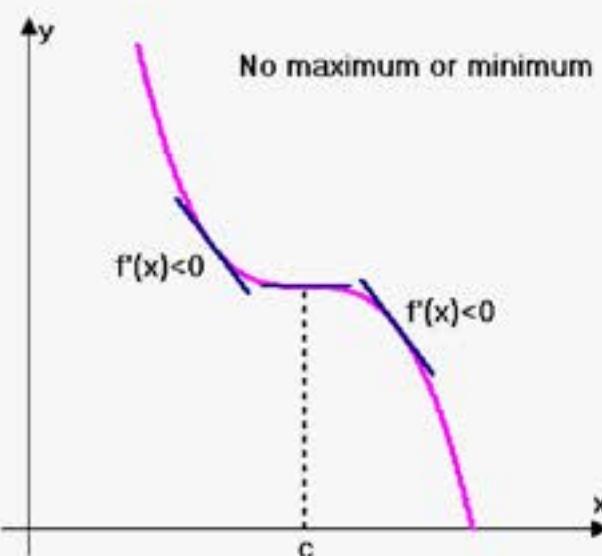
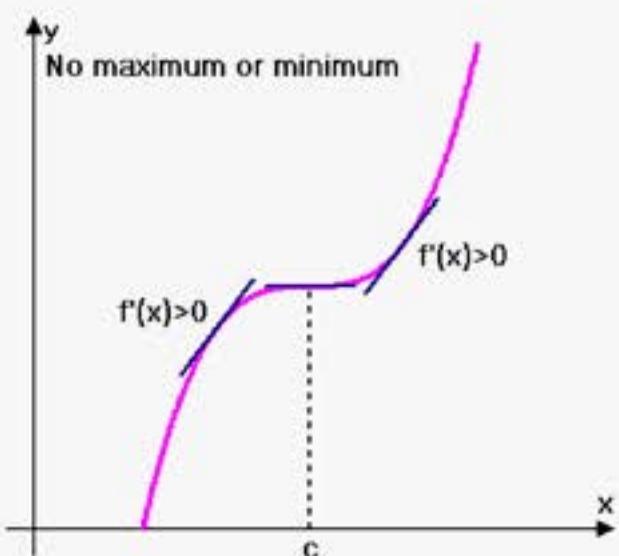
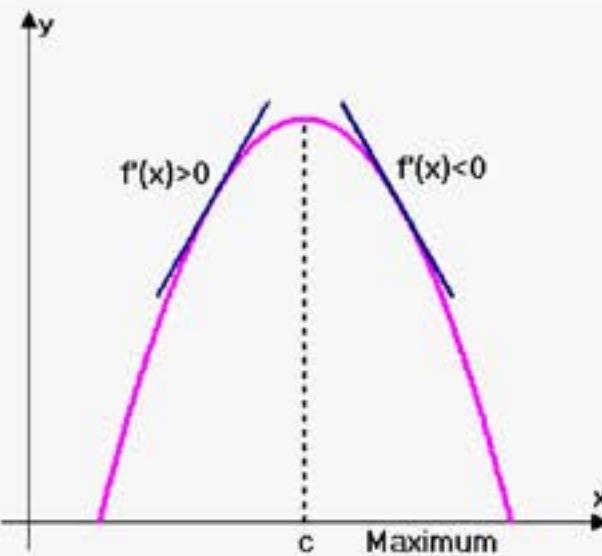
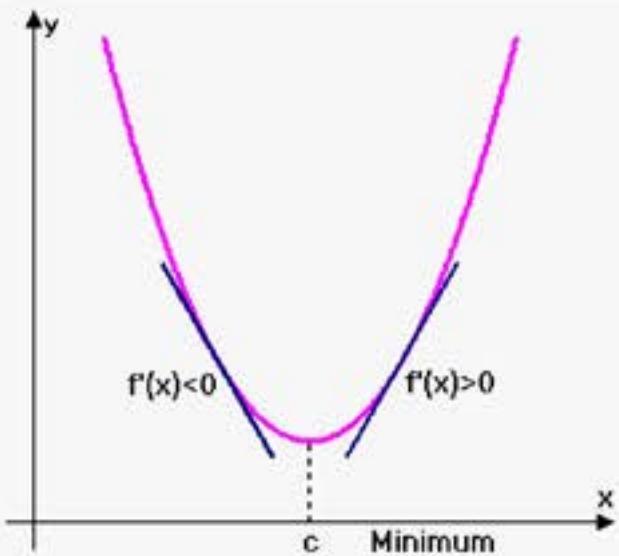


Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

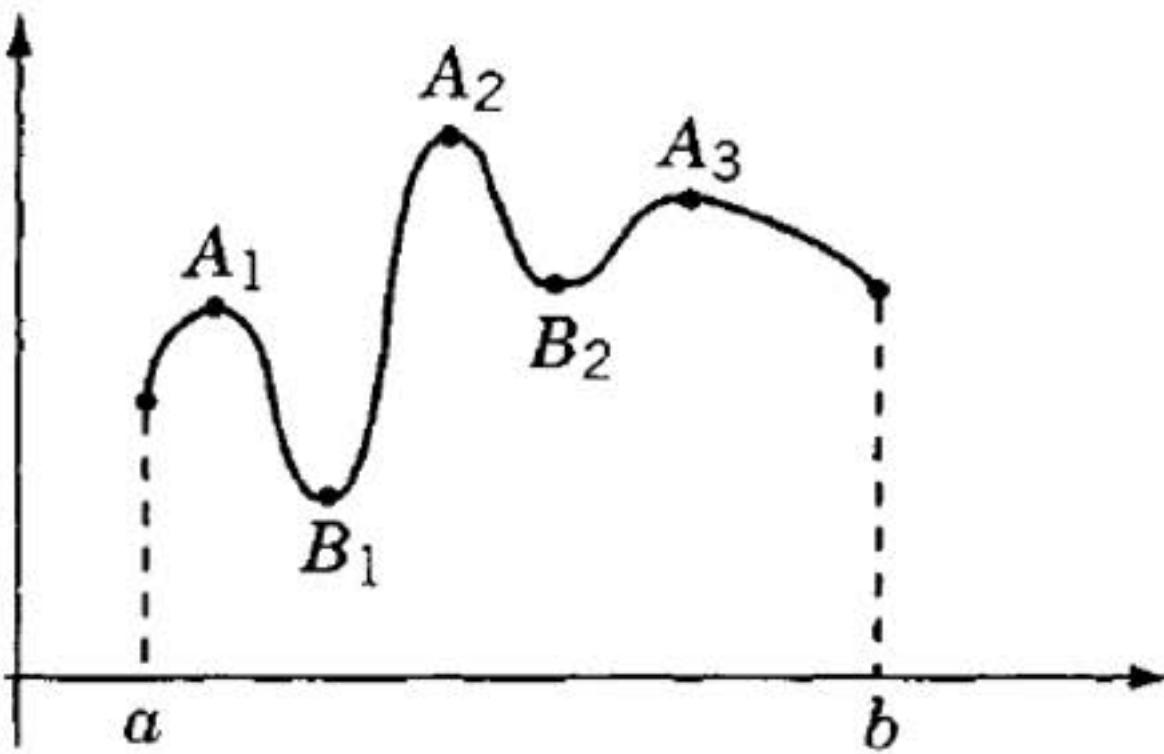
- Recap of Optimization
- Support Vectors

Functions, Derivatives & Convexity



Source: Internet

Local & Global – Minimum, Maximum; Saddle point



- Brute force search
- Gradient Descent Search
- Genetic algorithm based approaches
- Evolutionary computation techniques
- Tabu Search
- Simulated Annealing
- Hill Climbing techniques
- Ant colony optimization
- Particle swarm optimization
- Random forest optimization

Source: Engineering Optimization, S S Rao

Derivation of Gradient Descent Algorithm

$$o(\vec{x}) = \vec{w} \cdot \vec{x}$$

$$E(\vec{w}) \equiv \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$$

$$\vec{w} \leftarrow \vec{w} + \Delta \vec{w}$$

$$\Delta \vec{w} = -\eta \nabla E(\vec{w})$$

$$\nabla E(\vec{w}) \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \dots, \frac{\partial E}{\partial w_n} \right]$$

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

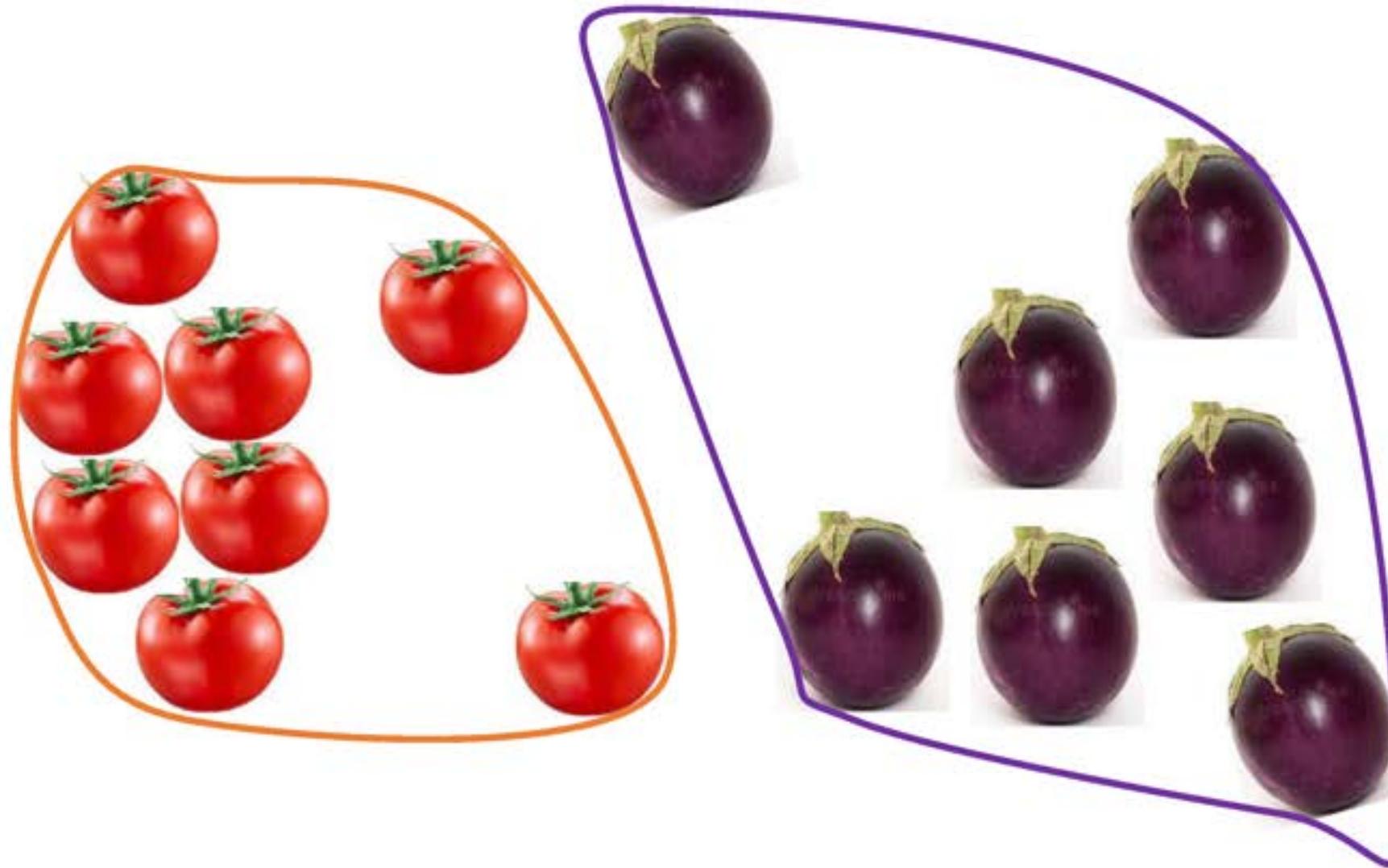
$$\begin{aligned}\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i} \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2 \\&= \frac{1}{2} \sum_{d \in D} \frac{\partial}{\partial w_i} (t_d - o_d)^2 \\&= \frac{1}{2} \sum_{d \in D} 2(t_d - o_d) \frac{\partial}{\partial w_i} (t_d - o_d) \\&= \sum_{d \in D} (t_d - o_d) \frac{\partial}{\partial w_i} (t_d - \vec{w} \cdot \vec{x}_d) \\ \frac{\partial E}{\partial w_i} &= \sum_{d \in D} (t_d - o_d) (-x_{id}) \\ \Delta w_i &= \eta \sum_{d \in D} (t_d - o_d) x_{id}\end{aligned}$$

Notes on Gradient Descent Algorithm

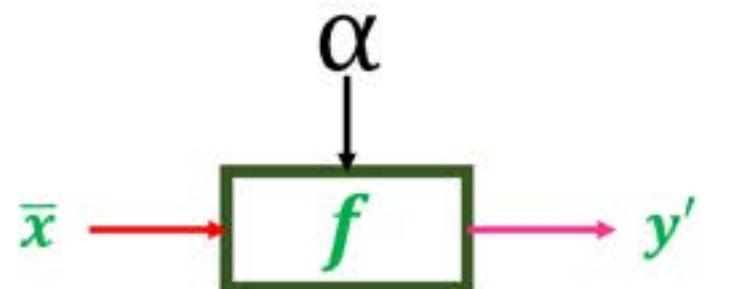
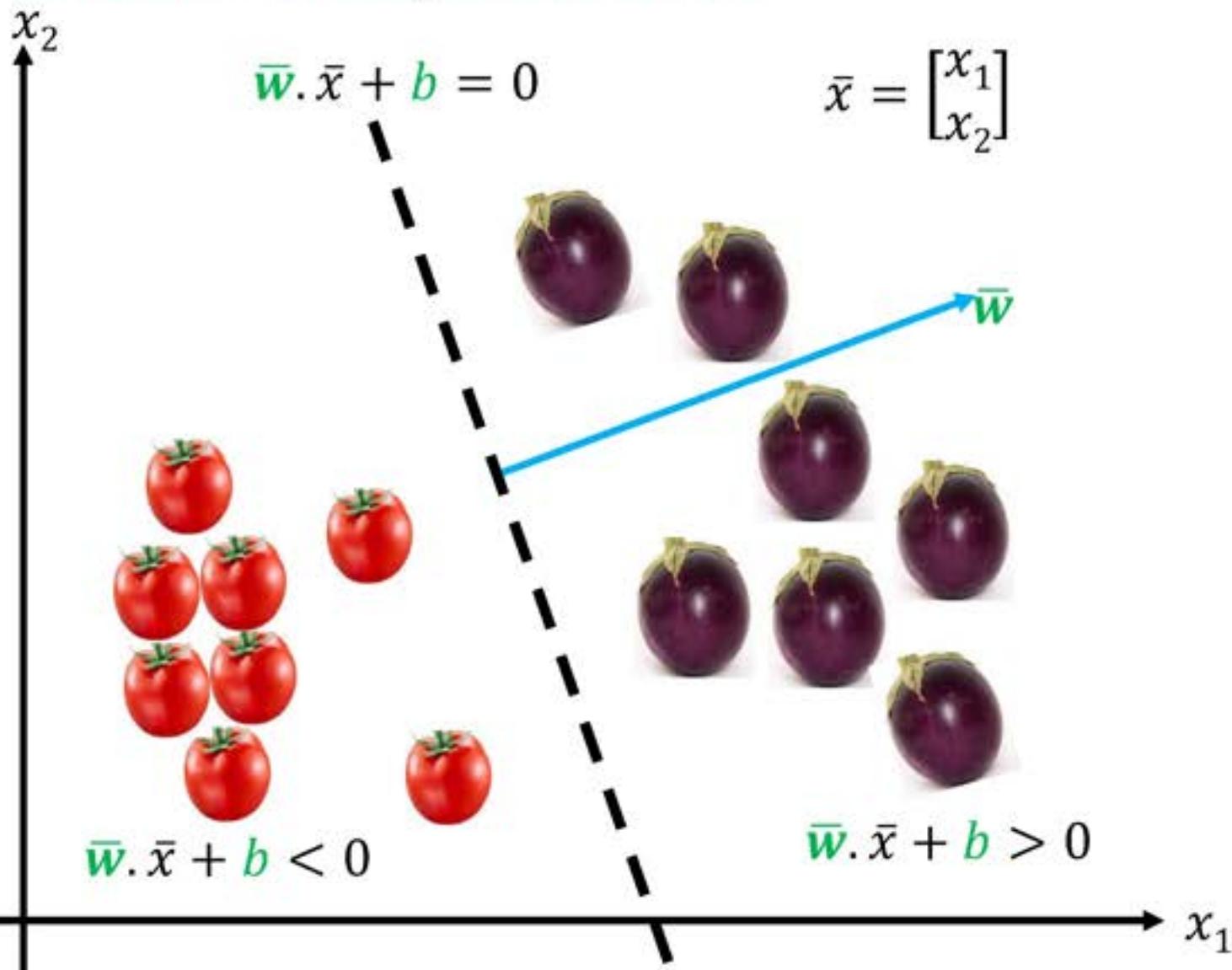
- Error is summed over all inputs and then the weights are updated
- Linear (pass through) activation function is used $\rightarrow f(x) = x$
- Assumes a convex error space
- If there are local minima, the search may get stuck and never come-out
- Since error is summed over all inputs, the convergence may be slow
- Incremental or stochastic gradient descent is a variation of the algorithm
 - Weight update done with each input
 - Sometimes helps in overcoming the local minima
- The same principle can be applied with other activation functions as well. However, mathematical proof of convergence may be difficult.

Support Vector Machines

Classes & Boundaries



Linear binary classifier



$$y' = f(\bar{x}, \bar{w}, b) \\ = \text{sign}(\bar{w} \cdot \bar{x} + b)$$

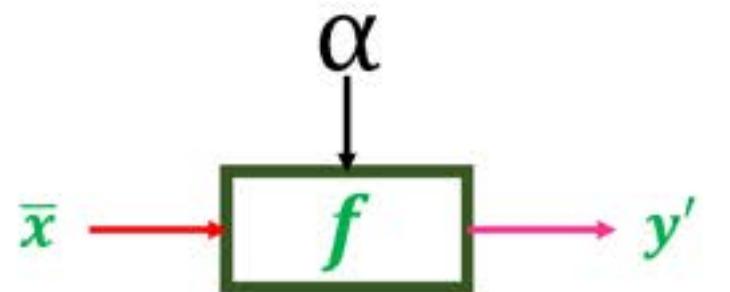
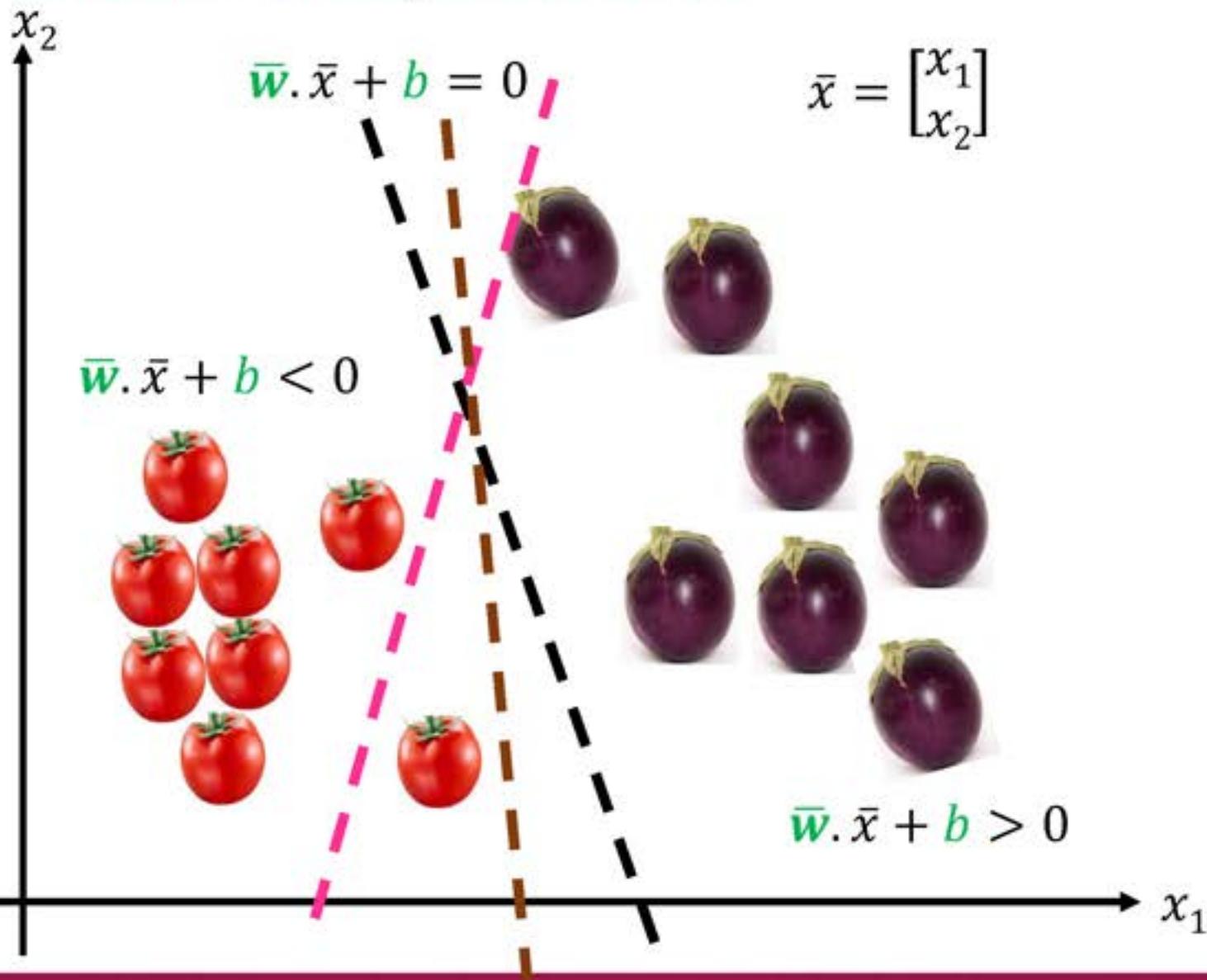


Denotes +ve class



Denotes -ve class

Linear binary classifier



$$y' = f(\bar{x}, \bar{w}, b)$$
$$= \text{sign}(\bar{w} \cdot \bar{x} + b)$$

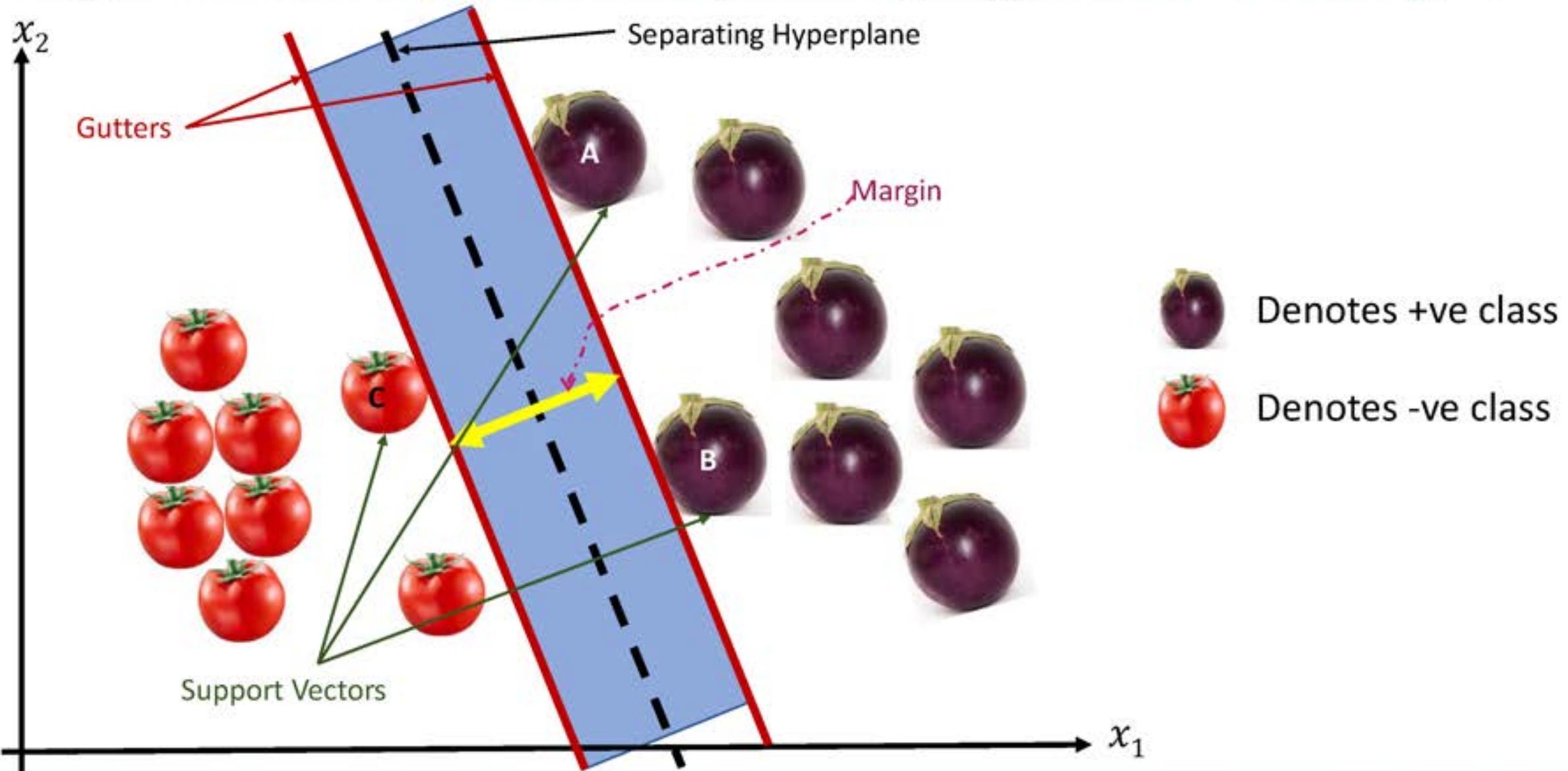


Denotes +ve class

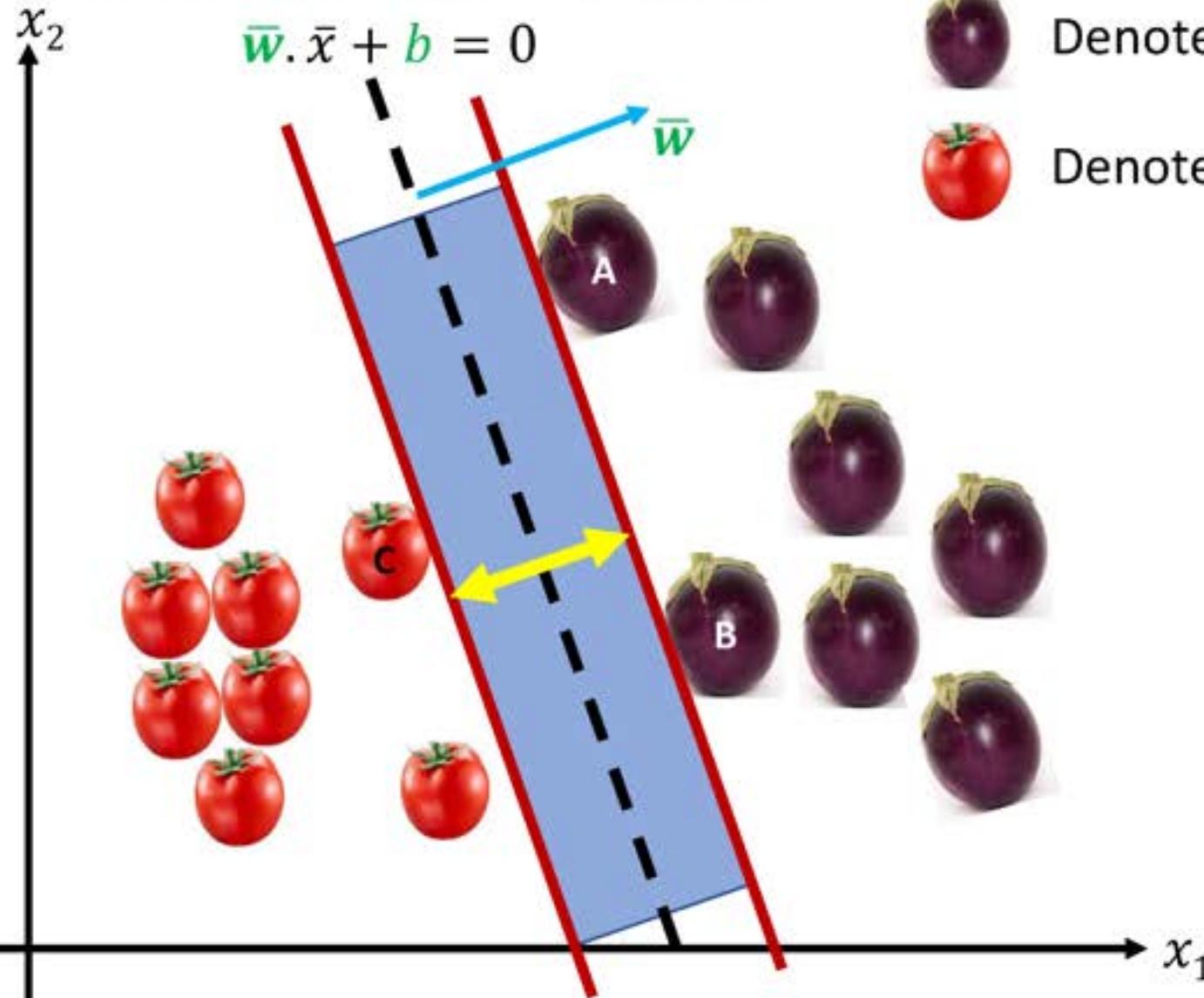


Denotes -ve class

Support Vectors, Gutters, Separating Hyperplane & Margin



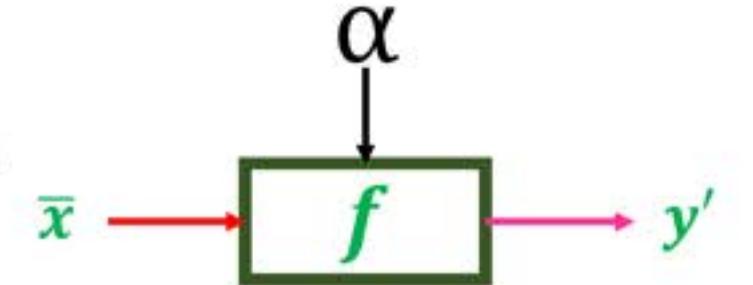
Linear binary classifier



Denotes +ve class



Denotes -ve class



$$\begin{aligned}y' &= f(\bar{x}, \bar{w}, b) \\&= \text{sign}(\bar{w} \cdot \bar{x} + b)\end{aligned}$$

$$\bar{w} \cdot \bar{x}_A + b = +1$$

$$\bar{w} \cdot \bar{x}_B + b = +1$$

$$\bar{w} \cdot \bar{x}_C + b = -1$$

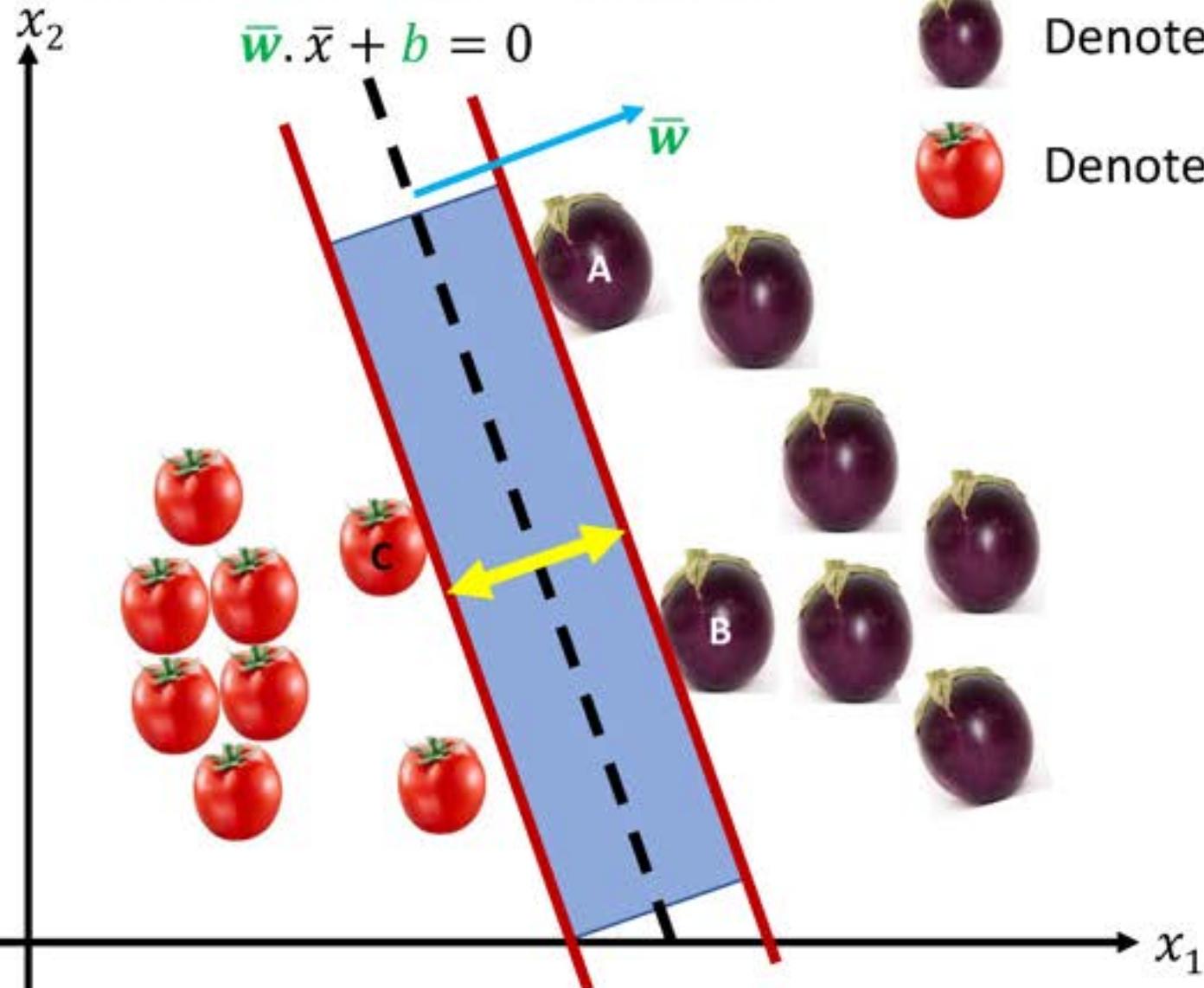
Generalization

$$\bar{w} \cdot \bar{x}_{+ve} + b \geq +1$$

$$\bar{w} \cdot \bar{x}_{-ve} + b \leq -1$$

1

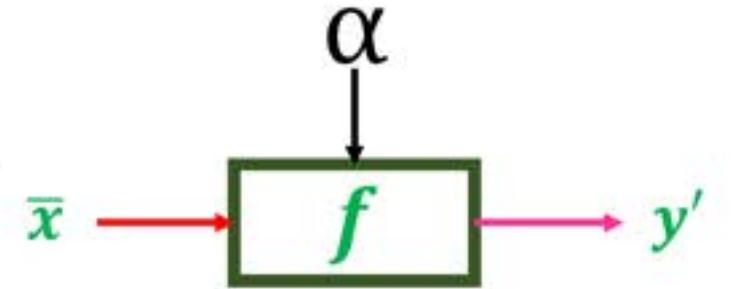
Linear binary classifier



Denotes +ve class



Denotes -ve class



$$\begin{aligned}y' &= f(\bar{x}, \bar{w}, b) \\&= \text{sign}(\bar{w} \cdot \bar{x} + b)\end{aligned}$$

$$\bar{w} \cdot \bar{x}_A + b = +1$$

$$\bar{w} \cdot \bar{x}_B + b = +1$$

$$\bar{w} \cdot \bar{x}_C + b = -1$$

$$M \cdot \|\bar{w}\| = (\bar{w} \cdot \bar{x}_A + b) - (\bar{w} \cdot \bar{x}_C + b)$$

$$M = \frac{2}{\|\bar{w}\|}$$

2

Optimization for SVM

1

$$\bar{w} \cdot \bar{x}_{+ve} + b \geq +1$$

$$\bar{w} \cdot \bar{x}_{-ve} + b \leq -1$$

Multiply each equation with y_i

$y_i = +1$ for all +ve class vectors

$y_i = -1$ for all -ve class vectors

$$(\bar{w} \cdot \bar{x}_i + b)y_i \geq +1$$

2

$$M \cdot ||\bar{w}|| = (\bar{w} \cdot \bar{x}_A + b) - (\bar{w} \cdot \bar{x}_C + b)$$

$$M = \frac{2}{||\bar{w}||}$$

We want to maximize the margin M .

→ Minimize $||\bar{w}||$

→ Minimize $(\bar{w}^T \cdot \bar{w})$

Formulate the optimization problem & constraints

$$\phi(\bar{w}) = \frac{1}{2} (\bar{w}^T \cdot \bar{w})$$

subject to $(\bar{w} \cdot \bar{x}_i + b)y_i \geq +1$
for all i

$$\begin{aligned}y' &= f(\bar{x}) \\&= \sum_i \alpha_i y_i \bar{x}_i^T \cdot \bar{x} + b\end{aligned}$$

Optimization for SVM

Formulate the optimization problem & constraints

$$\phi(\bar{w}) = \frac{1}{2} (\bar{w}^T \cdot \bar{w})$$

subject to $(\bar{w} \cdot \bar{x}_i + b) y_i \geq +1$
for all i

$$\sum_i \alpha_i y_i = 0$$

y_i are all scalars; hence,
 α_i are also scalars.


$$\begin{aligned}y' &= f(\bar{x}) \\&= \sum_i \alpha_i y_i \bar{x}_i^T \cdot \bar{x} + b\end{aligned}$$

SVM

$$\begin{aligned}y' &= f(\bar{x}) \\&= \sum_i \alpha_i y_i \boxed{\bar{x}_i^T \cdot \bar{x}} + b\end{aligned}$$

Kernel function is some function which maintains the sanctity of dot product in some expanded space

Kernel function:

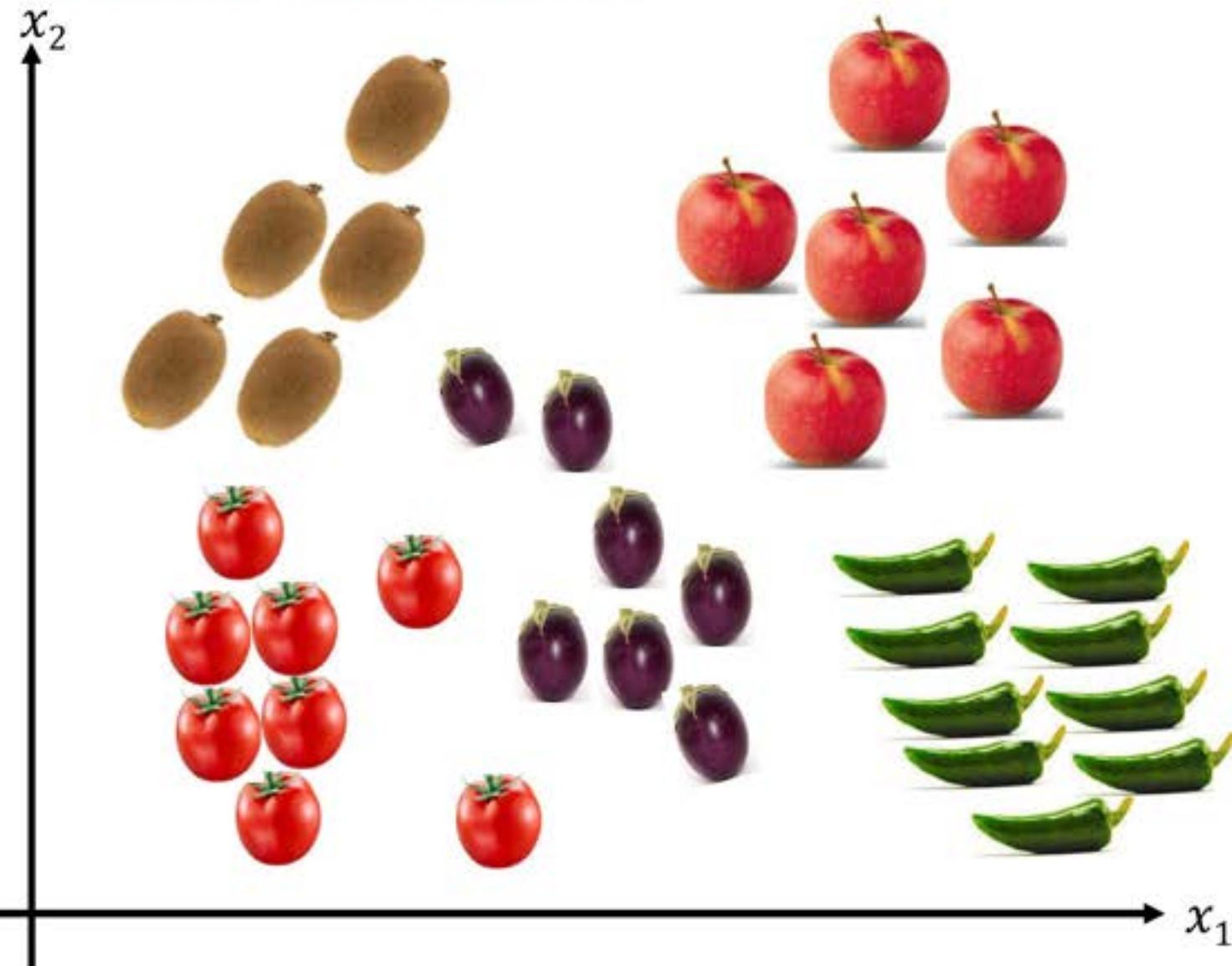
$$K(\bar{x}_i, \bar{x}_j) = \bar{x}_i^T \cdot \bar{x}_j$$

Kernel function generalized as:

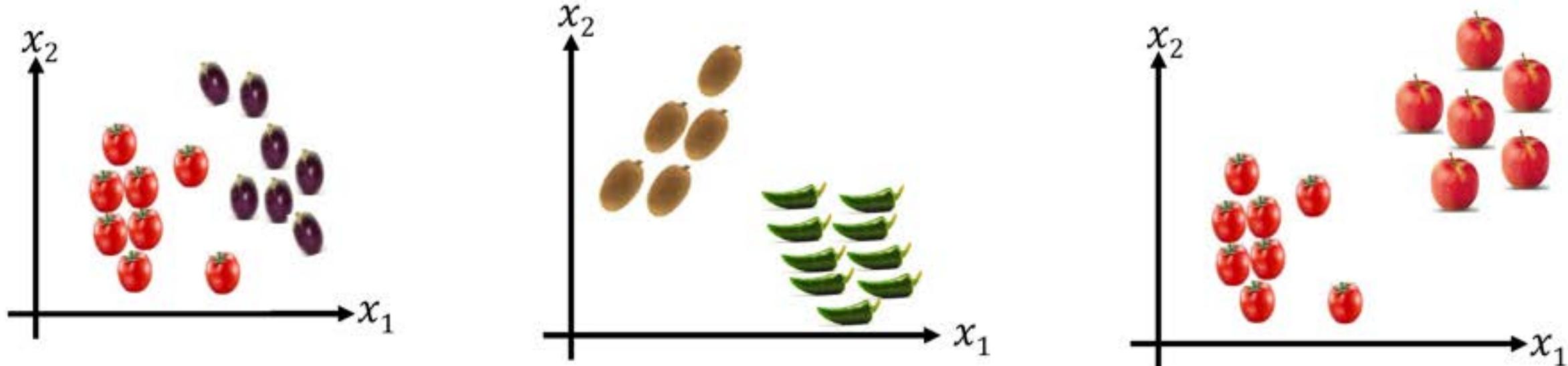
$$K(\bar{x}_i, \bar{x}_j) = \varphi(\bar{x}_i)^T \cdot \varphi(\bar{x}_j)$$

Kernel function by mapping the vectors to some expanded space, introduces linearity which is absent in the original space.

Multiclass Scenario

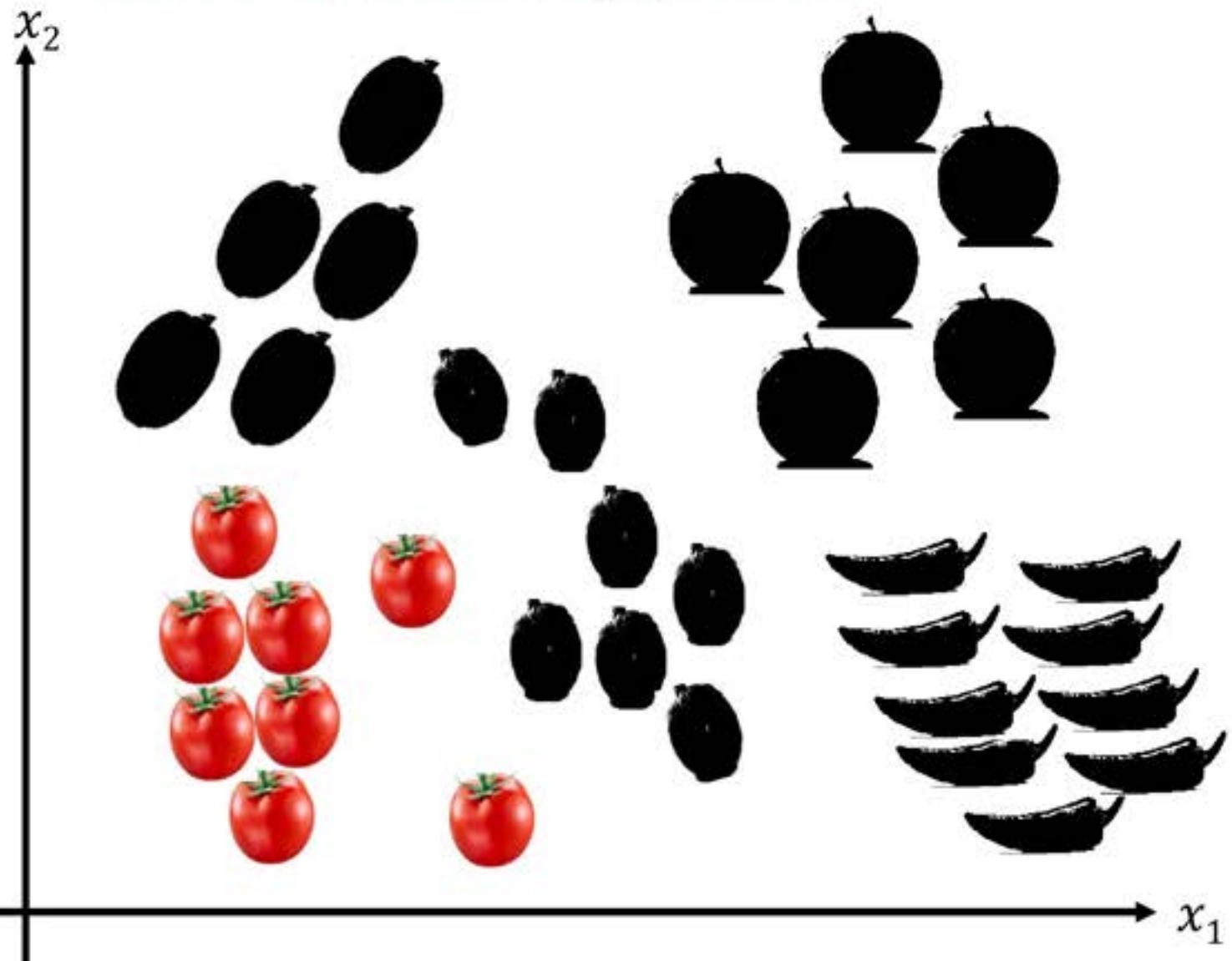


One against another approach



- Consider each class separately against all other classes combined
 - ✓ Tomato against brinjal
 - ✓ Kiwi against green chilies
 - ✓ Apples against tomatoes
- $N * (N-1) / 2$ SVM models generated (10 models in this example)
- Test pattern is run through each models
- Class value assigned based on absolute maximum score & sign (+/-ve) from all models

One vs all others approach



- Consider each class separately against all other classes combined
 - ✓ Tomato against all others
 - ✓ Kiwi against all others
 - ✓ Apples against all others
- SVM models generated as many classes are available (5 models in this example)
- Test pattern is run through each models
 - ✓ Score of being an apple
 - ✓ Score of being a tomato
- Class value assigned based on maximum membership score

Python Code

```
>>> from sklearn import svm  
>>> X = [[0, 0], [1, 1]]  
>>> y = [0, 1]  
>>> clf = svm.SVC()  
>>> clf.fit(X, y)
```

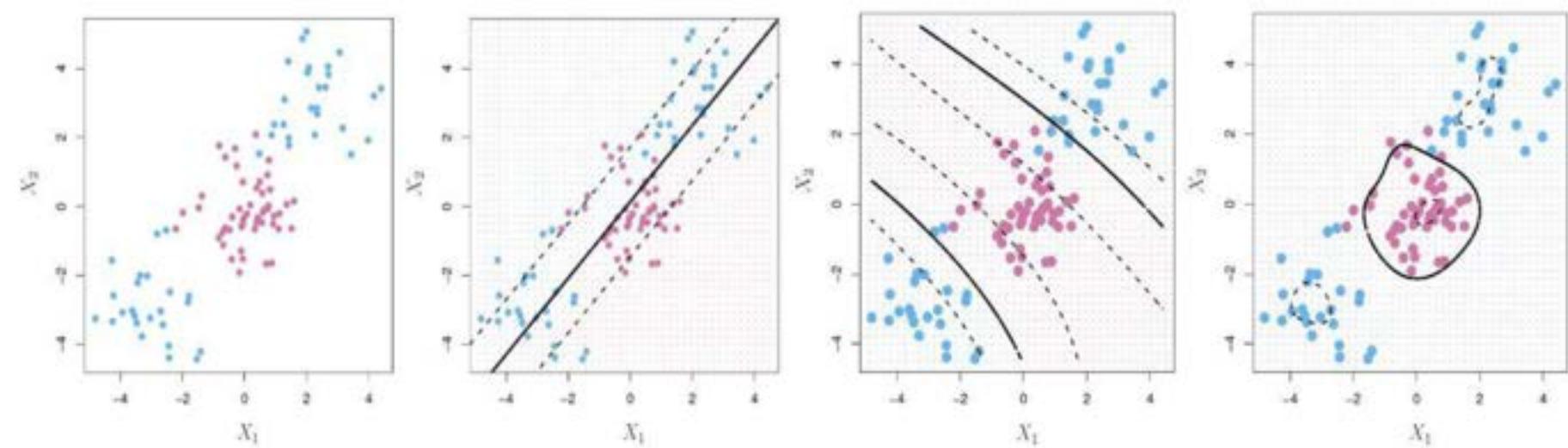
```
>>> clf.predict([[2., 2.]])  
array([1])
```

Thank you !!!!!



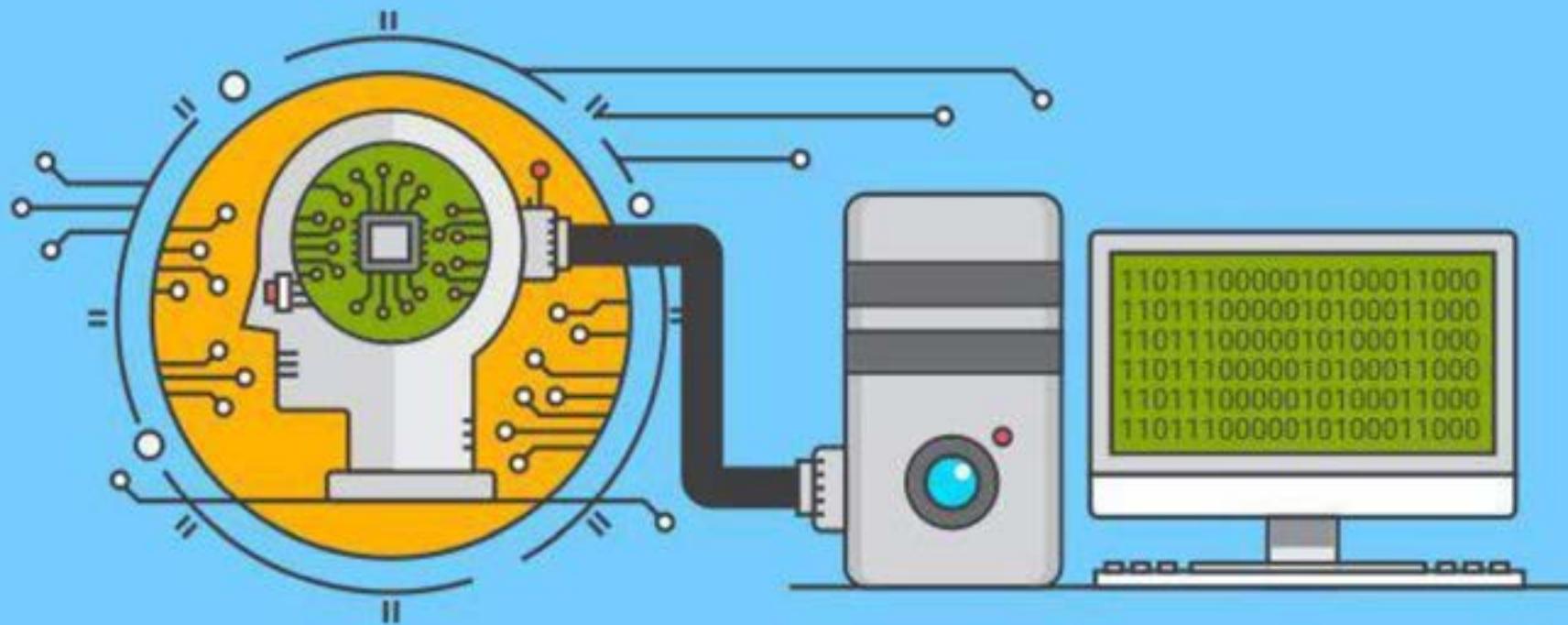
SUPPORT VECTOR MACHINES

A brief introduction



What is machine learning?

Machine learning is the concept that a computer program can learn and adapt to new data with minimal human intervention.



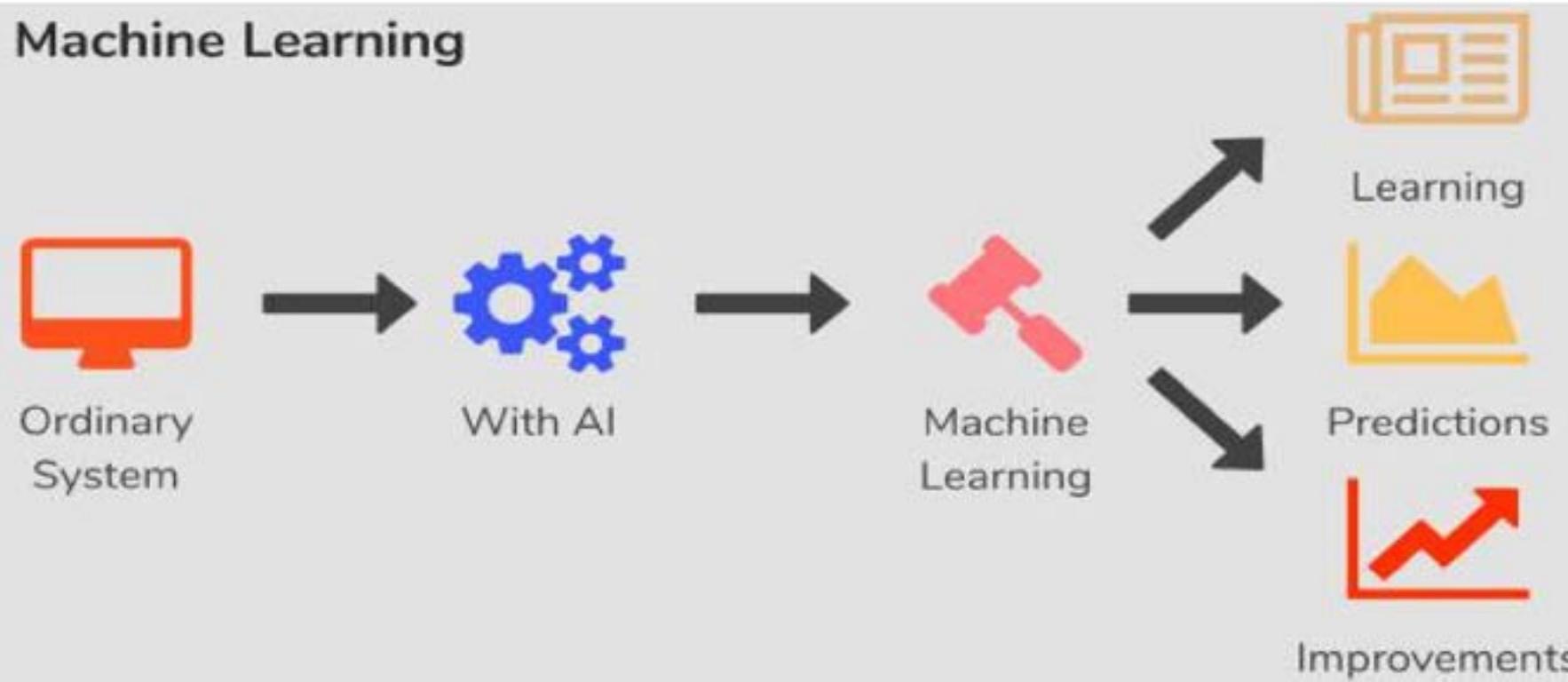
Why machine learning is important?

Data is the lifeblood of all business. Data-driven decisions increasingly make the difference between keeping up with competition or falling further behind.

Use cases:

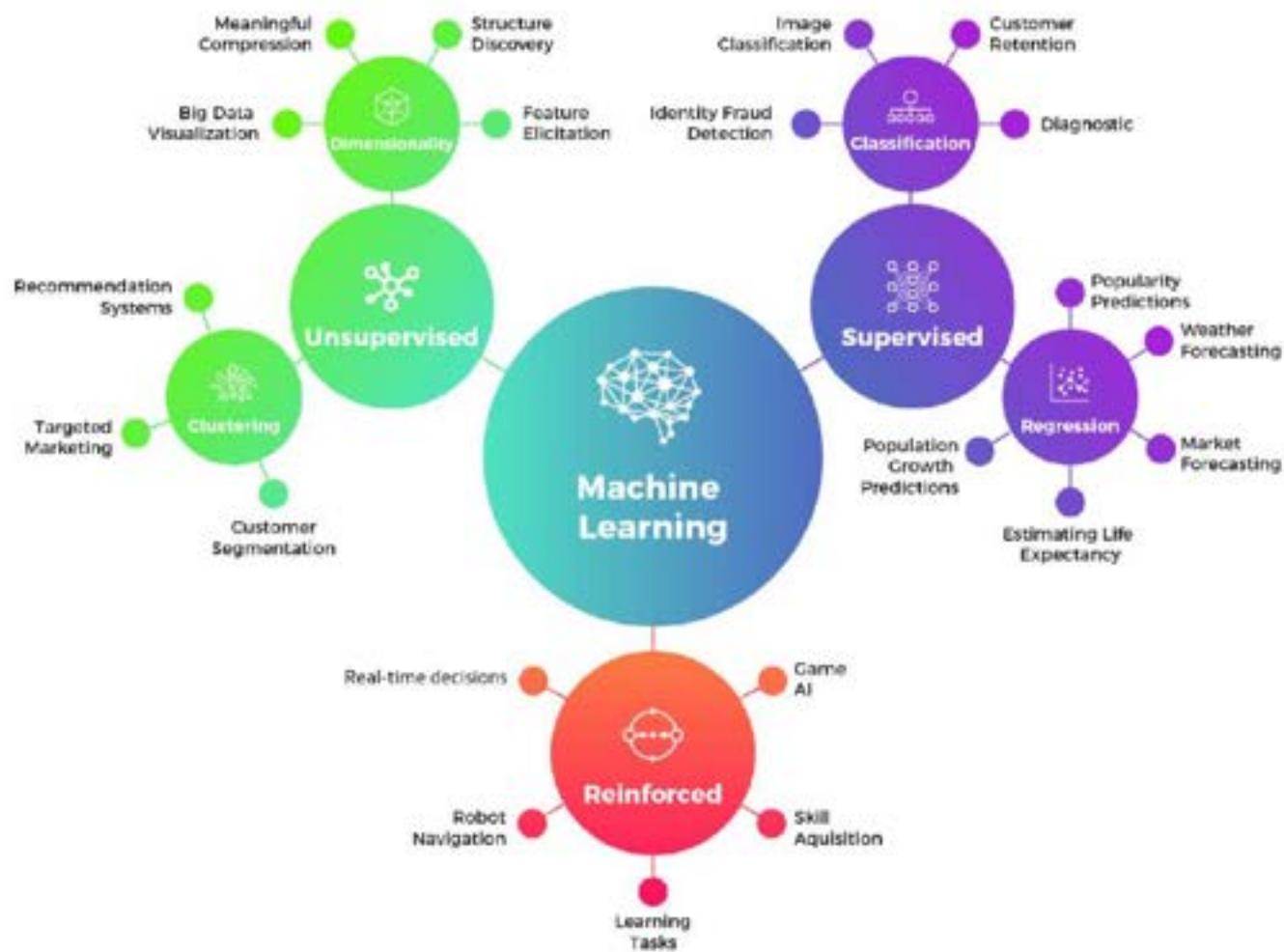
- Manufacturing.
- Retail.
- Healthcare and life sciences.
- Travel and hospitality.
- Financial services.
- Energy.

Machine Learning



What is a machine learning model?

A machine learning model is a file that has been trained to recognize certain types of patterns. You train a model over a set of data, providing it an algorithm that it can use to reason over and learn from those data.



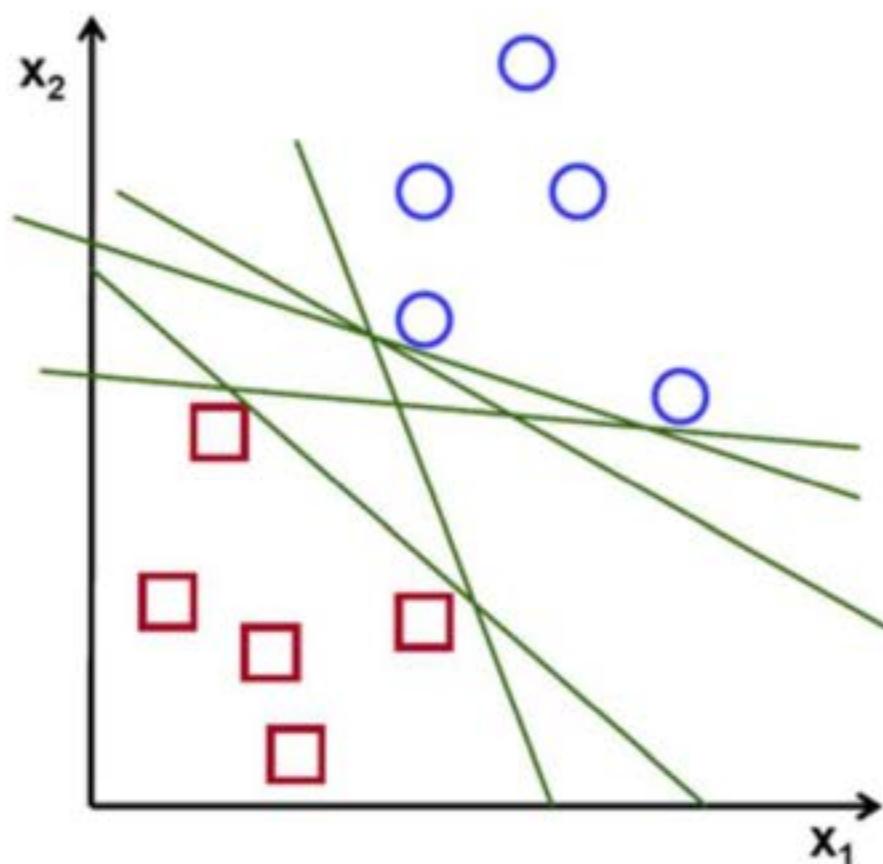
What do we do after choosing a model?



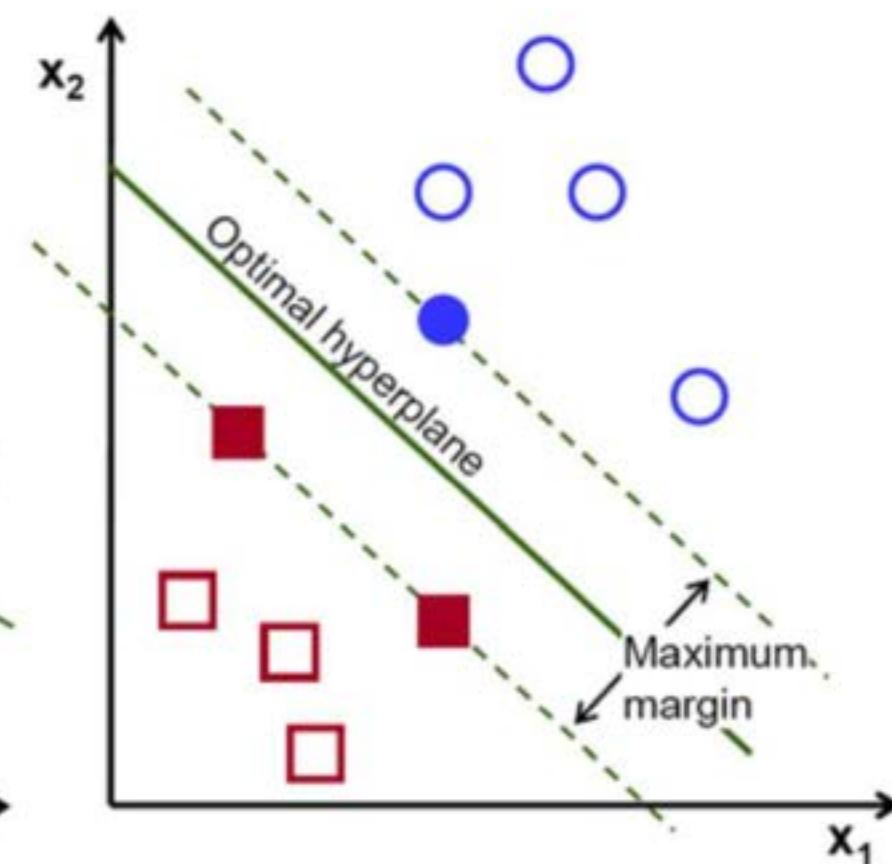
How do we choose which model works best for the data we have?



Support Vector Machine (SVM)

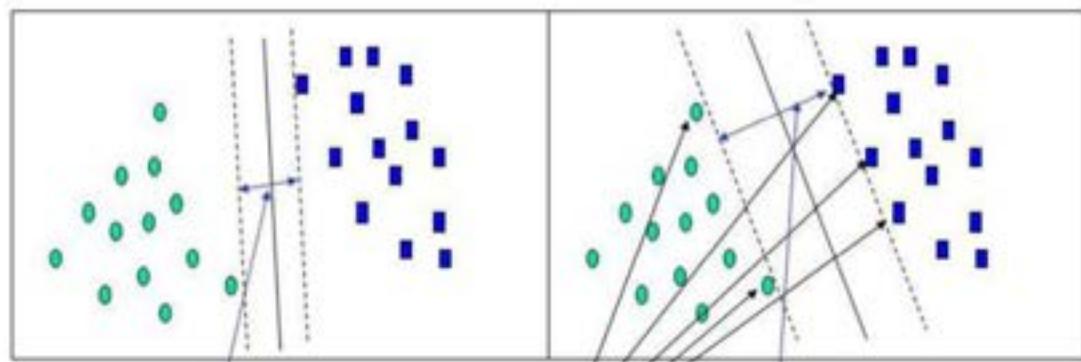
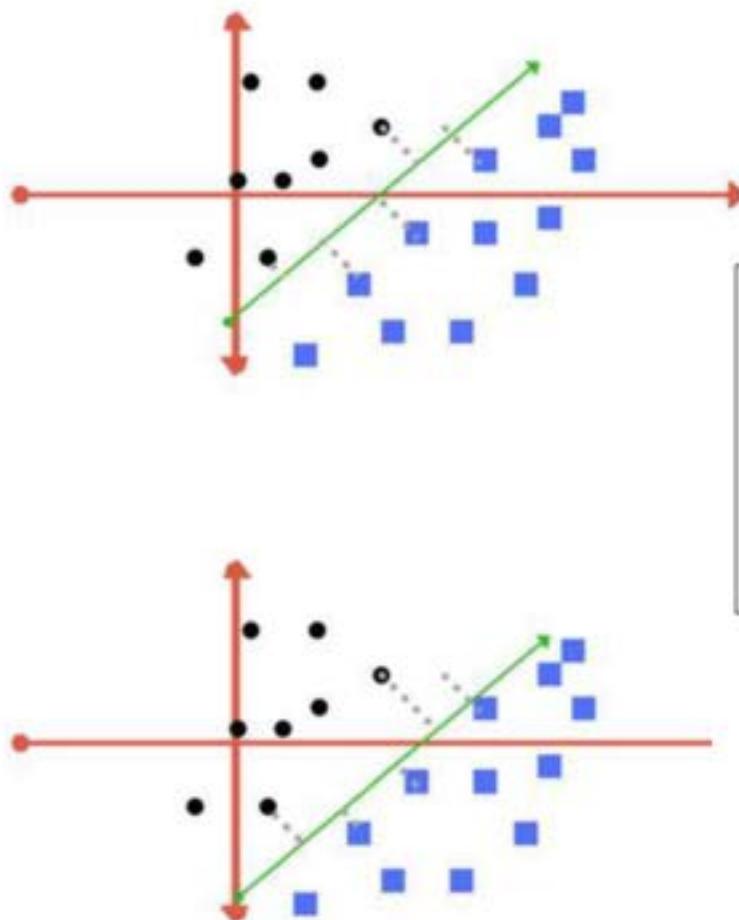


Few possible hyperplanes



Optimal hyperplane

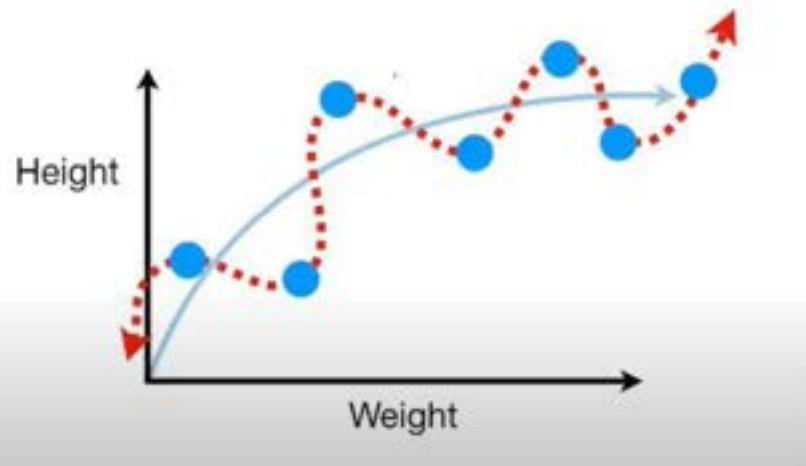
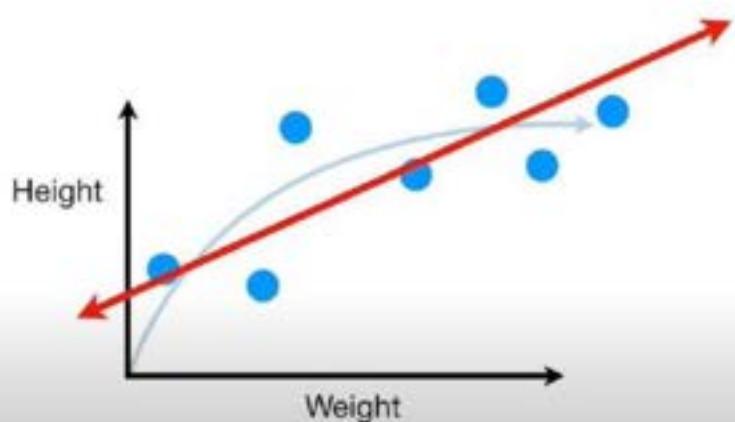
Margins



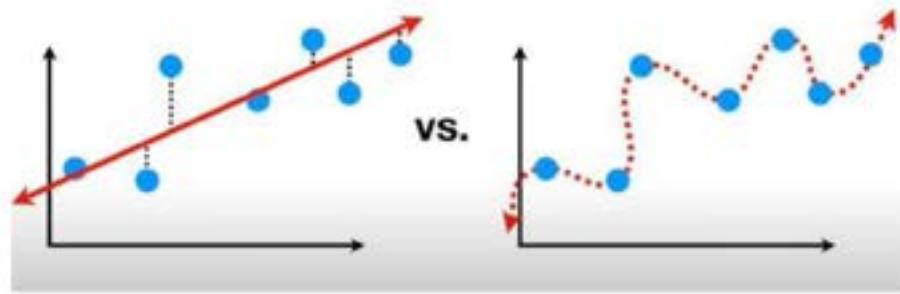
Support Vectors

What are support vectors?

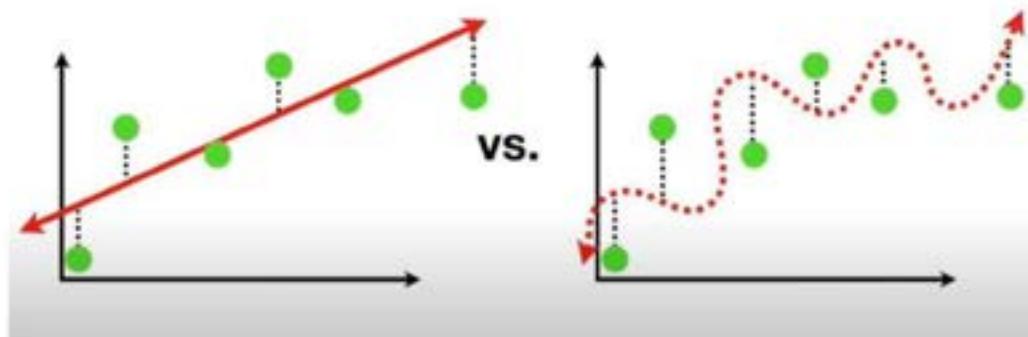
Regularization



Training



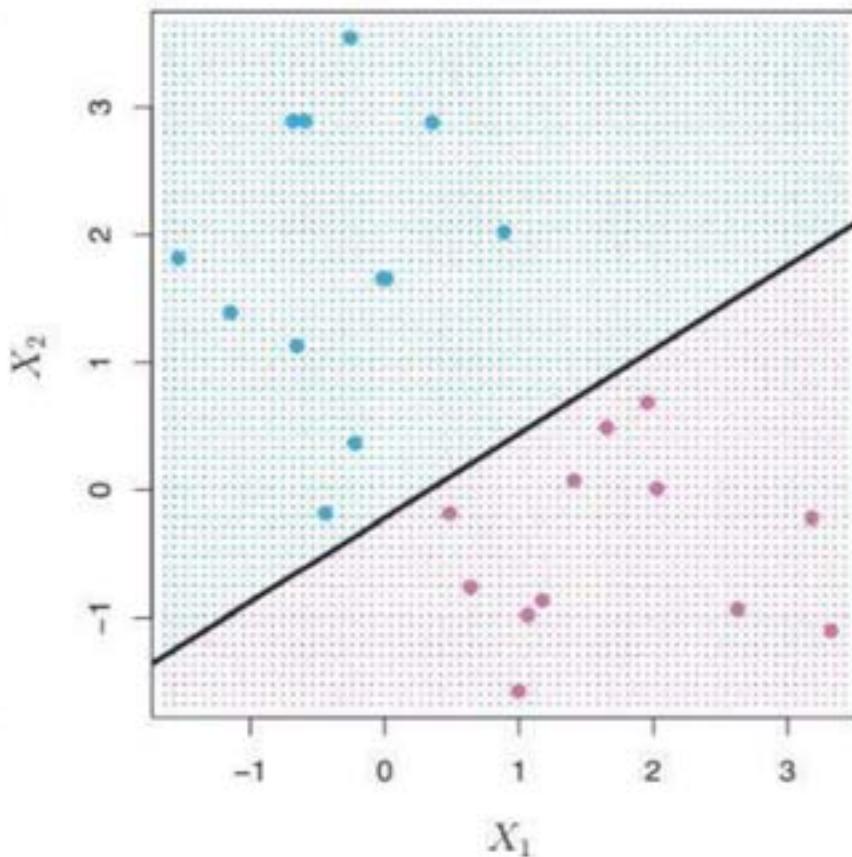
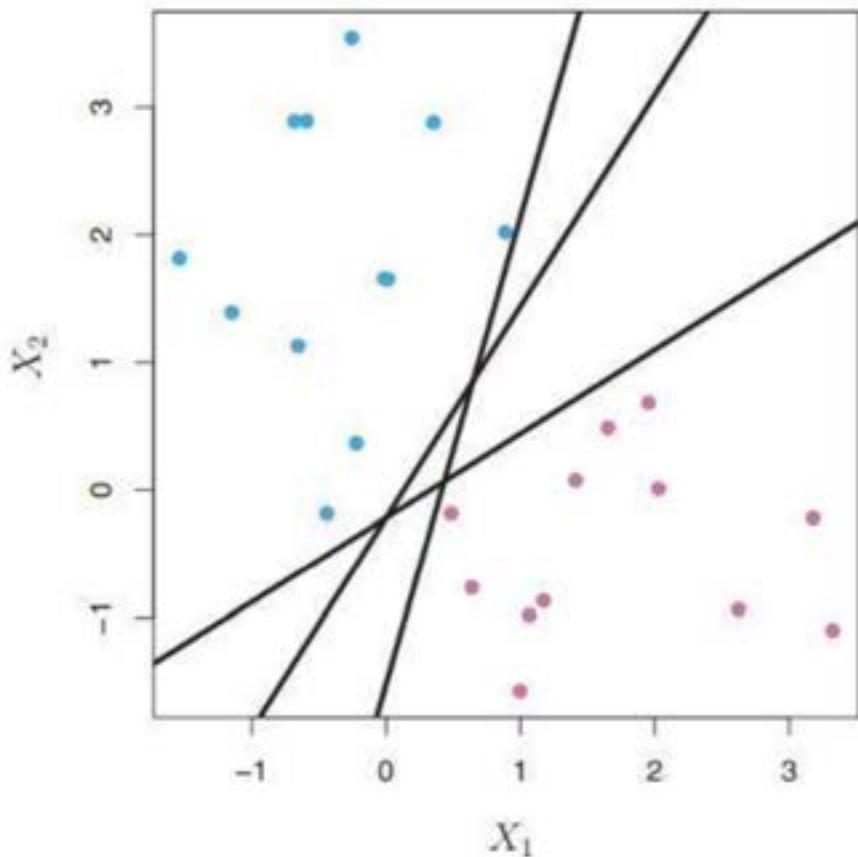
Testing



Maximal margin classifier

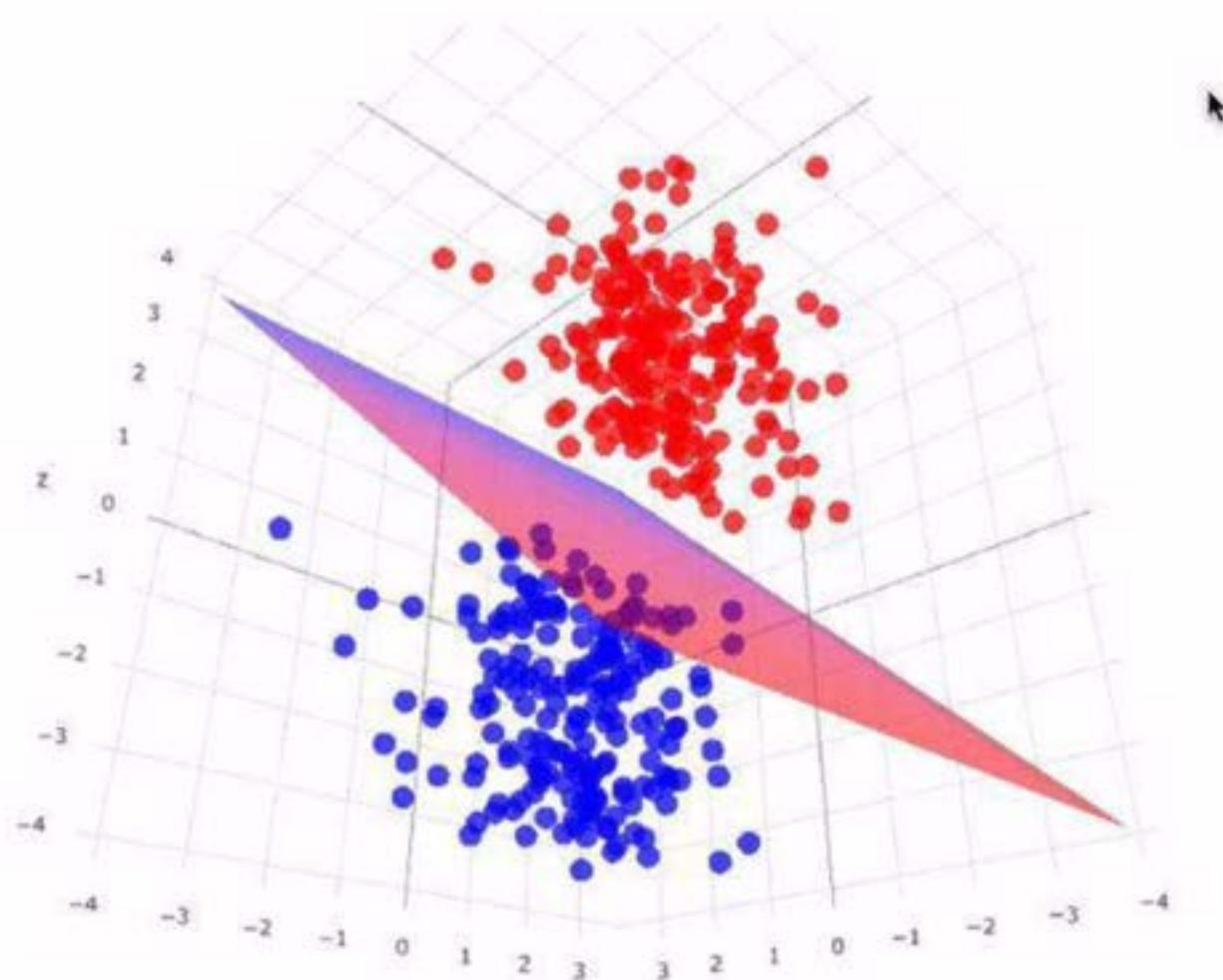
What is a hyperplane?

2-D

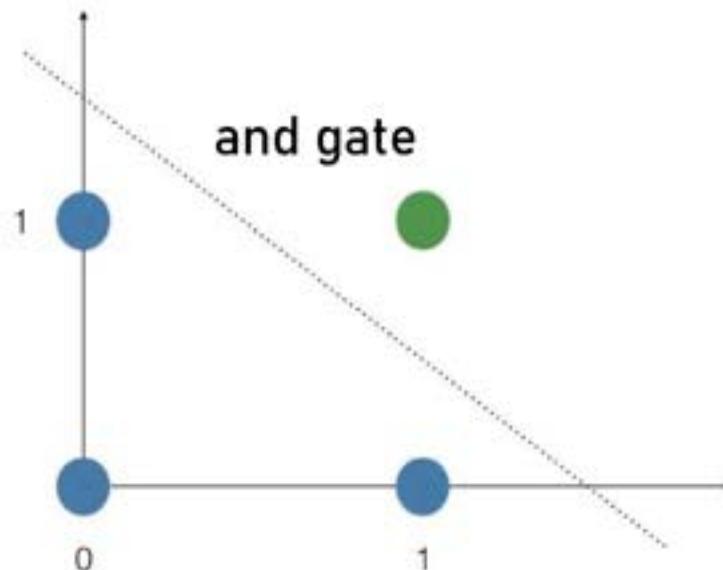


Why is it called a hyperplane?

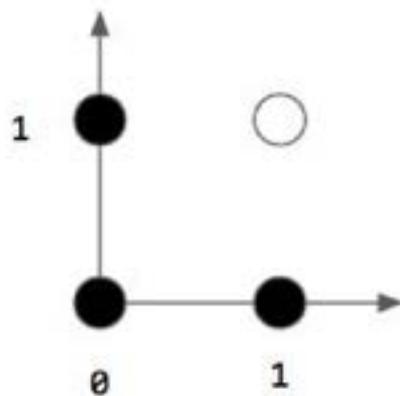
Hyperplane in 3-D



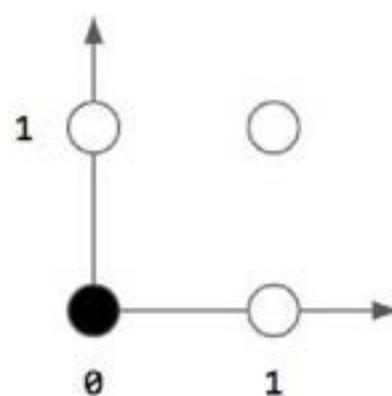
Logic gates



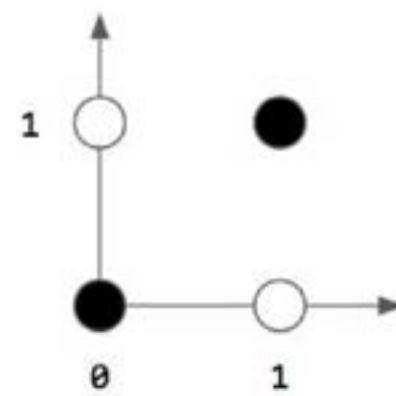
AND



OR

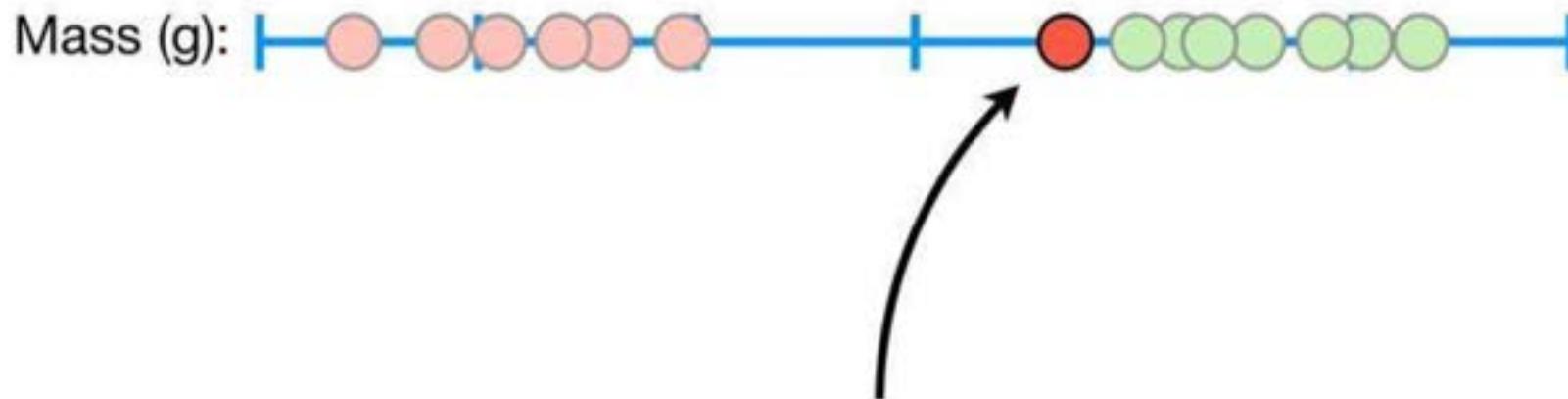


XOR



Soft Margin

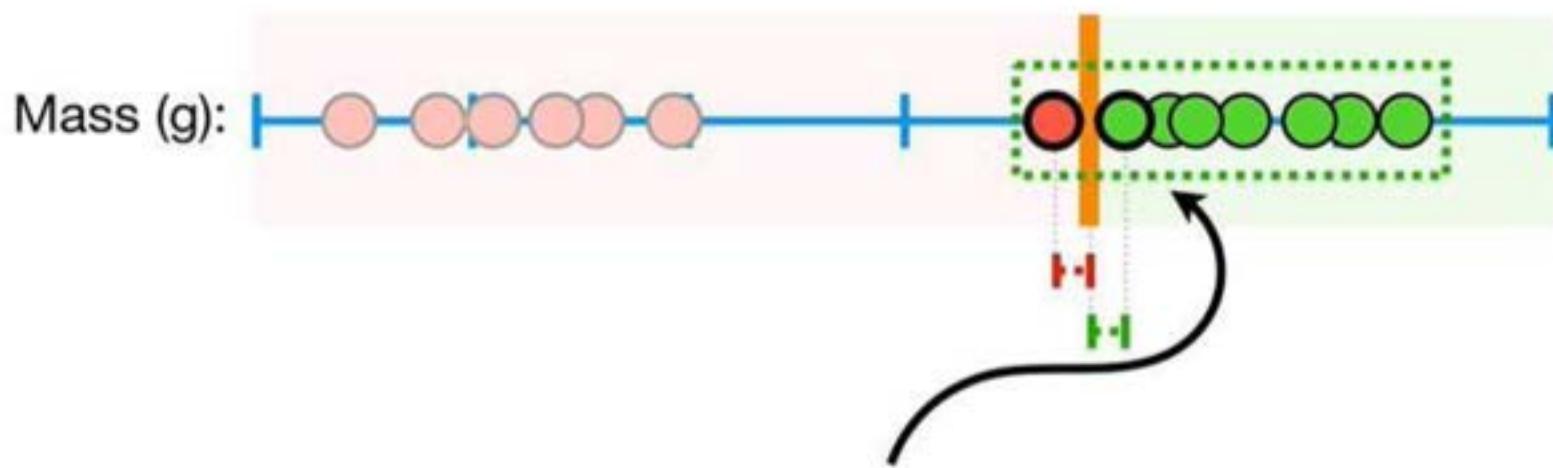
Problems with Maximal margin classifier



...and we had an outlier
observation that was classified as
not obese, but was much closer
to the ***obese*** observations.

Soft Margin

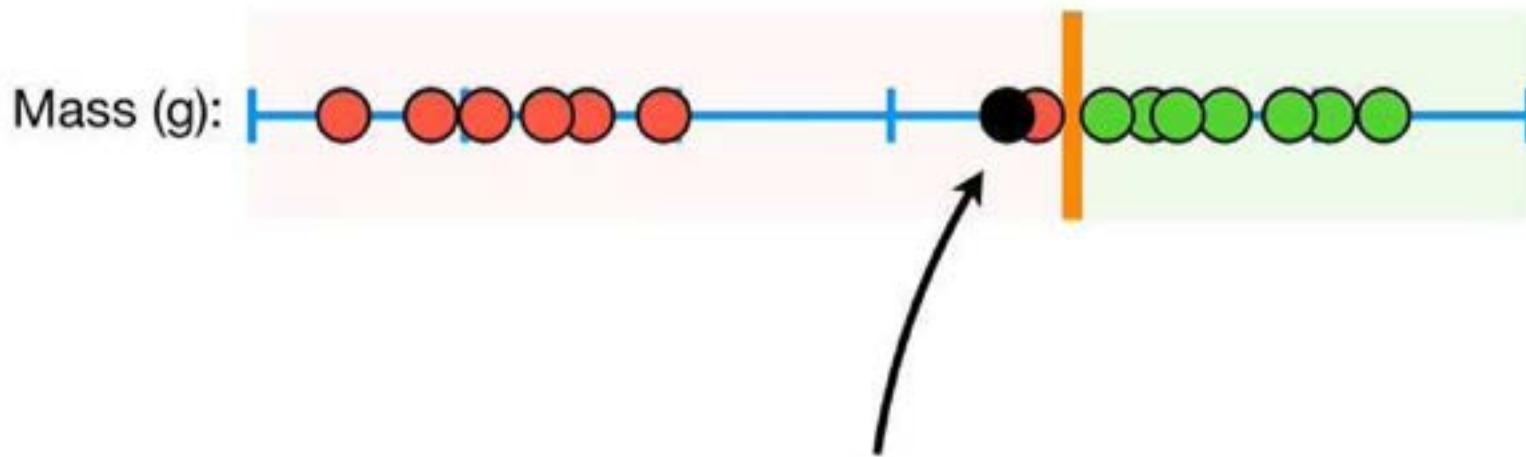
Problems with Maximal margin classifier



In this case, the **Maximum Margin Classifier** would be super close to the *obese* observations...

Soft Margin

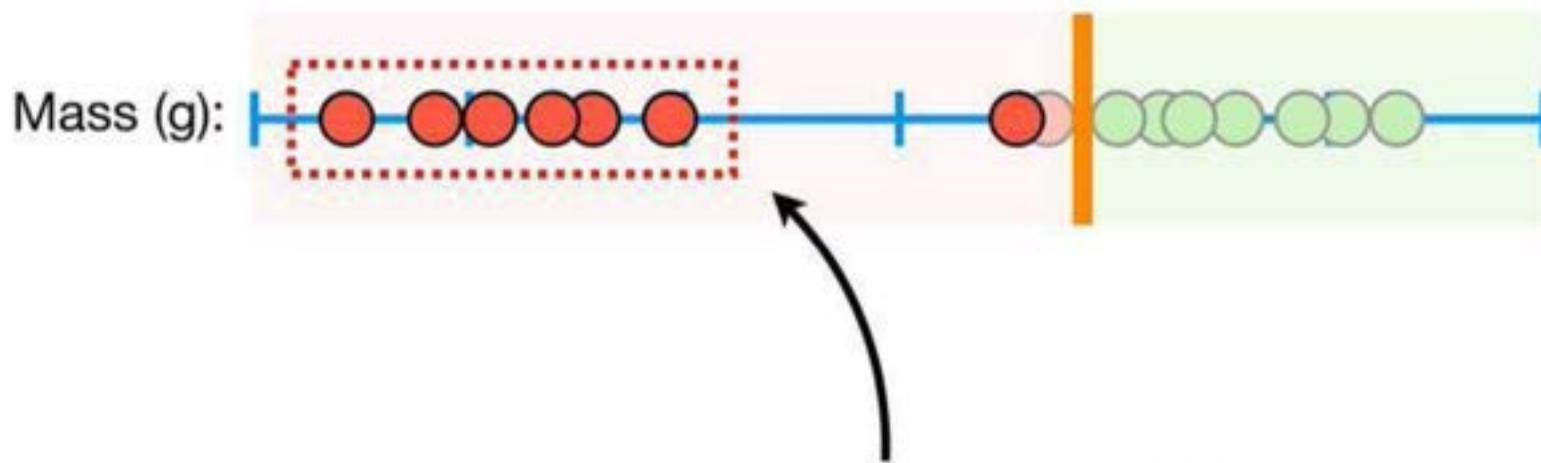
Problems with Maximal margin classifier



Now, if we got this new
observation...

Soft Margin

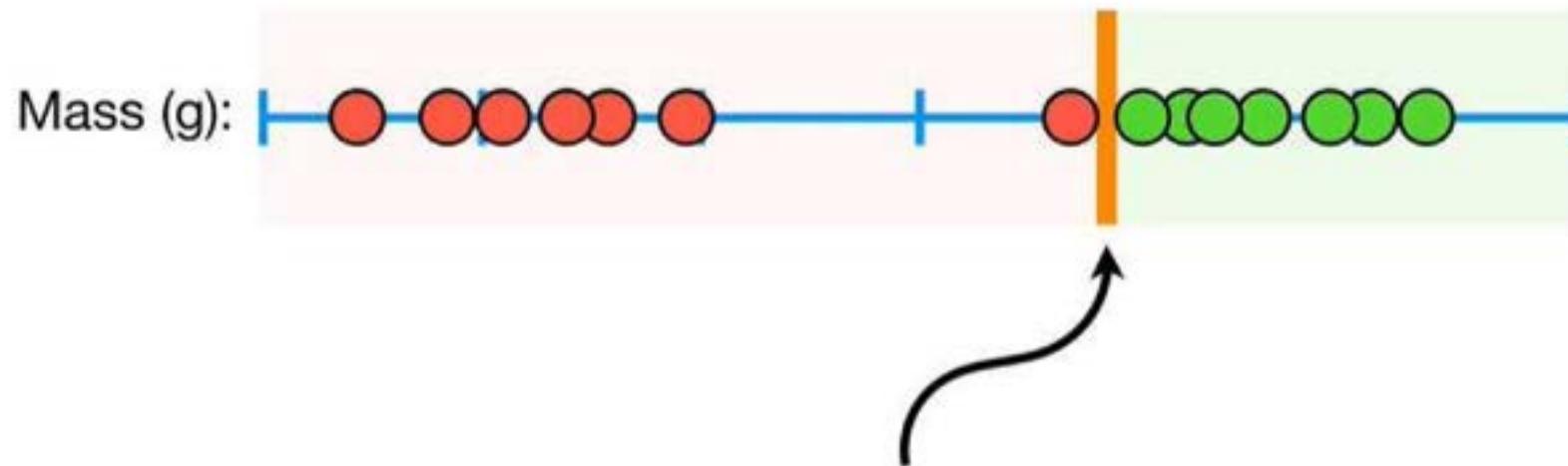
Problems with Maximal margin classifier



...we would classify it as **not obese**, even though most of the **not obese** observations are much further away than the **obese** observations.

Soft Margin

Problems with Maximal margin classifier

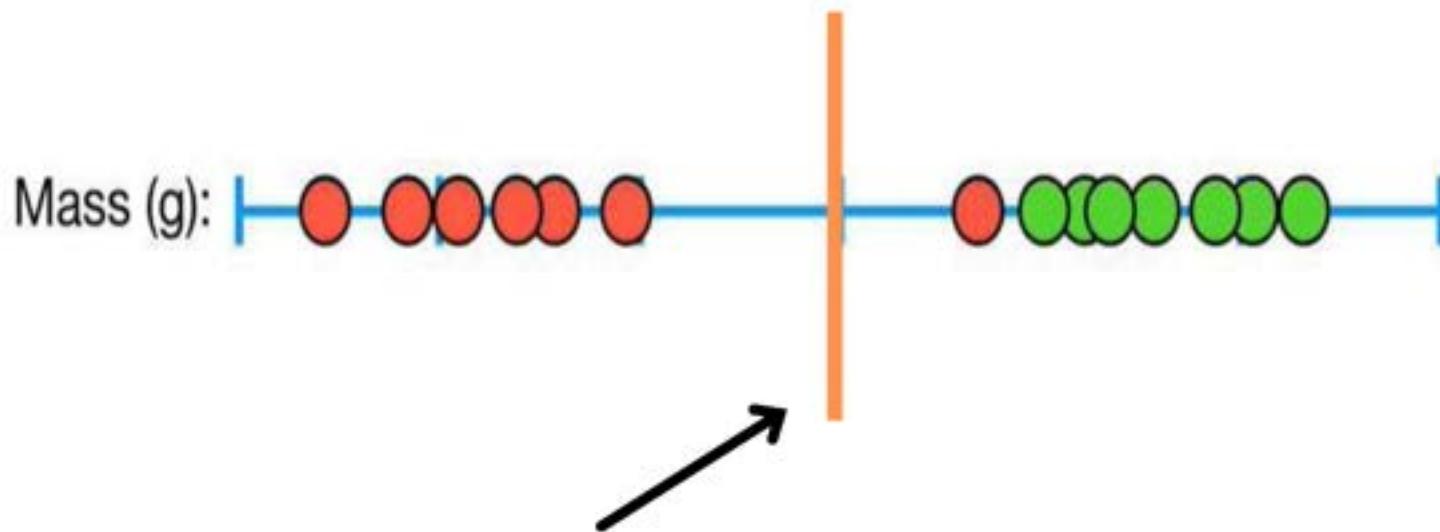


So **Maximal Margin Classifiers**
are *super sensitive to outliers* in the
training data and that makes them
pretty lame.

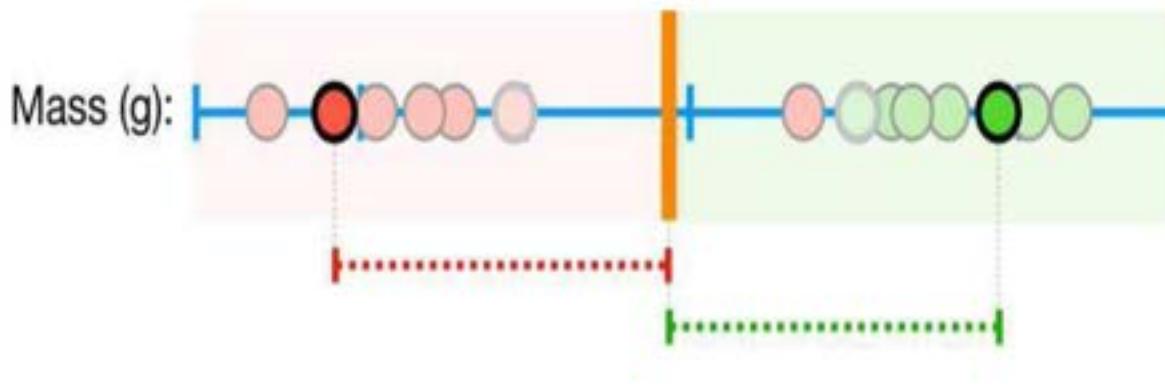
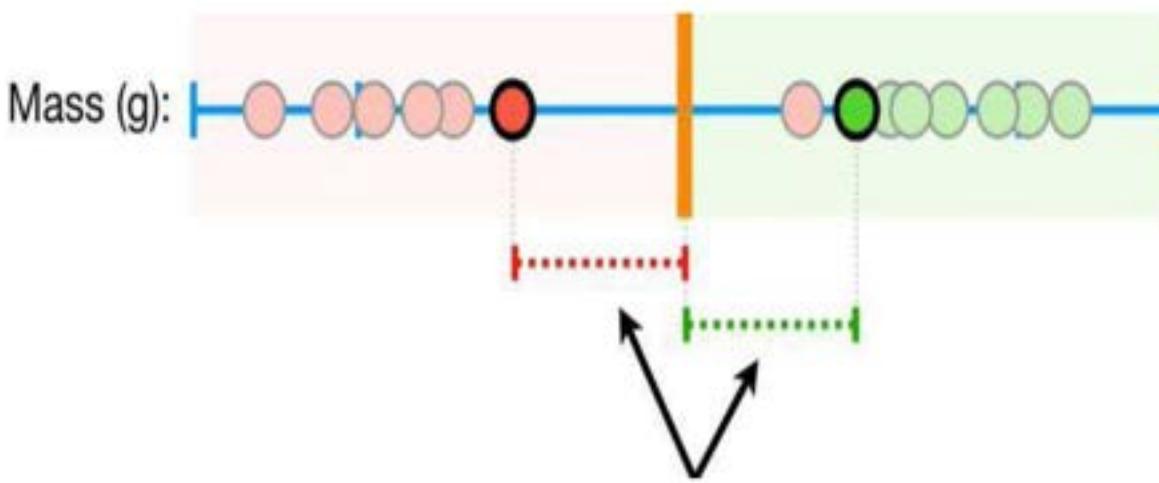
Soft Margin

Why Soft Margins?

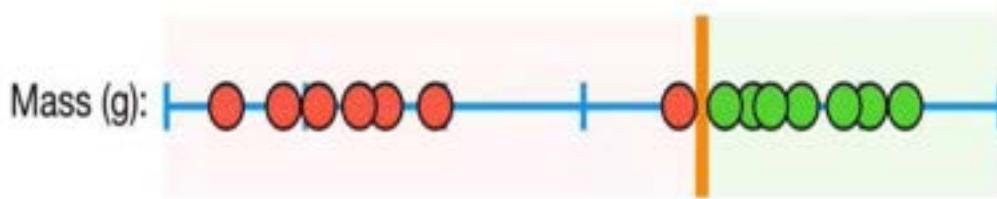
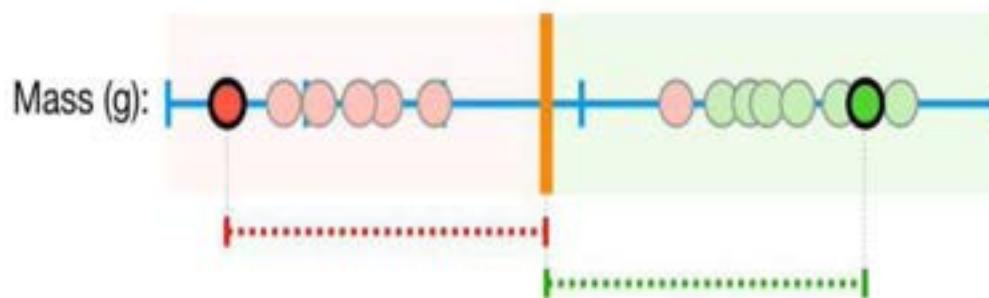
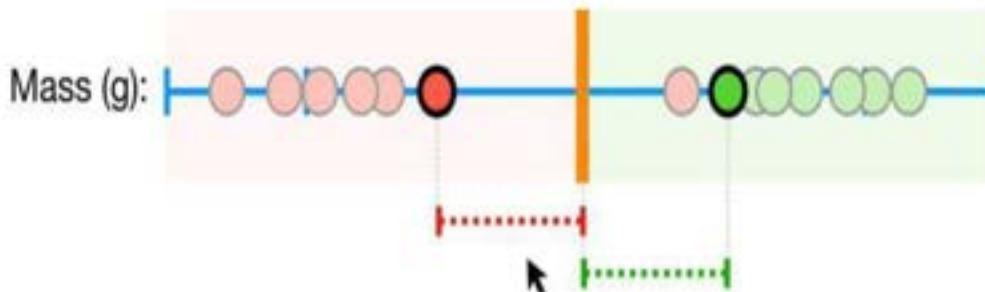
We've to draw a margin which allows misclassification



Why only here, and how is it determined?

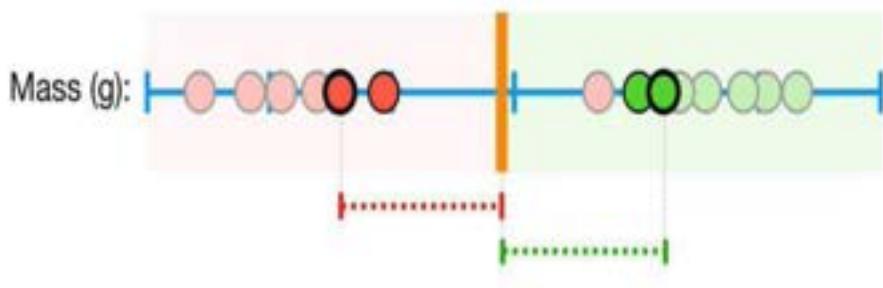


Which among them is better?

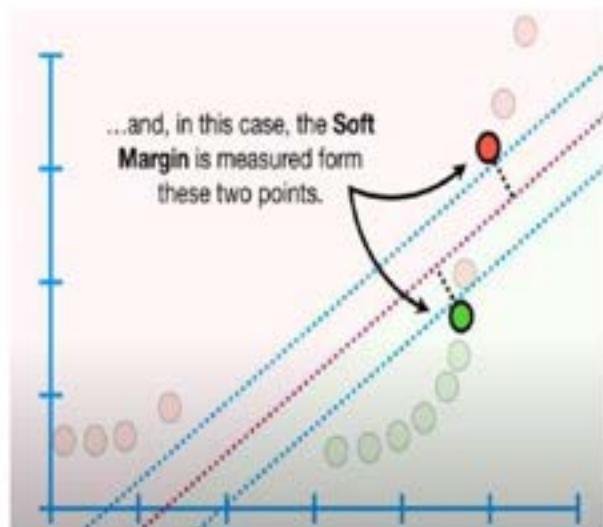


Cross-Validation is done with each pair of points to find
the optimum

Support vector Classifiers



Soft Margin is a Point



Soft Margin is a line

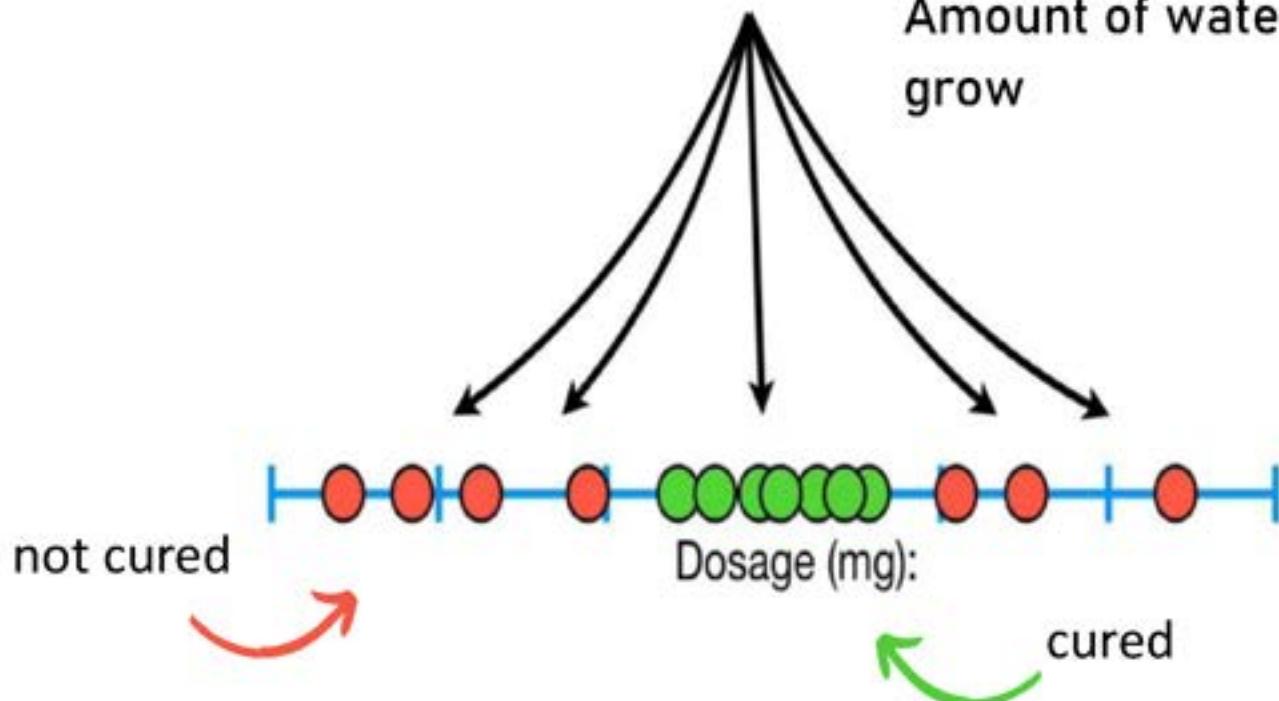
A plane in case of 3D
and rest are called as hyperplanes

SVM intuition

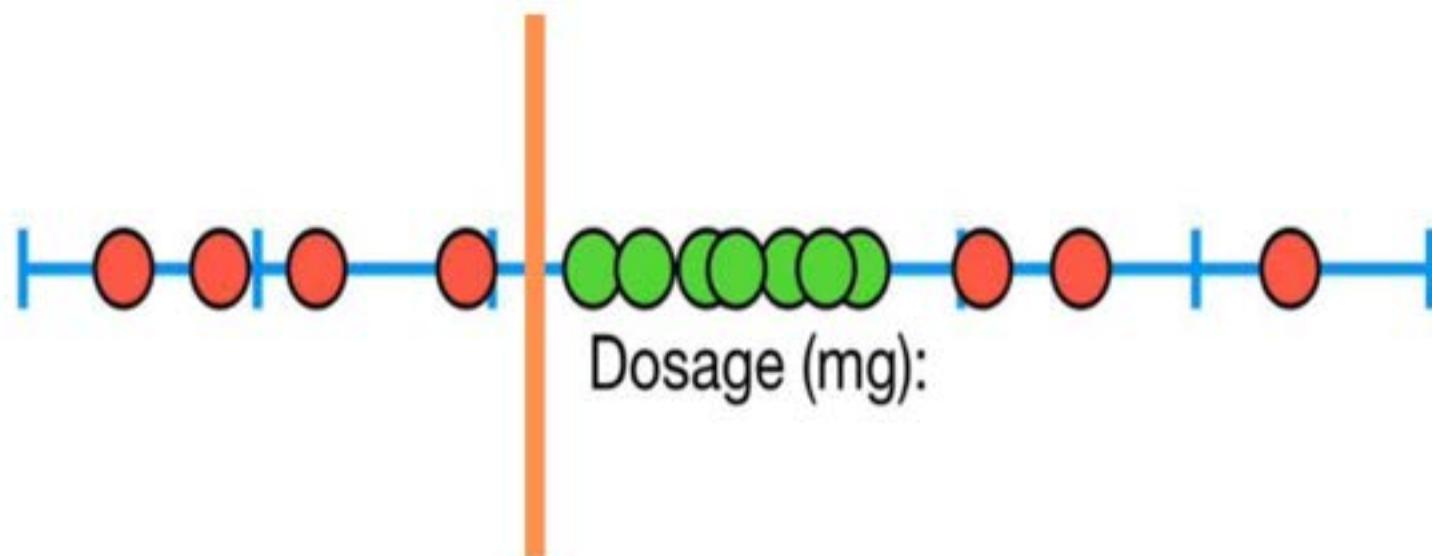
Example of such cases

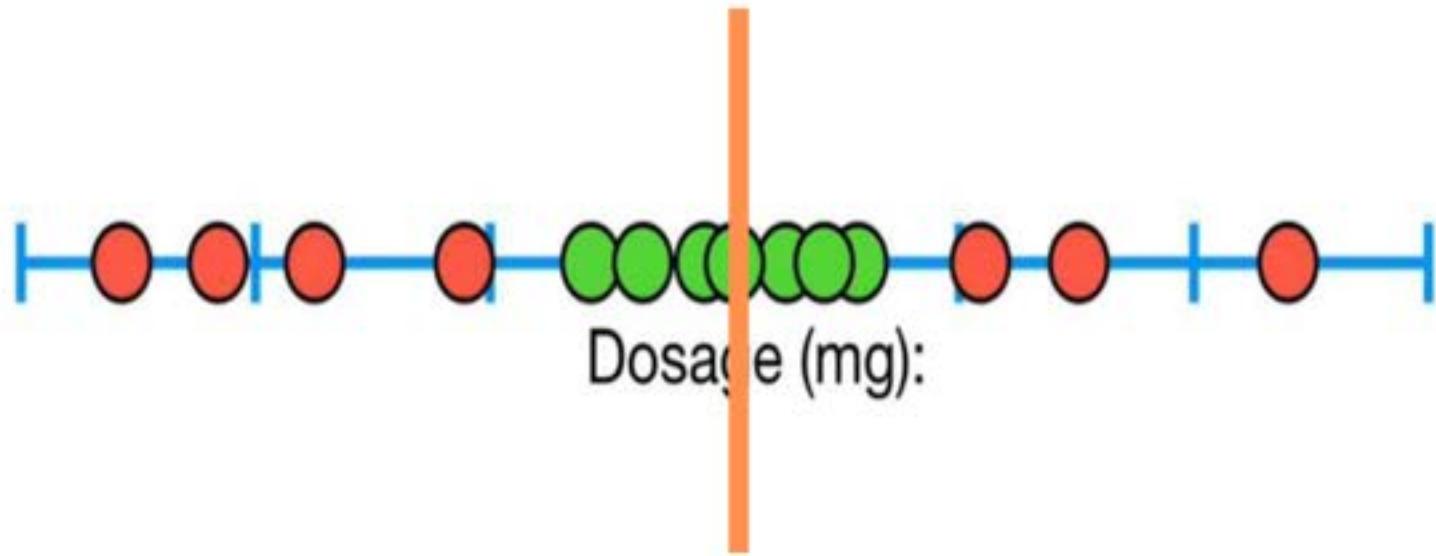
Dosage of medicine for recovery

Amount of water for a plant to grow

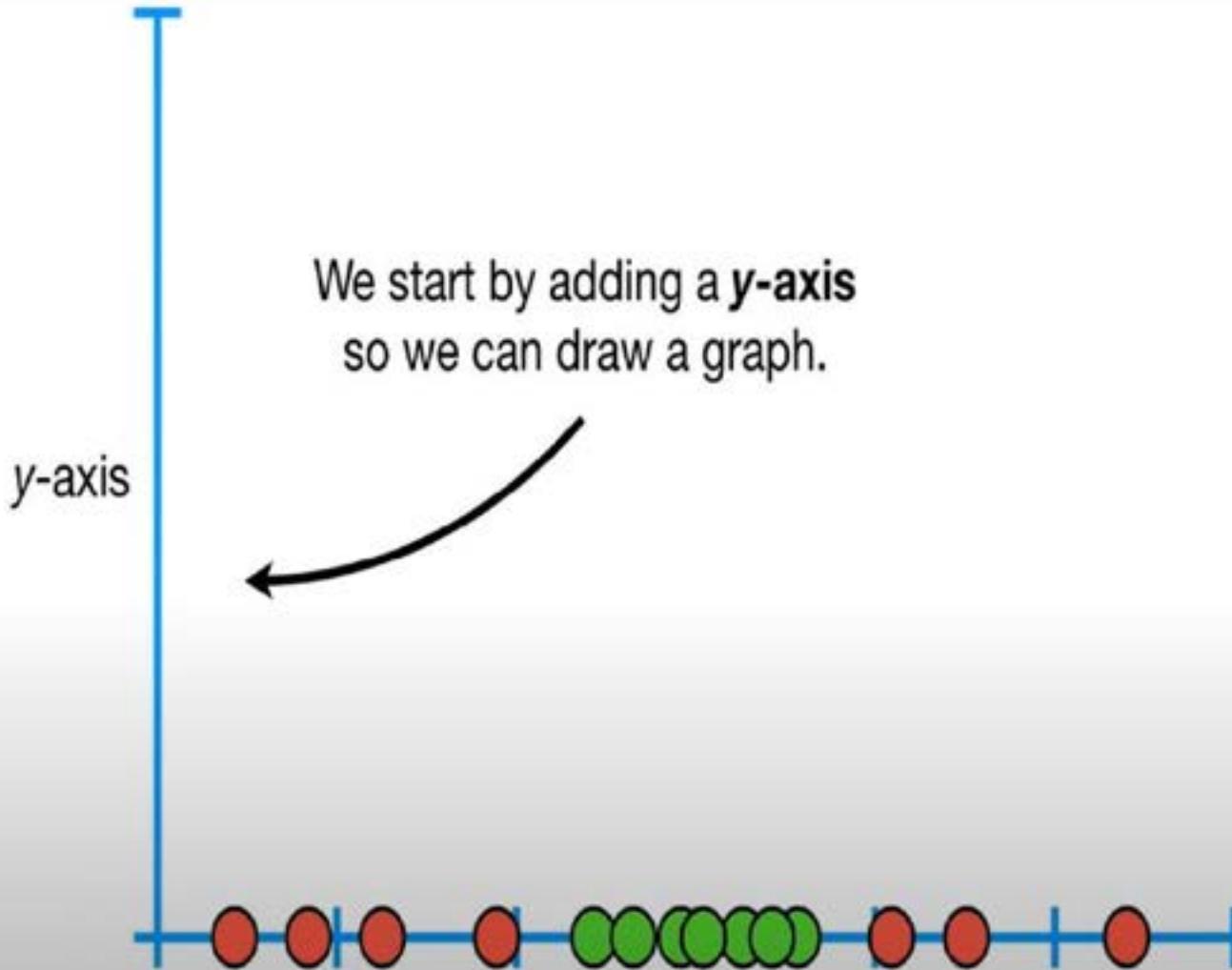


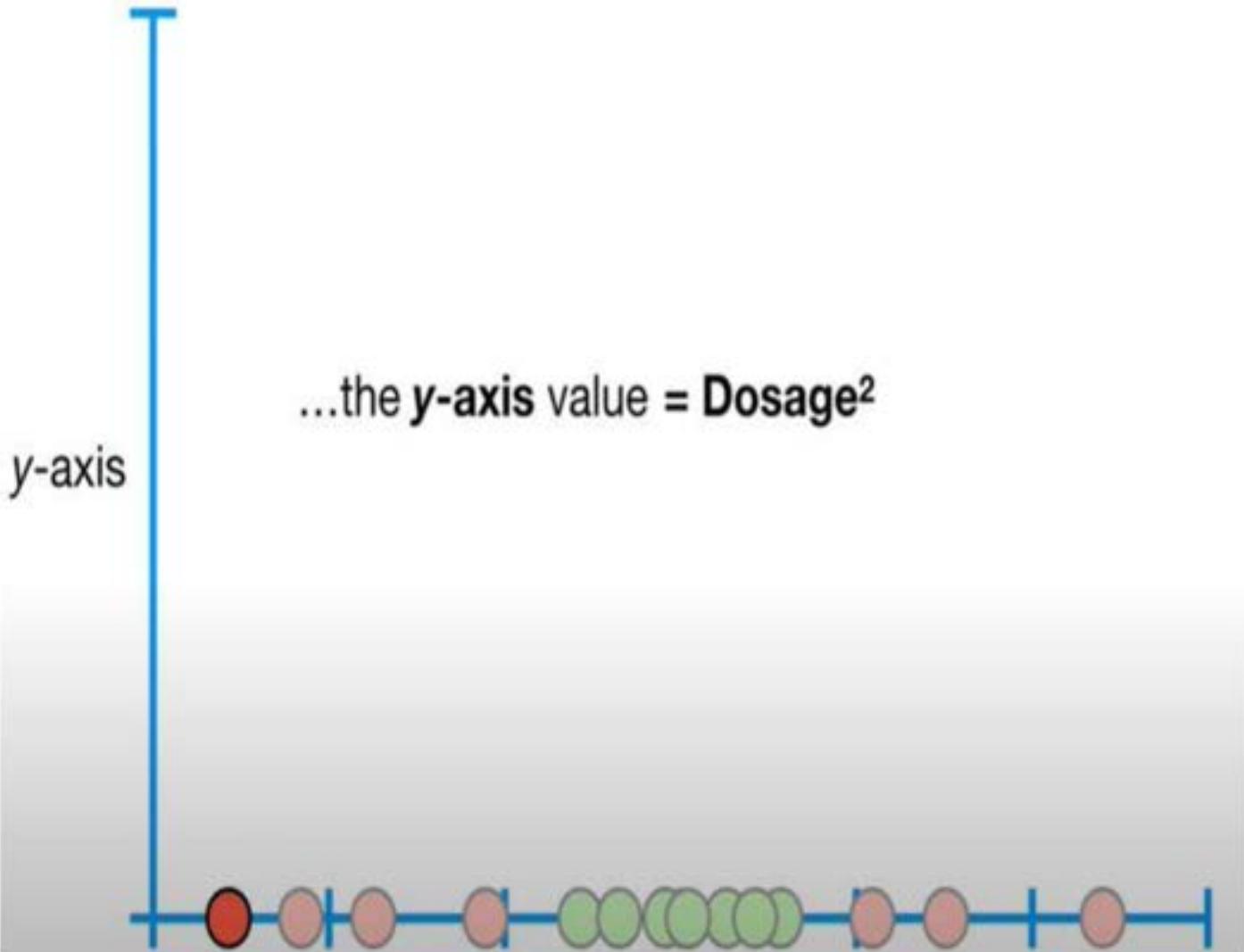
Does it work for all kinds of data?



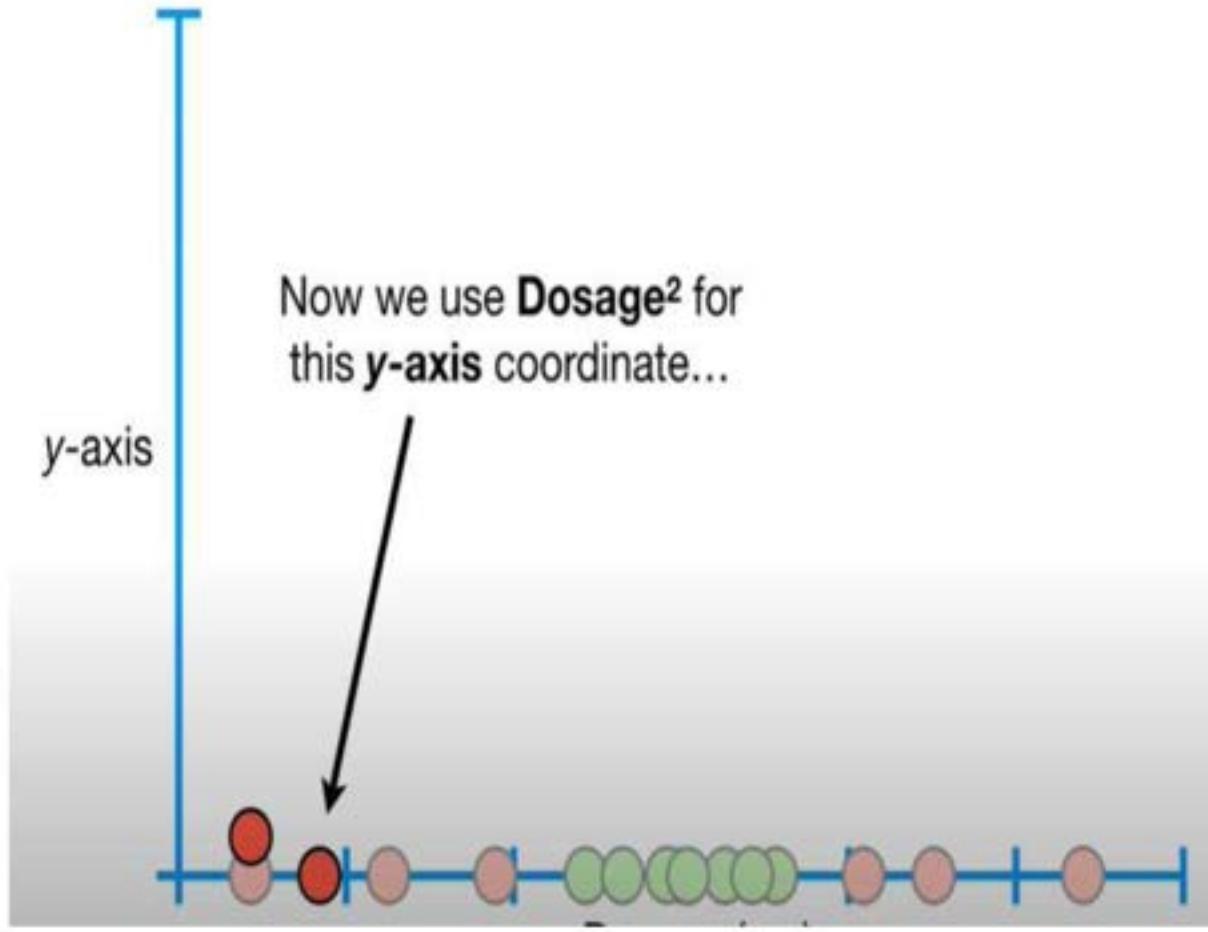


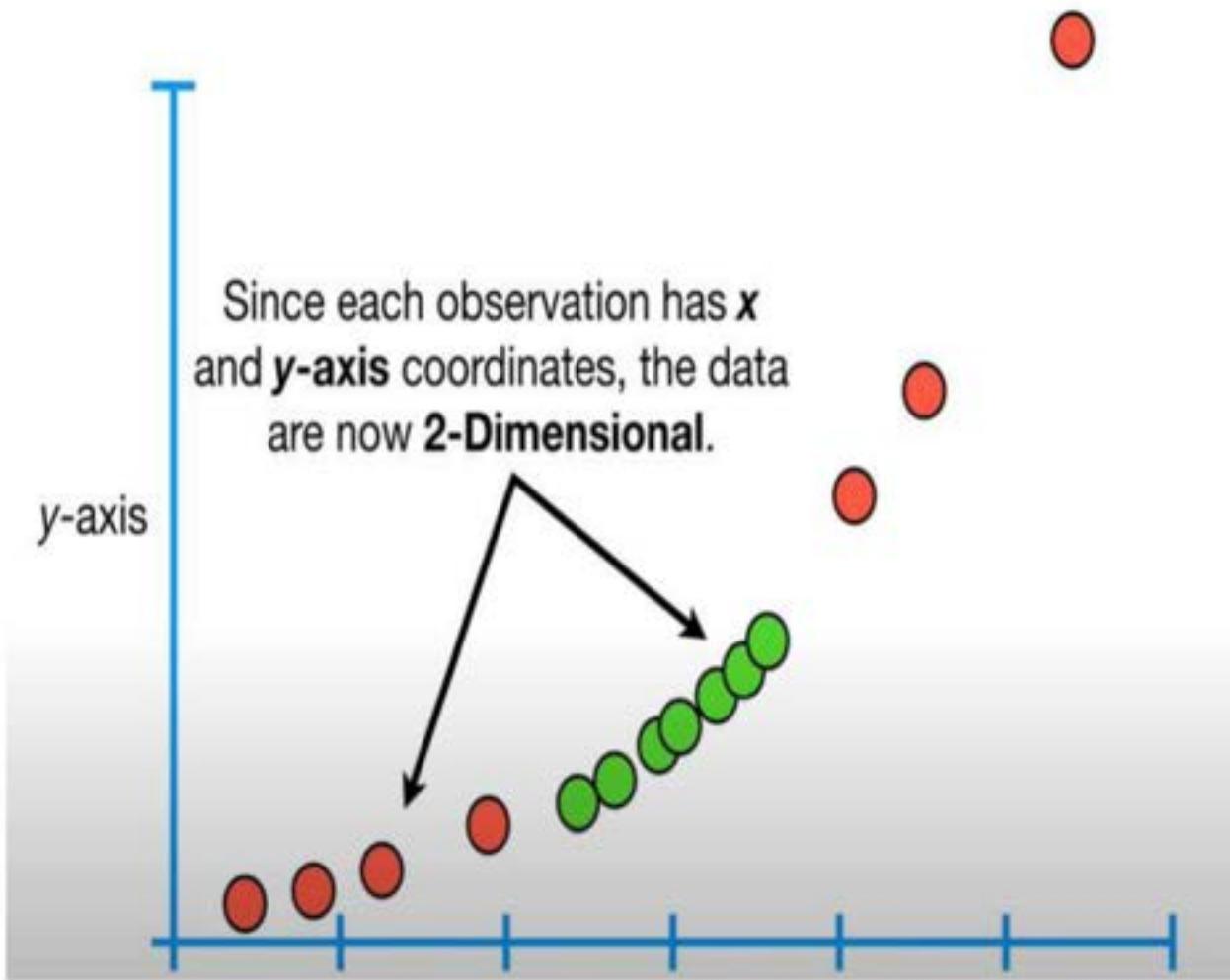
Can we do better?



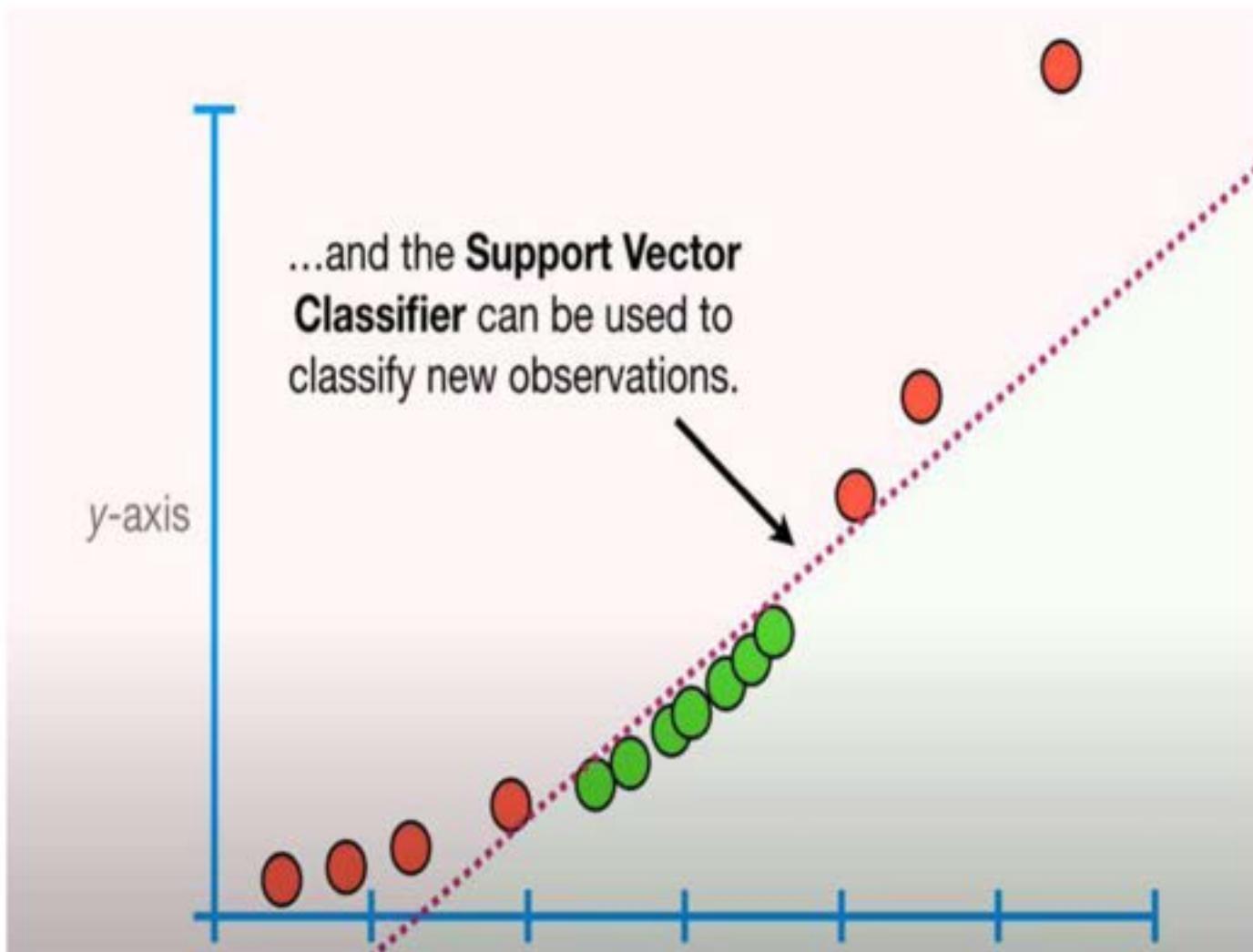


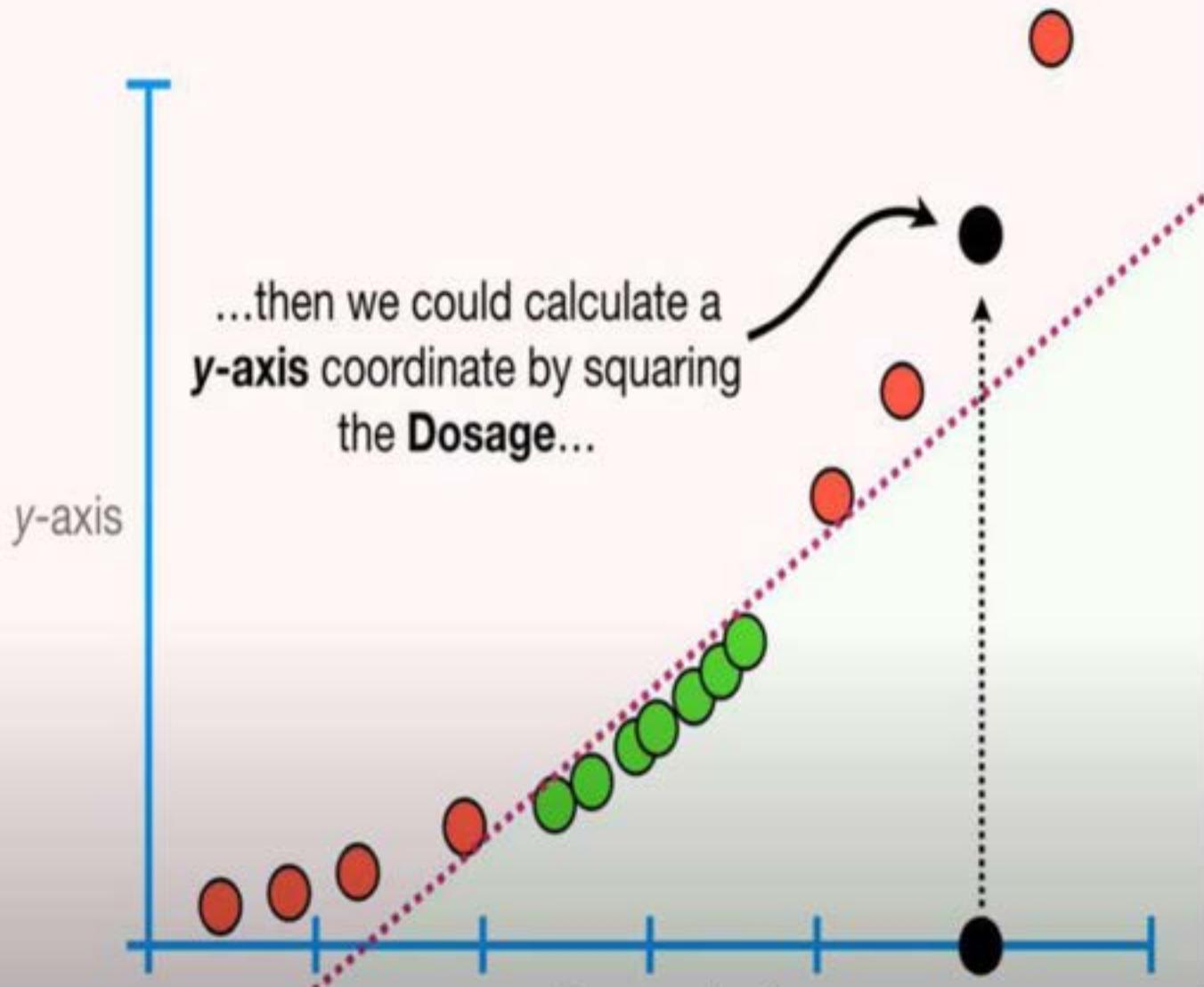
Now we use **Dosage²** for
this **y-axis** coordinate...



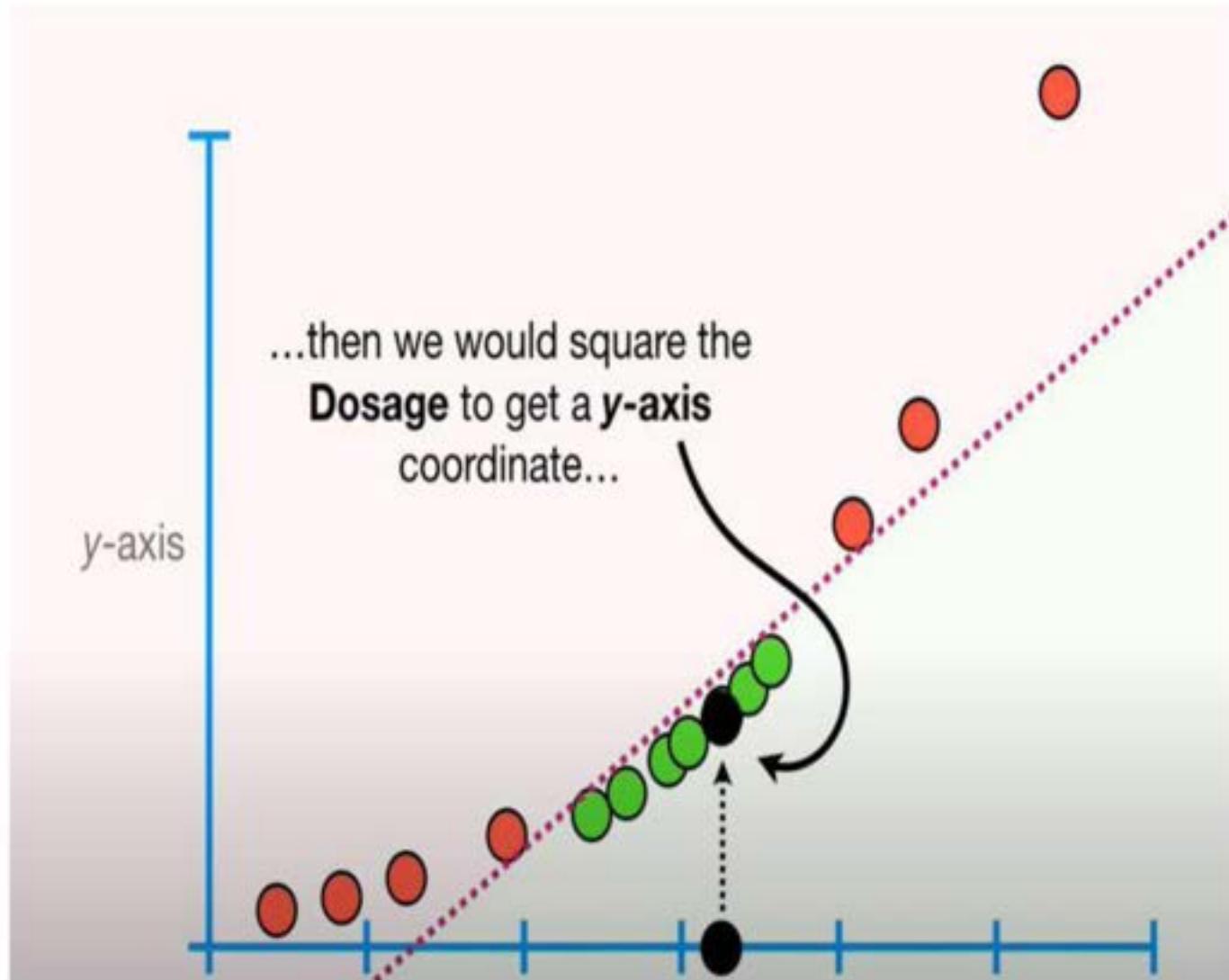


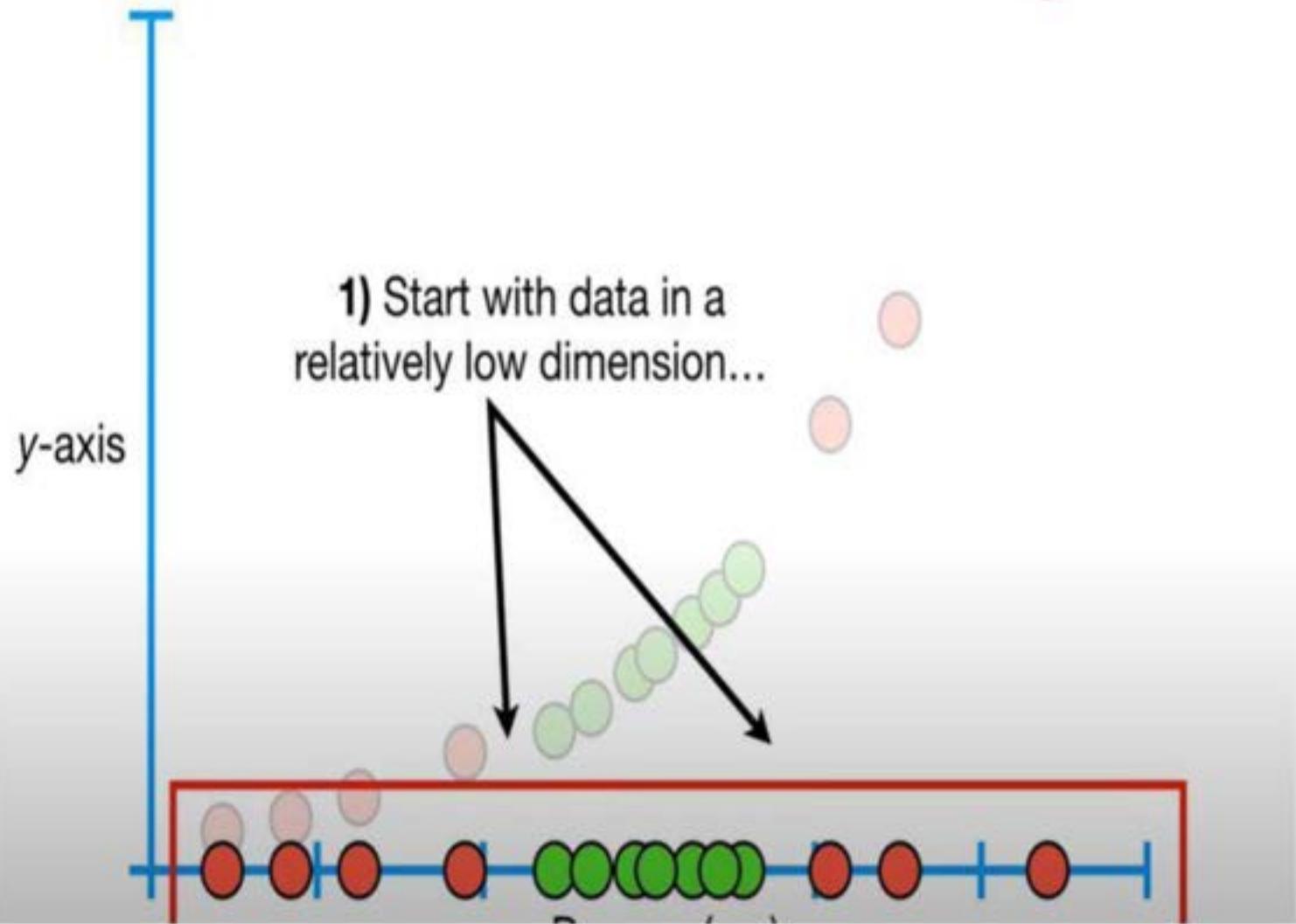
...and the **Support Vector Classifier** can be used to classify new observations.

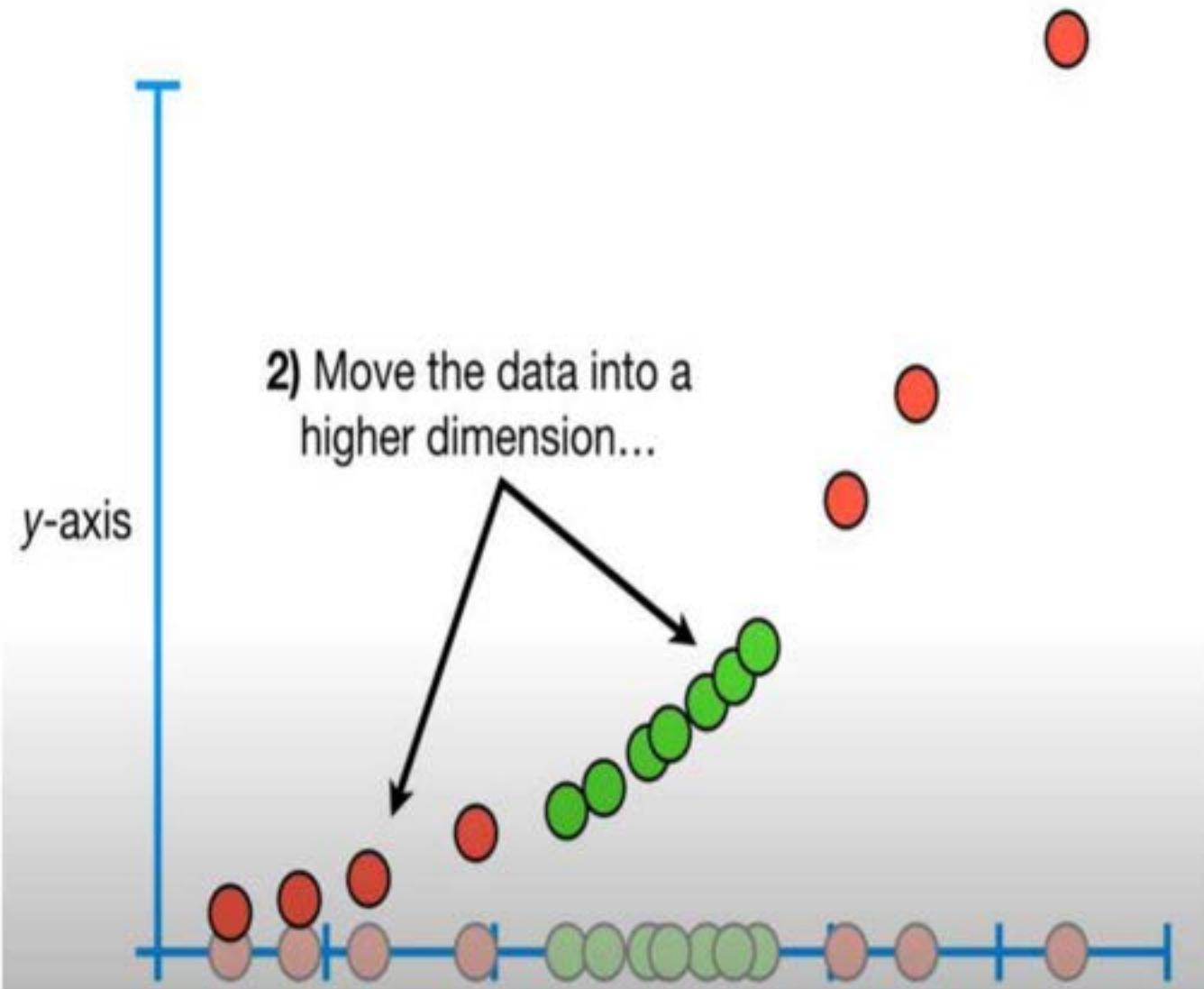




...then we would square the
Dosage to get a **y-axis**
coordinate...









3) Find a Support Vector Classifier that separates the higher dimensional data into two groups.

How to choose a function?

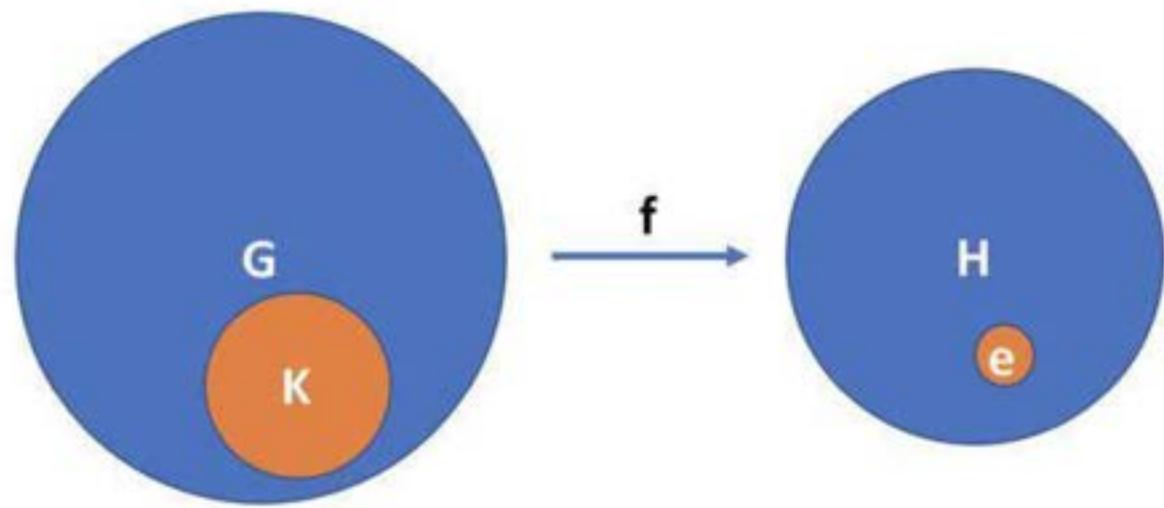
"Kernel" is a set of mathematical functions used in Support Vector Machine that provides the window to manipulate the data.

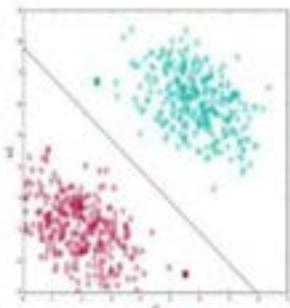
In our case we have used $y = x^2$, this is a polynomial of x and thus called a polynomial kernel.

There are several other kernels that are used by SVM, namely Radial Basis Function (RBF), Laplace RBF Kernel, Sigmoid Kernel, etc.

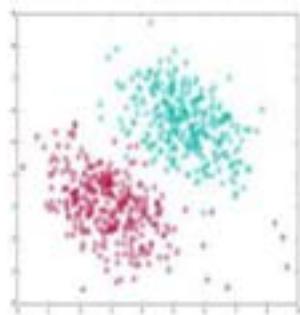


Kernels

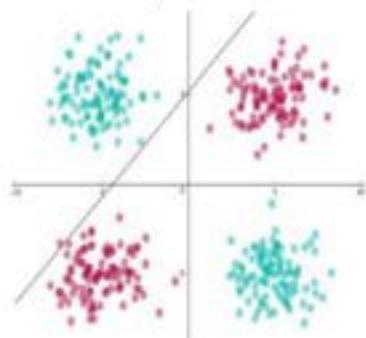




Linearly Separable



Almost Linearly Separable

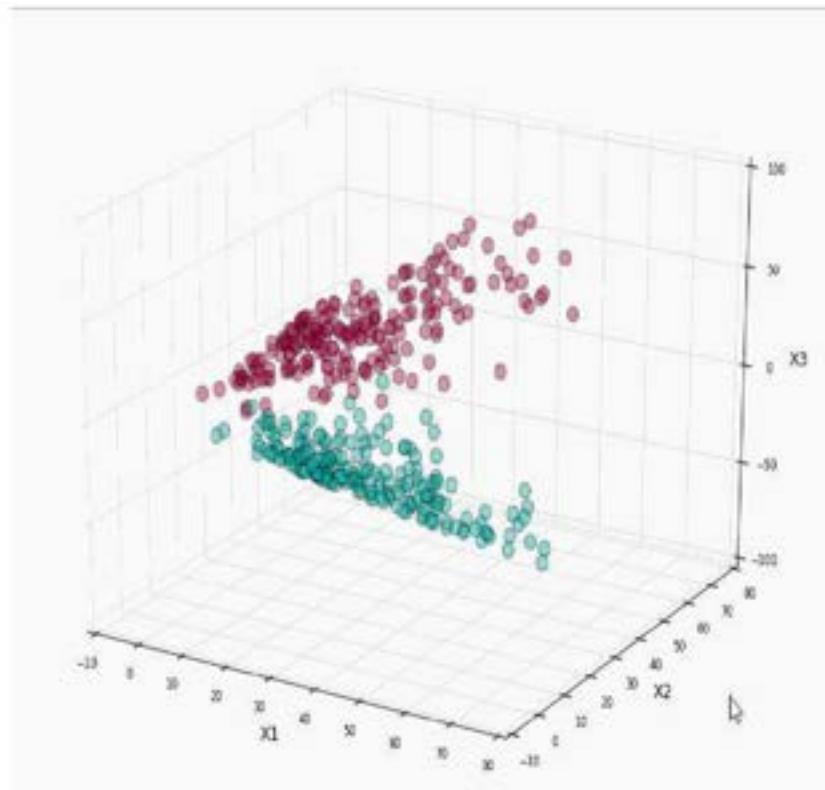


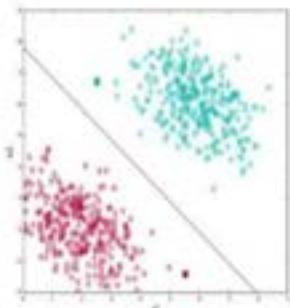
Non Linearly Separable

$$X_1 = x_1^2$$

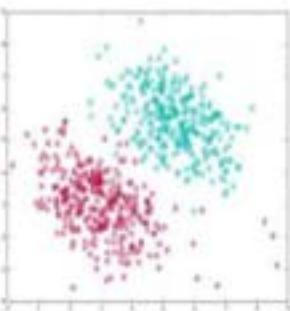
$$X_2 = x_2^2$$

$$X_3 = \sqrt{2}x_1x_2$$

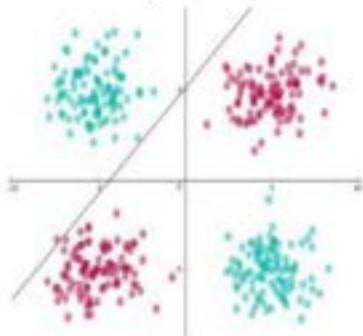




Linearly Separable



Almost Linearly Separable

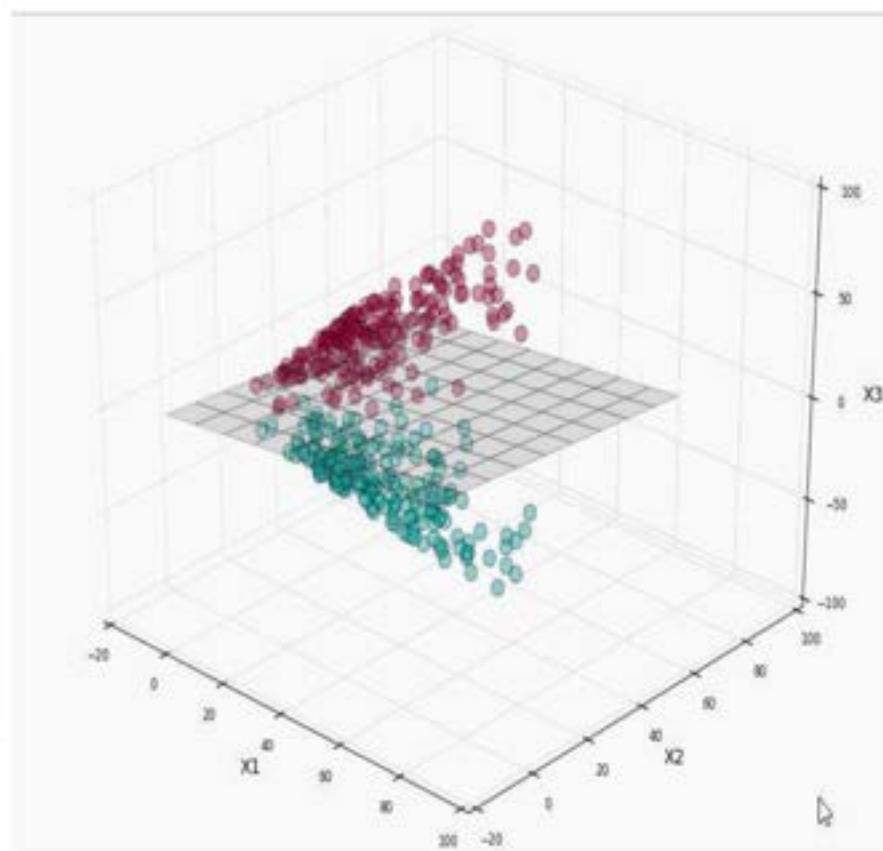
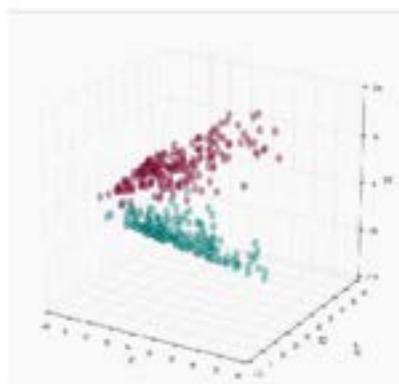


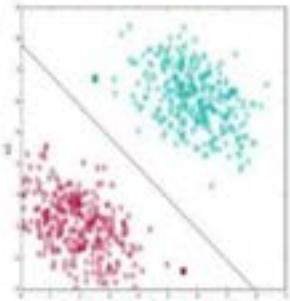
Non Linearly Separable

$$X_1 = x_1^2$$

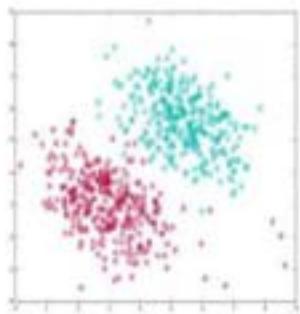
$$X_2 = x_2^2$$

$$X_3 = \sqrt{2}x_1x_2$$

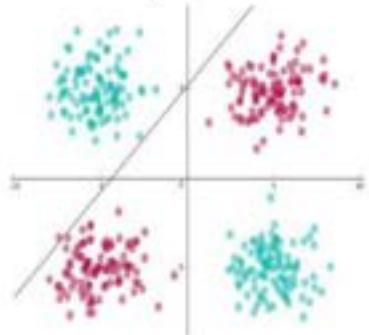




Linearly Separable



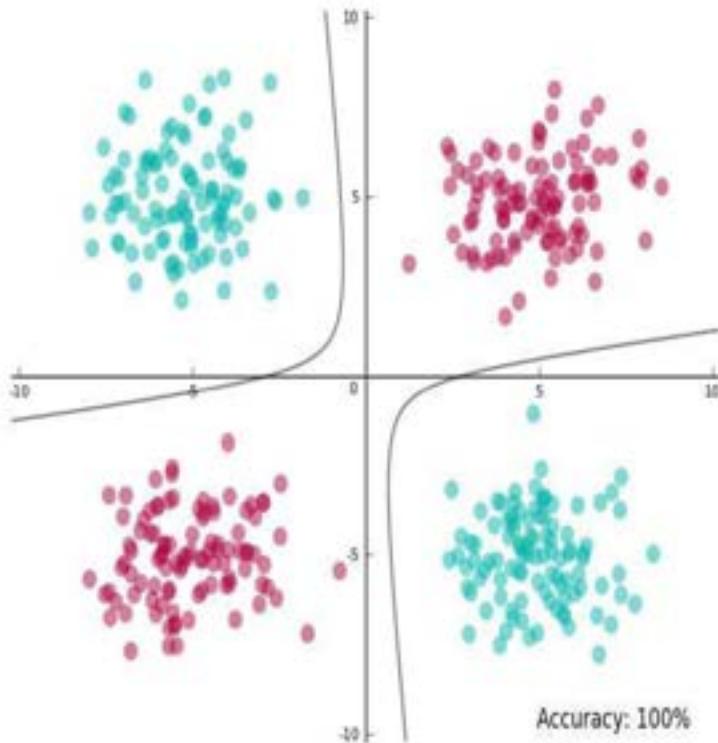
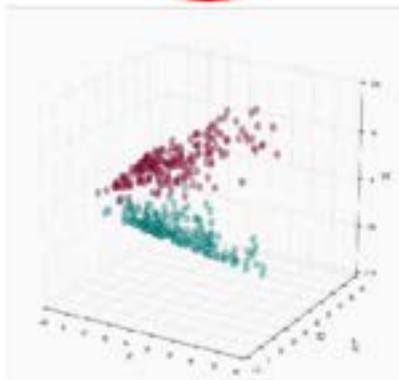
Almost Linearly Separable



Non Linearly Separable

$$\begin{aligned}X_1 &= x_1^2 \\X_2 &= x_2^2 \\X_3 &= \sqrt{2}x_1x_2\end{aligned}$$

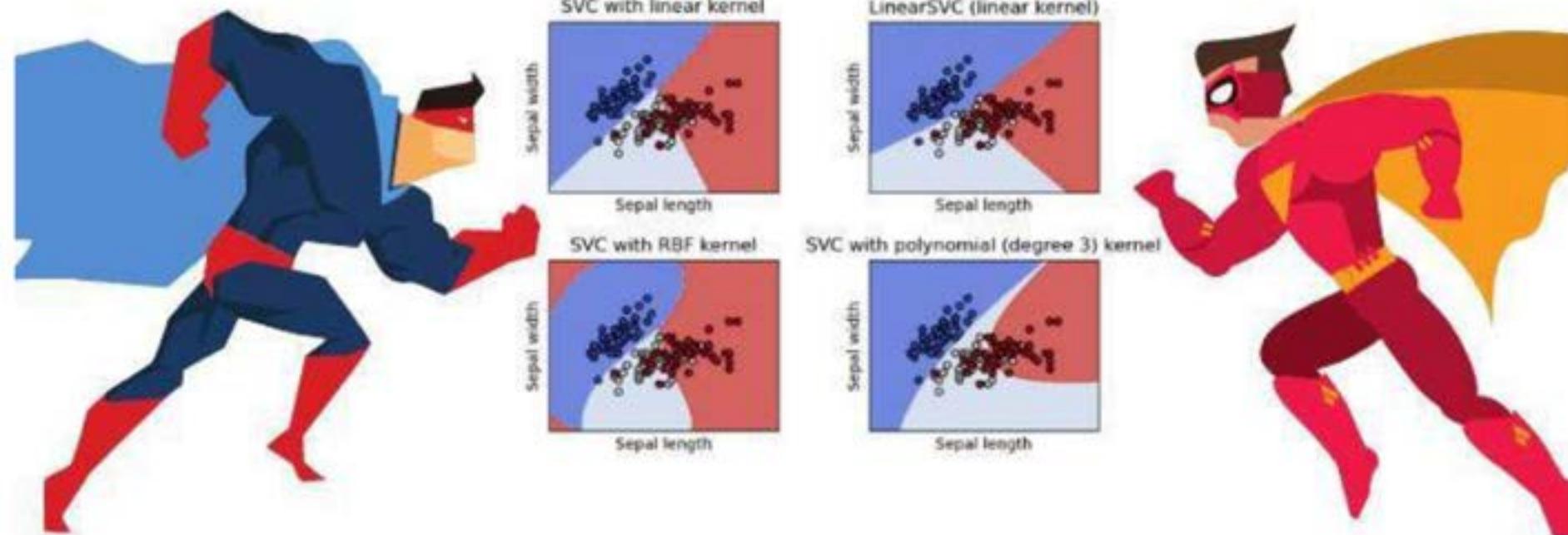
Kernel : set of mathematical functions to manipulate the data



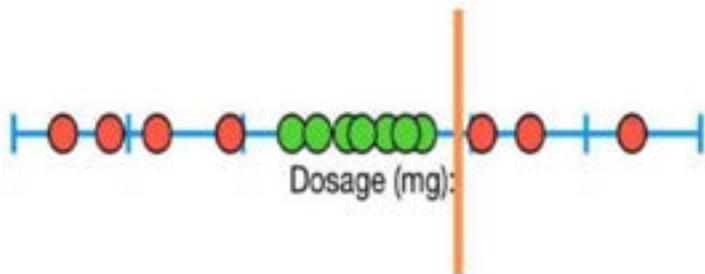
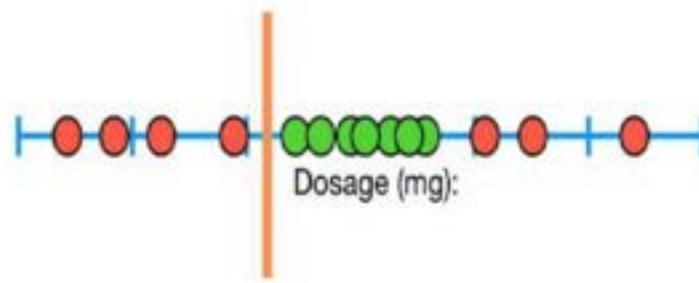
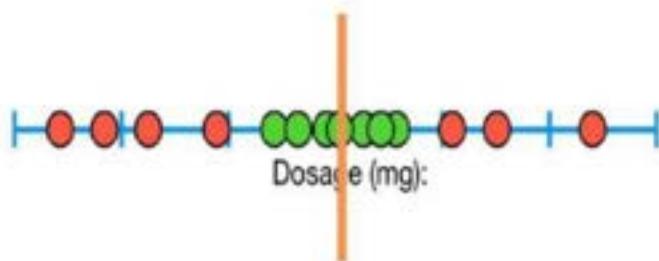
Accuracy: 100%

Types of Kernels

- Popular Kernels
 - Polynomial Kernel
 - Radial Kernel
 - Sigmoid



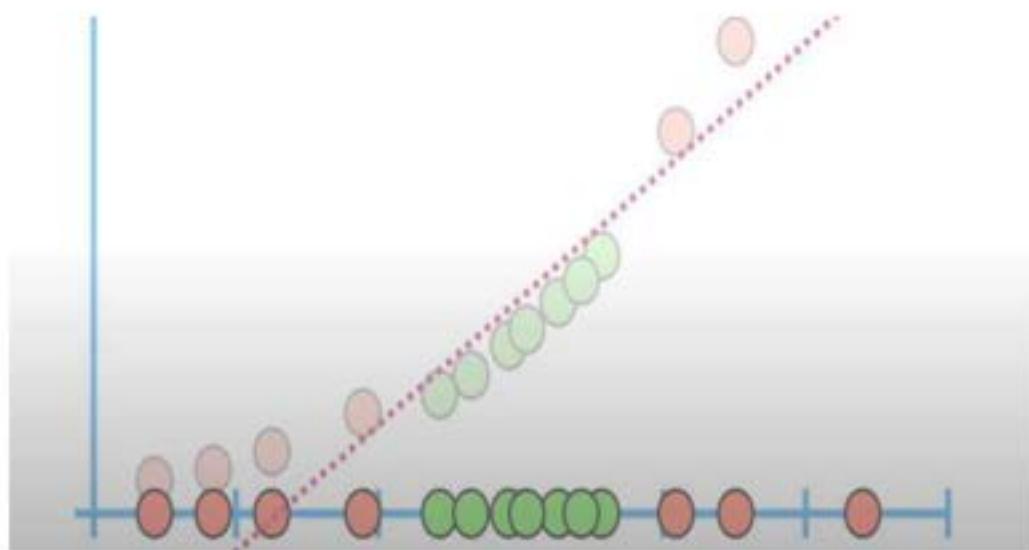
The Polynomial Kernel



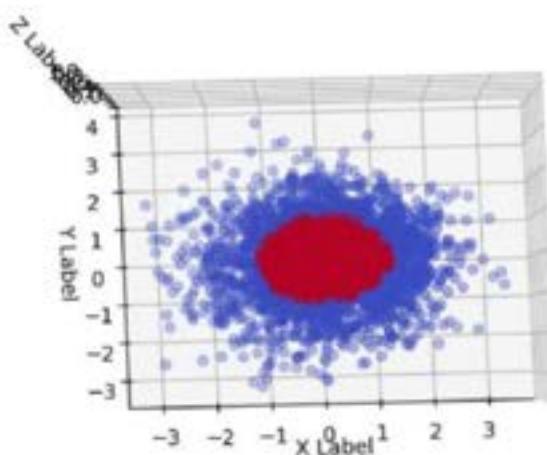
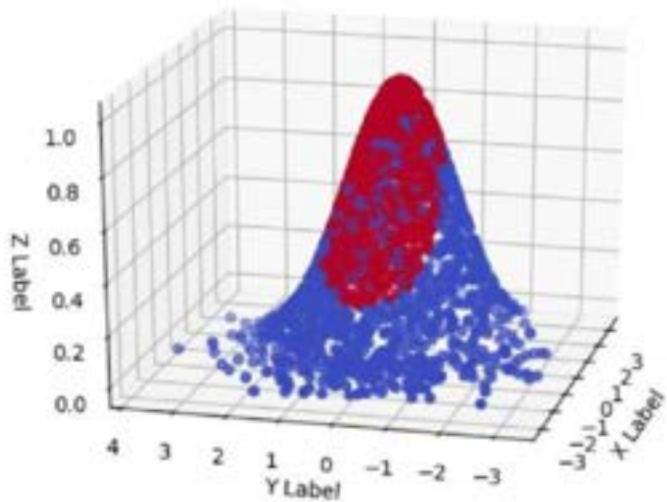
The Polynomial Kernel

$$(a \times b + r)^d$$

- **a** and **b** – two different observations on the dataset
- **r** – coefficient of the polynomial
- **d** – degree of the polynomial



What if we have data like this?



The Radial Kernel

- Finds Support Vector Classifiers in infinite dimensions
- For samples like the below data, Radial Kernel behaves like a **Weighted Nearest Model**

$$e^{-\gamma(a-b)^2}$$

- a and b - observations
- Gamma - scales the influence





Machine Learning (19CSE305)

Decision Tree

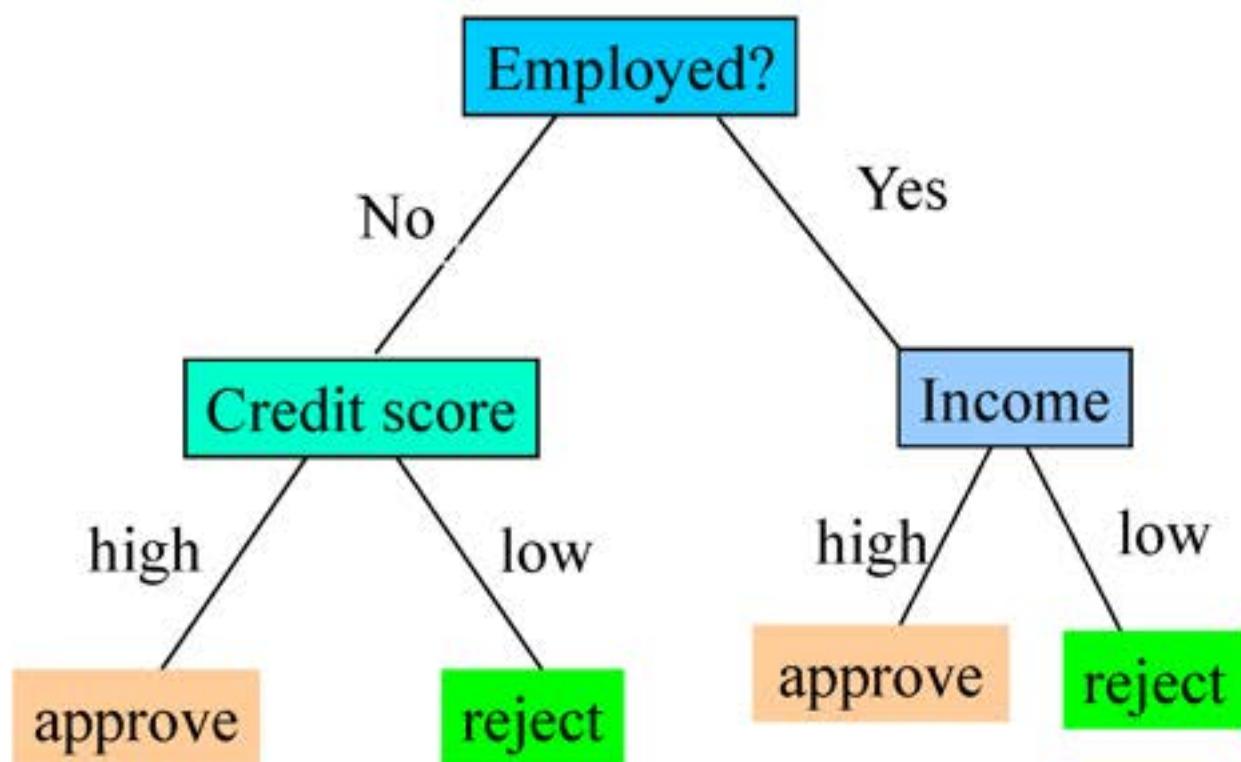


Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

What are Decision trees?

- A decision tree is a tree in which each branch node represents a choice between a number of alternatives, and each leaf node represents a decision.
- A type of supervised learning algorithm.

Decision Tree An Example



Whether to approve/reject
a loan application?

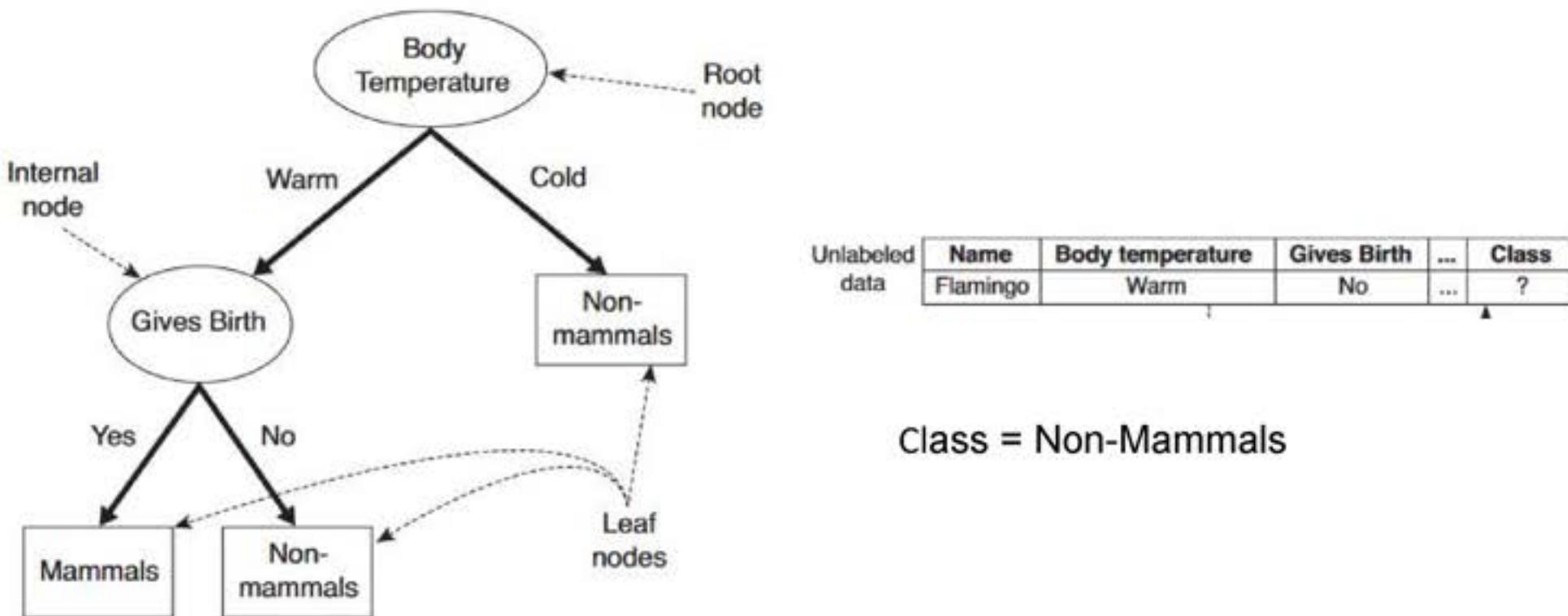
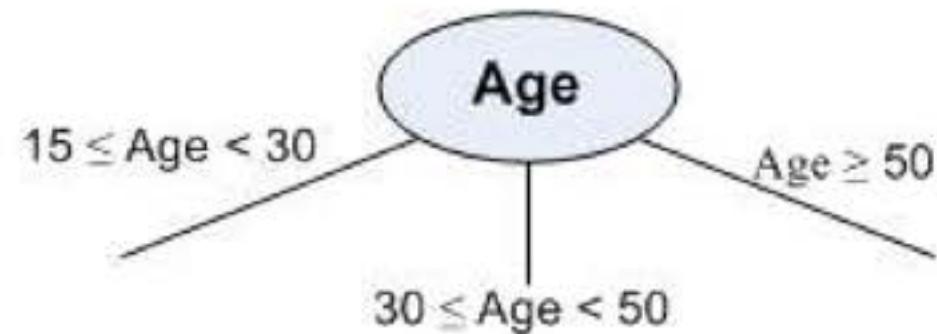
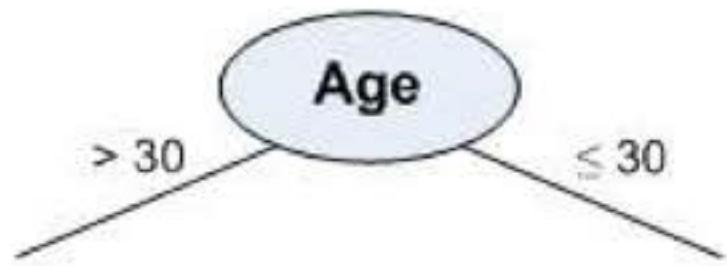
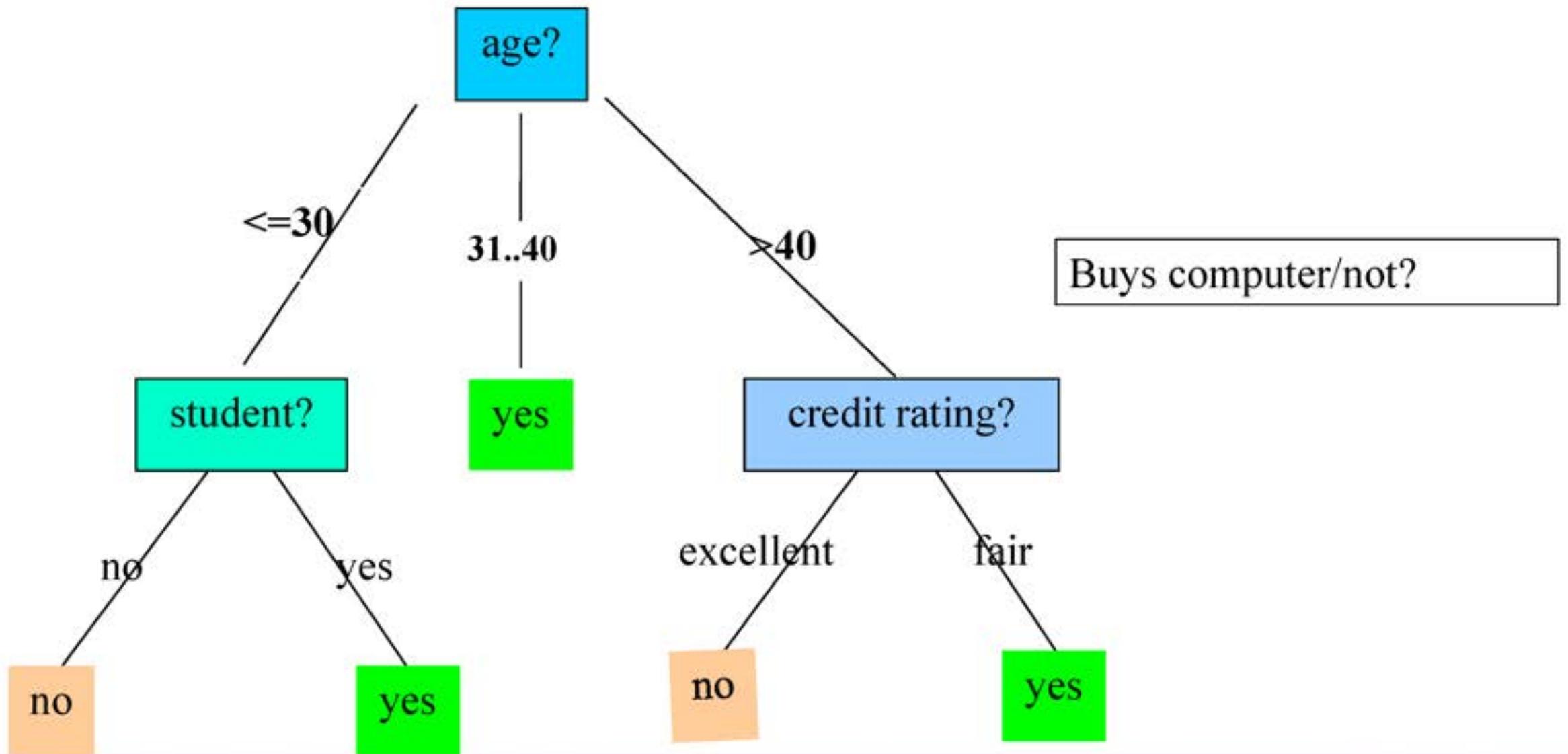


Figure 4.4. A decision tree for the mammal classification problem.

Numerical attribute





Example data

Training Examples:

	Action	Author	Thread	Length	Where
e1	skips	known	new	long	Home
e2	reads	unknown	new	short	Work
e3	skips	unknown	old	long	Work
e4	skips	known	old	long	home
e5	reads	known	new	short	home
e6	skips	known	old	long	work

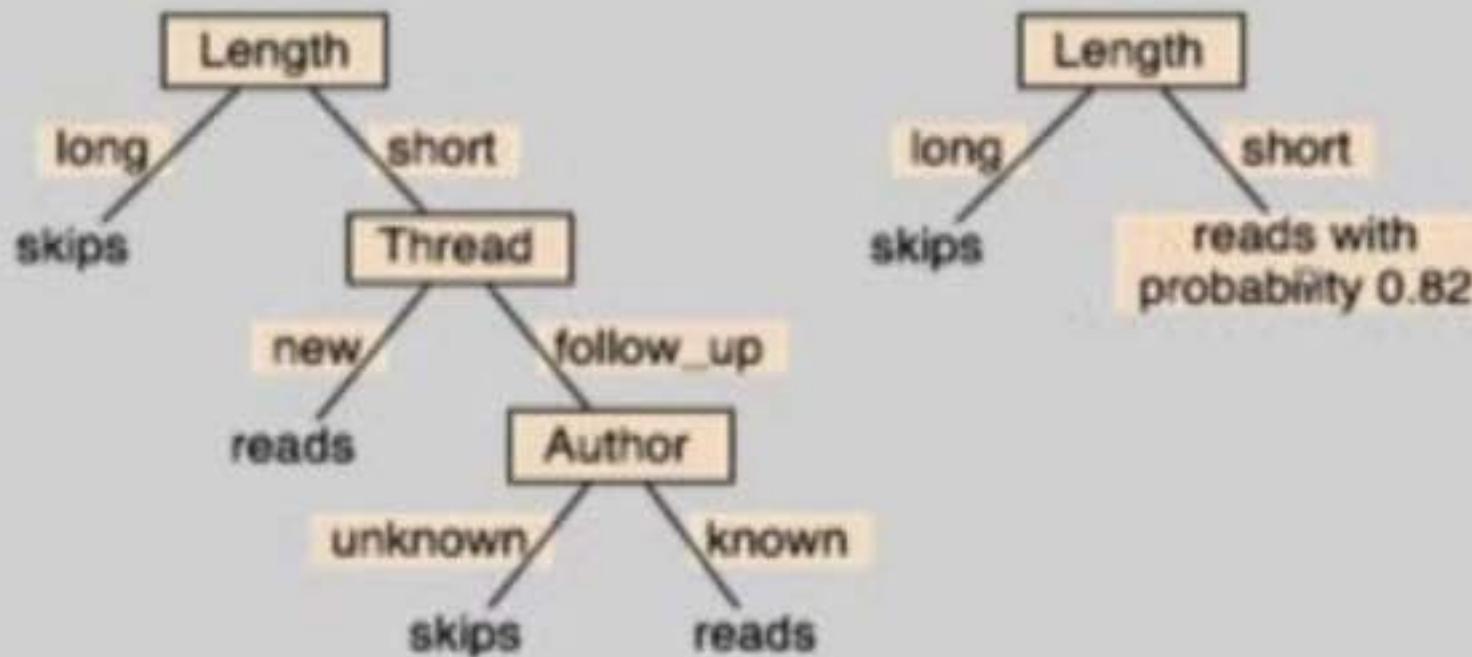
New Examples:

e7	???	known	new	short	work
e8	???	unknown	new	short	work

Possible splits



Two Example DTs



Basic Algorithm for Top-Down Induction of Decision Trees

[ID3, C4.5 by Quinlan]

node = root of decision tree

Main loop:

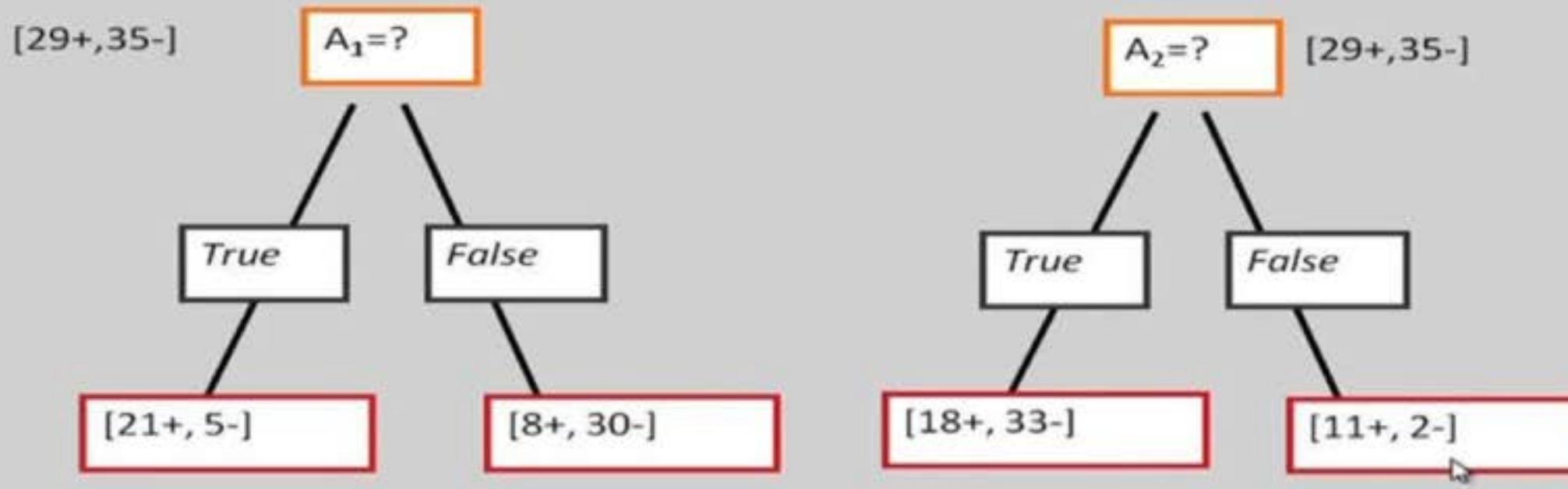
1. $A \leftarrow$ the “best” decision attribute for the next node.
2. Assign A as decision attribute for *node*.
3. For each value of A , create a new descendant of *node*.
4. Sort training examples to leaf nodes.
5. If training examples are perfectly classified, stop.
Else, recurse over new leaf nodes.

How do we choose which attribute is best?

Choices

- When to stop
 - no more input features
 - all examples are classified the same
 - too few examples to make an informative split
- Which test to split on
 - split gives smallest error.
 - With multi-valued features
 - split on all values or
 - split values into half.

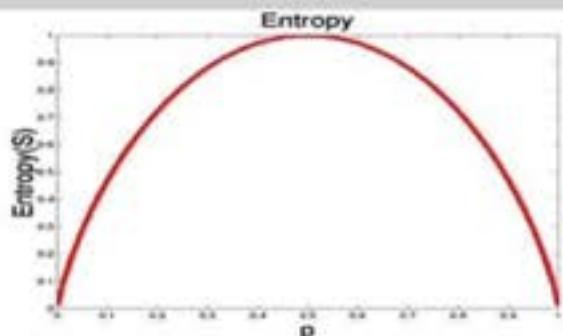
Which Attribute is "best"?



Principled Criterion

- Selection of an attribute to test at each node - choosing the most useful attribute for classifying examples.
- **information gain**
 - measures how well a given attribute separates the training examples according to their target classification
 - This measure is used to select among the candidate attributes at each step while growing the tree
 - Gain is measure of how much we can reduce uncertainty (Value lies between 0,1)

Entropy



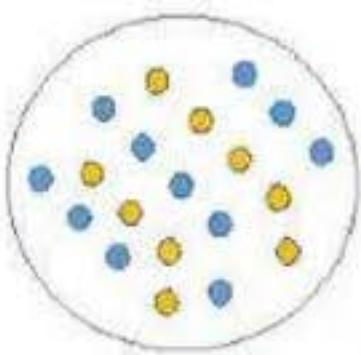
- The entropy is 0 if the outcome is "certain".
- The entropy is maximum if we have no knowledge of the system (or any outcome is equally possible).

- S is a sample of training examples
- p_+ is the proportion of positive examples
- p_- is the proportion of negative examples
- Entropy measures the impurity of S

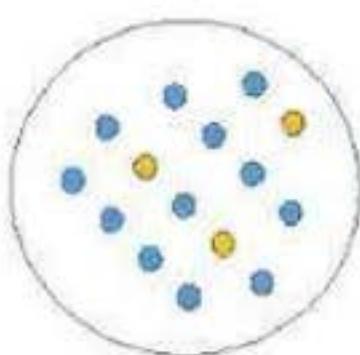
$$\text{Entropy}(S) = -p_+ \log_2 p_+ - p_- \log_2 p_-$$

How to choose best decision node

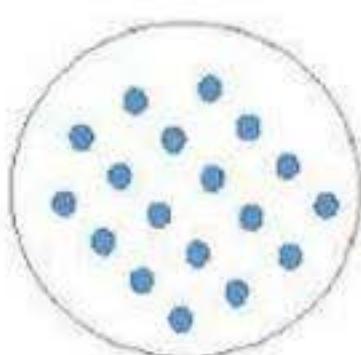
Which node can be described easily?



A



B



C

- Less impure node requires less information to describe it.
- More impure node requires more information.

====> Information theory is a measure to define this degree of disorganization in a system known as **Entropy**.

- **Entropy is 0** if all the members of S belong to the same class.
- **Entropy is 1** when the collection contains an equal no. of +ve and -ve examples.
- **Entropy is between 0 and 1** if the collection contains unequal no. of +ve and -ve examples.

Information Gain

$\text{Gain}(S,A)$: expected reduction in entropy due to partitioning S on attribute A

$$\text{Gain}(S,A) = \text{Entropy}(S) - \sum_{v \in \text{values}(A)} |S_v|/|S| \text{Entropy}(S_v)$$

$$\begin{aligned}\text{Entropy}([29+, 35-]) &= -29/64 \log_2 29/64 - 35/64 \log_2 35/64 \\ &= 0.99\end{aligned}$$

Information Gain

$$\text{Entropy}([21+, 5-]) = 0.71$$

$$\text{Entropy}([8+, 30-]) = 0.74$$

$$\text{Gain}(S, A_1) = \text{Entropy}(S)$$

$$-26/64 * \text{Entropy}([21+, 5-])$$

$$-38/64 * \text{Entropy}([8+, 30-])$$

$$= 0.27$$

$$\text{Entropy}([18+, 33-]) = 0.94$$

$$\text{Entropy}([8+, 30-]) = 0.62$$

$$\text{Gain}(S, A_2) = \text{Entropy}(S)$$

$$-51/64 * \text{Entropy}([18+, 33-])$$

$$-13/64 * \text{Entropy}([11+, 2-])$$

$$= 0.12$$



Example

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

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

$$\begin{aligned} \text{Entropy } (S) &= -\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14} \\ &= 0.940 \end{aligned}$$

Age

$\text{Gain}(S, \text{Age})$

$$\text{Gain}(S, A) \equiv \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$$

$$= \text{Entropy}(S) - \frac{5}{14} \text{Entropy}([2+, 3-])$$

$$- \frac{4}{14} \text{Entropy}[4+, 0-]$$

$\frac{-5}{14} \text{Entropy}([2+, 3-])$

$$= 0.94 - \frac{5}{14} \left[\frac{-2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} \right]$$

$$+ \frac{4}{14} \left[\frac{-4}{4} \log_2 \frac{4}{4} \right] + \frac{5}{14} \left[\frac{-3}{5} \log_2 \frac{2}{5} - \frac{2}{5} \log_2 \frac{2}{1} \right]$$

$$= 0.94 - \frac{5}{14} [0.968] - \frac{4}{14} [0] - \frac{5}{14} [0.968]$$

$$= 0.94 - 0.36 = 0.58 - 0.36 = \underline{\underline{0.25}}$$

$$Gain(age) = 0.25$$

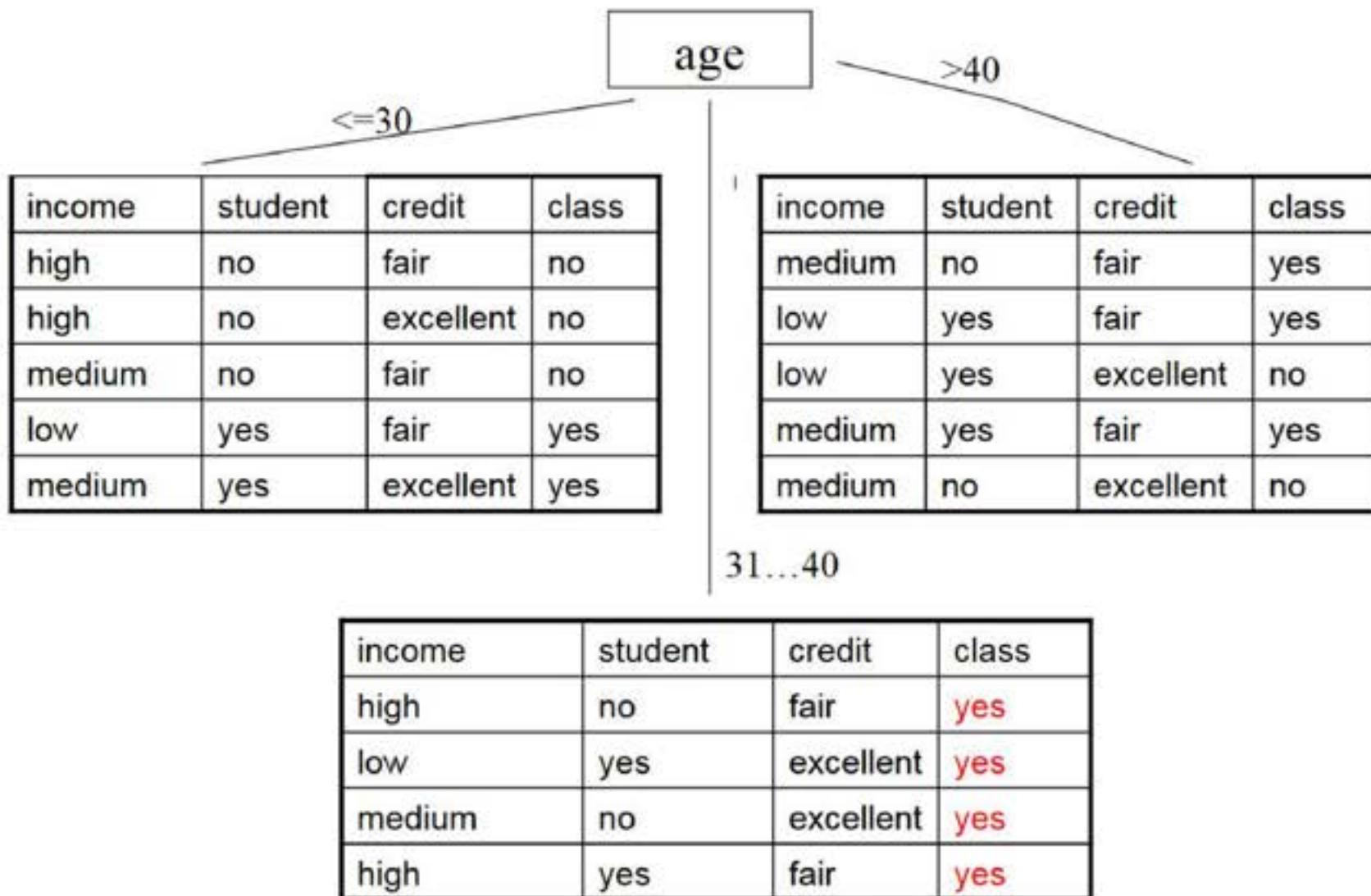
$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

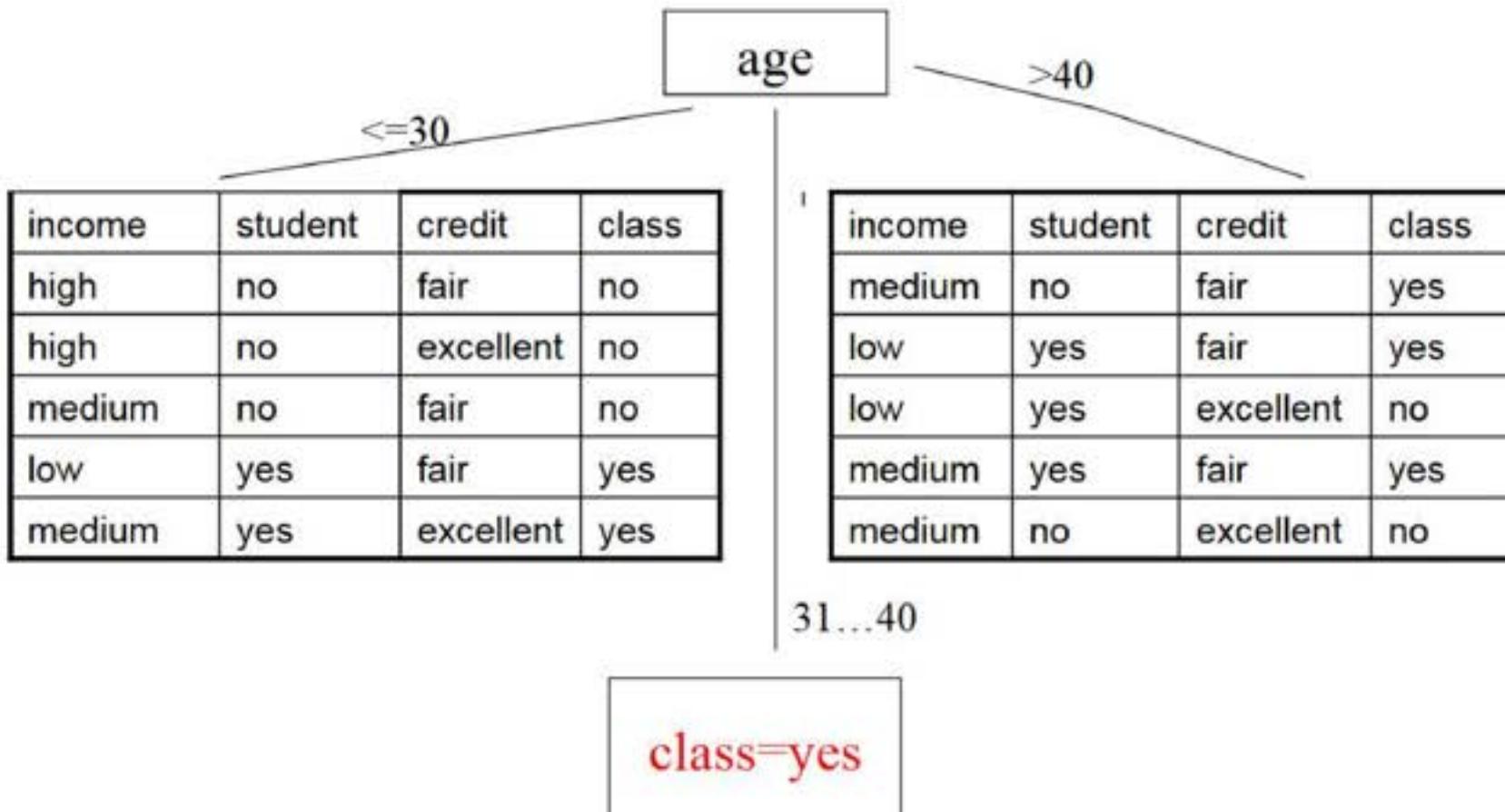
$$Gain(credit_rating) = 0.048$$

Age has maximum information gain. So age is selected as the best node to split . So age is selected as root node

Building The Tree: we choose “age” as a root



Building The Tree: “age” as the root



age			
income	student	credit	class
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

$S \rightarrow \text{age} \leq 30 [3^+, 2^-]$

$$E(S_{\text{age}}) = -\frac{3}{5} \log \frac{3}{5} - \frac{2}{5} \log \frac{2}{5}$$

$$= 0.968$$

$$\text{Gain}(S_{\text{age}}, \text{income}) = E(S_{\text{age}}) - \frac{2}{5} E([0^+, 2^-])$$

$$- \frac{2}{5} E([1^+, 1^-]) - \frac{1}{5} E([1^+, 0^-])$$

$$= 0.968 - 0 - \frac{2}{5} \left[\frac{1}{2} \log \frac{1}{2} + \frac{1}{2} \log \frac{1}{2} \right] - 0$$

$$\therefore 0.568$$

age			
<=30			
income	student	credit	class
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

Gain(S_1 , student)

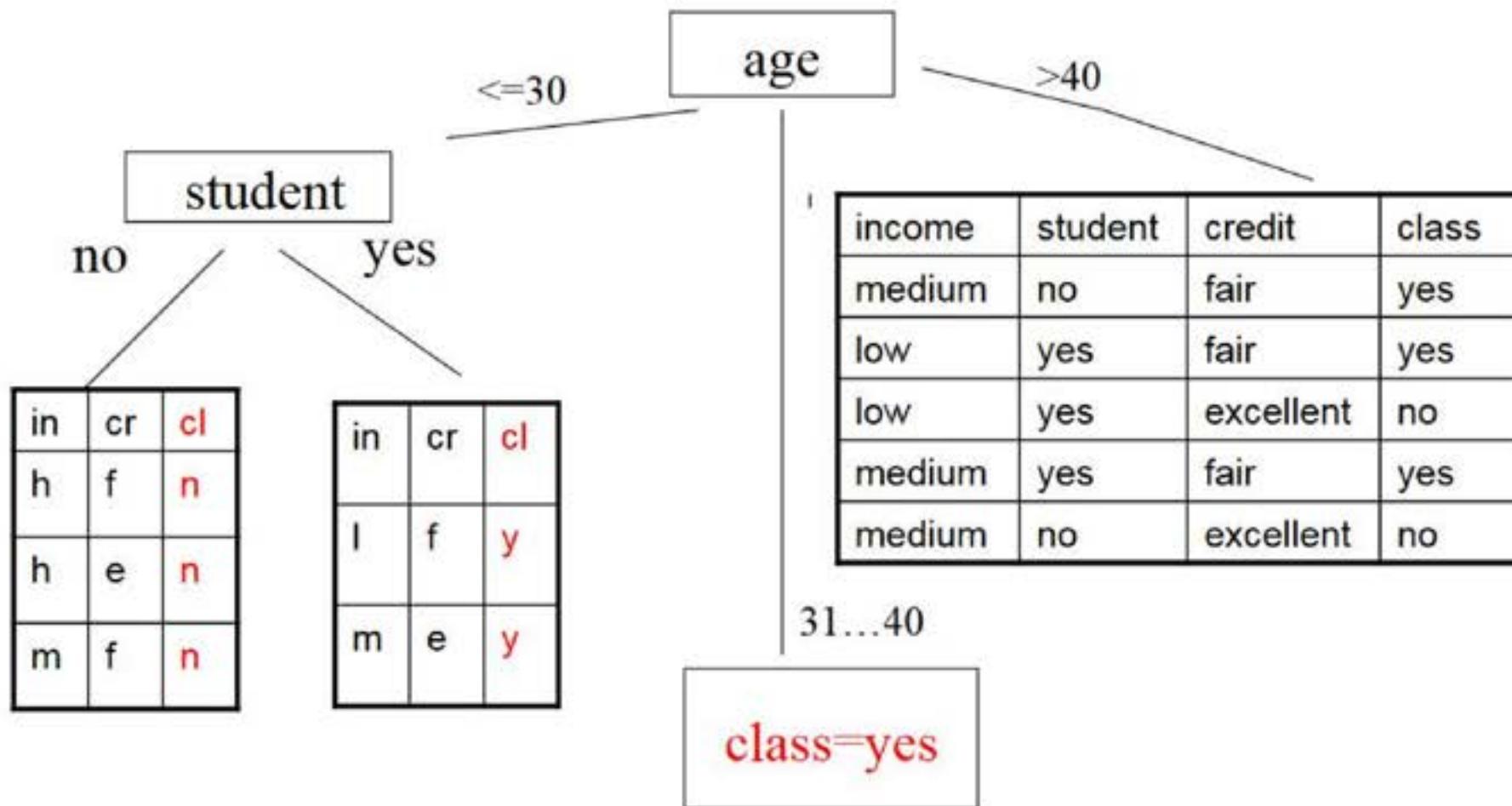
$$= 0.968 - \frac{3}{5} [E([0^+, 3^-])] - \frac{2}{5} [E([2^+, 0^-])] \\ = 0.968$$

Gain(S_2 , credit)

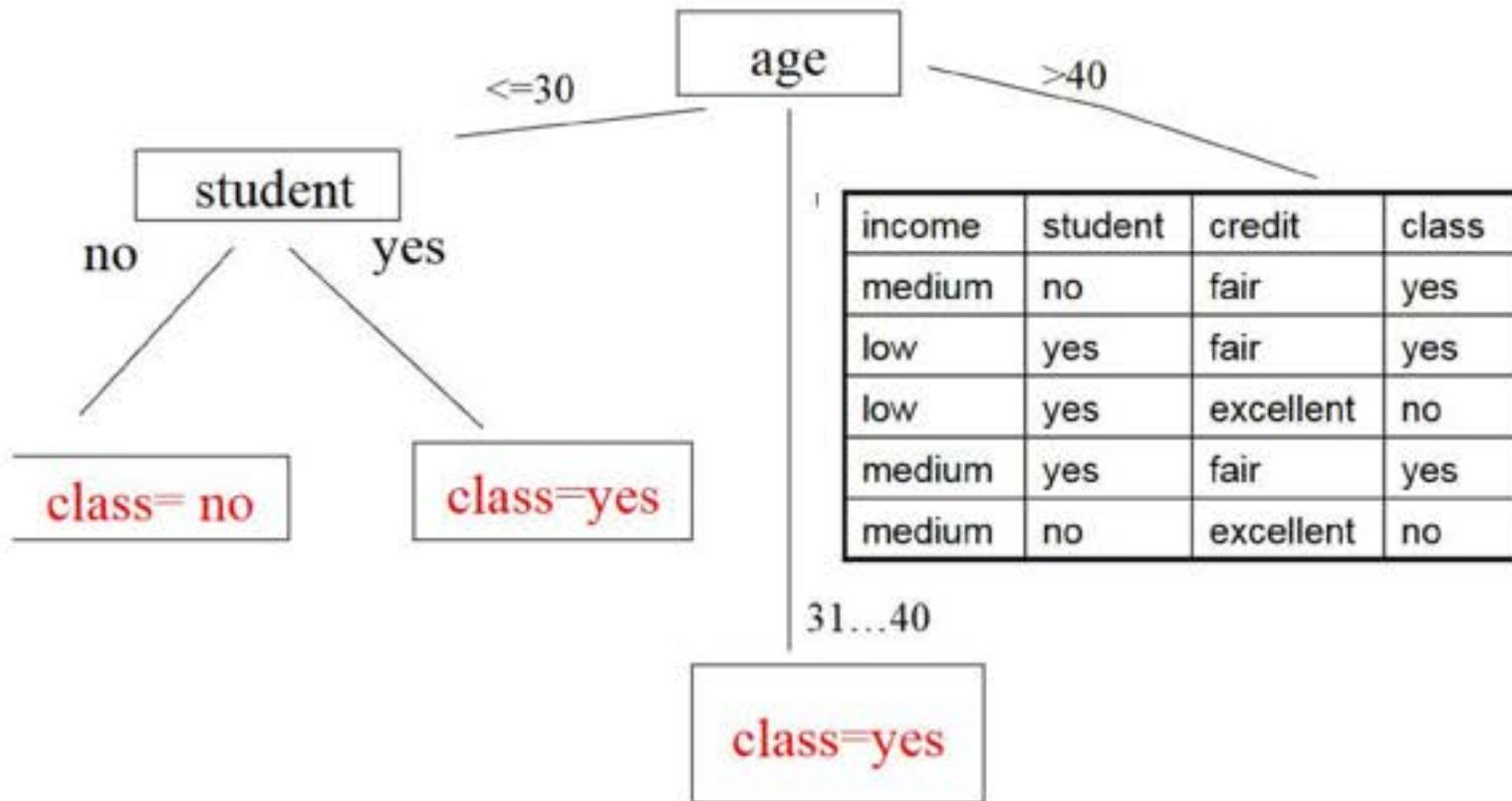
$$= 0.968 - \frac{3}{5} E([1^+, 2^-]) - \frac{2}{5} E([1^+, 1^-])$$

max for Student attribute

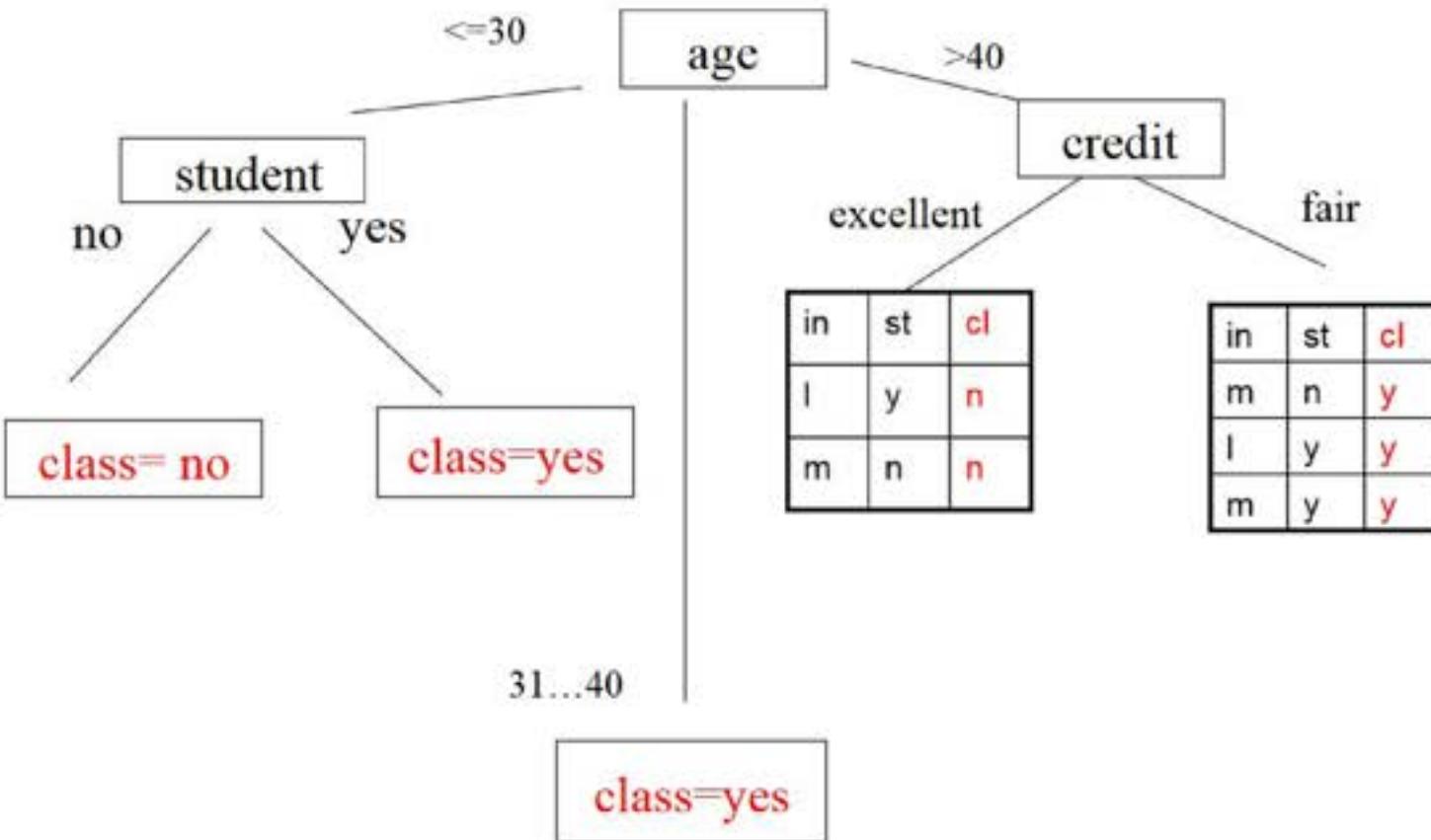
Building The Tree: we chose “student” on ≤ 30 branch



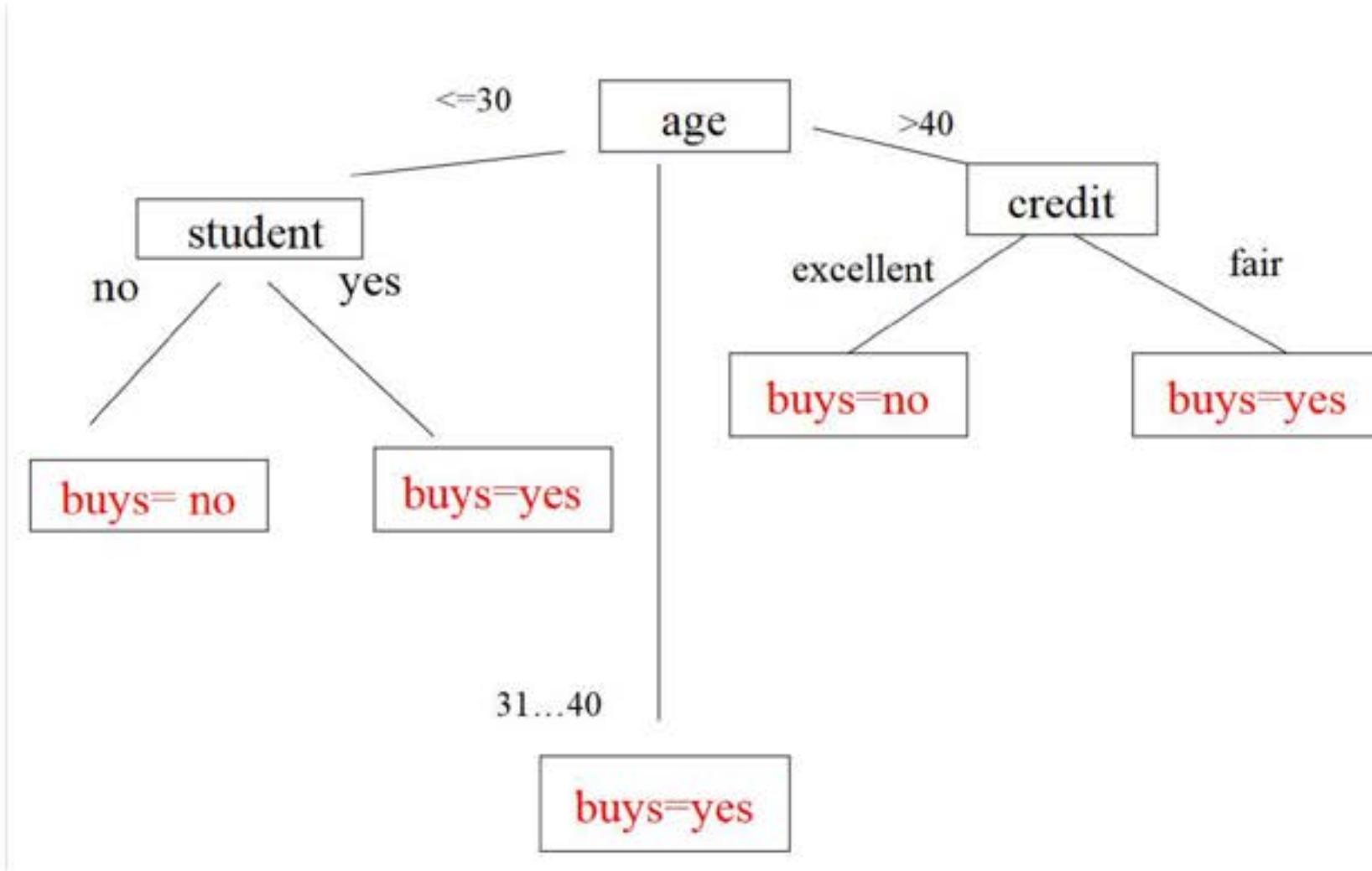
Building The Tree: we chose “student” on ≤ 30 branch



Building The Tree: we chose “credit” on >40 branch



Final tree



Rules extracted from the tree

- The rules are:

IF *age* = “ ≤ 30 ” AND *student* = “no” THEN
buys_computer = “no”

IF *age* = “ ≤ 30 ” AND *student* = “yes” THEN
buys_computer = “yes”

IF *age* = “31...40” THEN
buys_computer = “yes”

IF *age* = “ >40 ” AND *credit_rating* = “*excellent*” THEN
buys_computer = “no”

IF *age* = “ >40 ” AND *credit_rating* = “*fair*” THEN
buys_computer = “yes”

Inductive Bias

- Shorter trees are preferred over larger trees

Overfitting and Tree Pruning

- Overfitting: An induced tree may overfit the training data
 - Too many branches, some may reflect anomalies due to noise or outliers
 - Poor accuracy for unseen samples
- Two approaches to avoid overfitting
 - Prepruning: *Halt tree construction early*-do not split a node if this would result in the goodness measure falling below a threshold
 - Difficult to choose an appropriate threshold
 - Postpruning: *Remove branches* from a “fully grown” tree—get a sequence of progressively pruned trees
 - Use a set of data different from the training data to decide which is the “best pruned tree”

Attribute Selection Measures

- **Information gain:**
 - biased towards multivalued attributes
- **Gain ratio:**

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$$

$$SplitInformation(S, A) \equiv - \sum_{i=1}^c \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

- tends to prefer unbalanced splits in which one partition is much smaller than the others

- **Gini index:**

$$gini(D) = 1 - \sum_{j=1}^n p_j^2$$

- where p_j is the relative frequency of class j in D
- Choose attribute with low gini index
- has difficulty when # of classes is large
- tends to favor tests that result in equal-sized partitions and purity in both partitions

Decision tree suited when -

- Instances are represented by attribute-value pairs
- The target function has discrete output values
- The training data may contain errors.
- The training data may contain missing attribute values

Thank you !!!!!



Machine Learning (19CSE305)

Bayes Theorem, Naïve Bayes, BBN



Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Axioms of Probability
- Bayes Theorem
- Naïve Bayes
- BBN

Example – Handwriting Quality Analysis

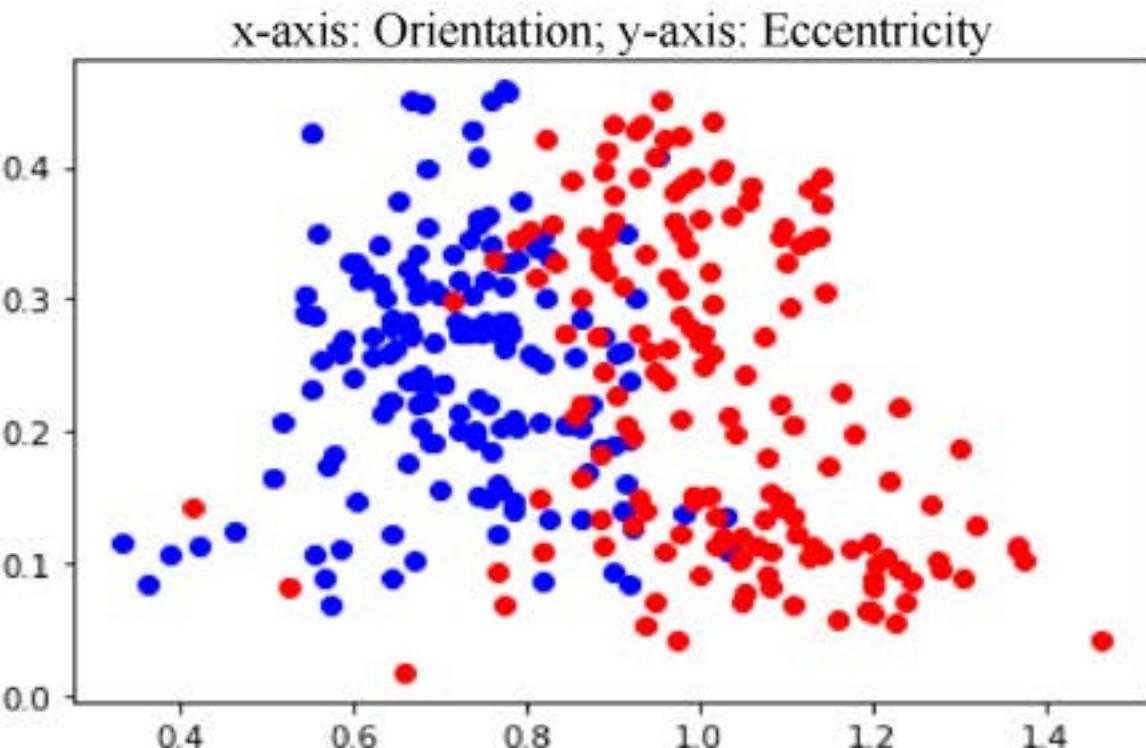
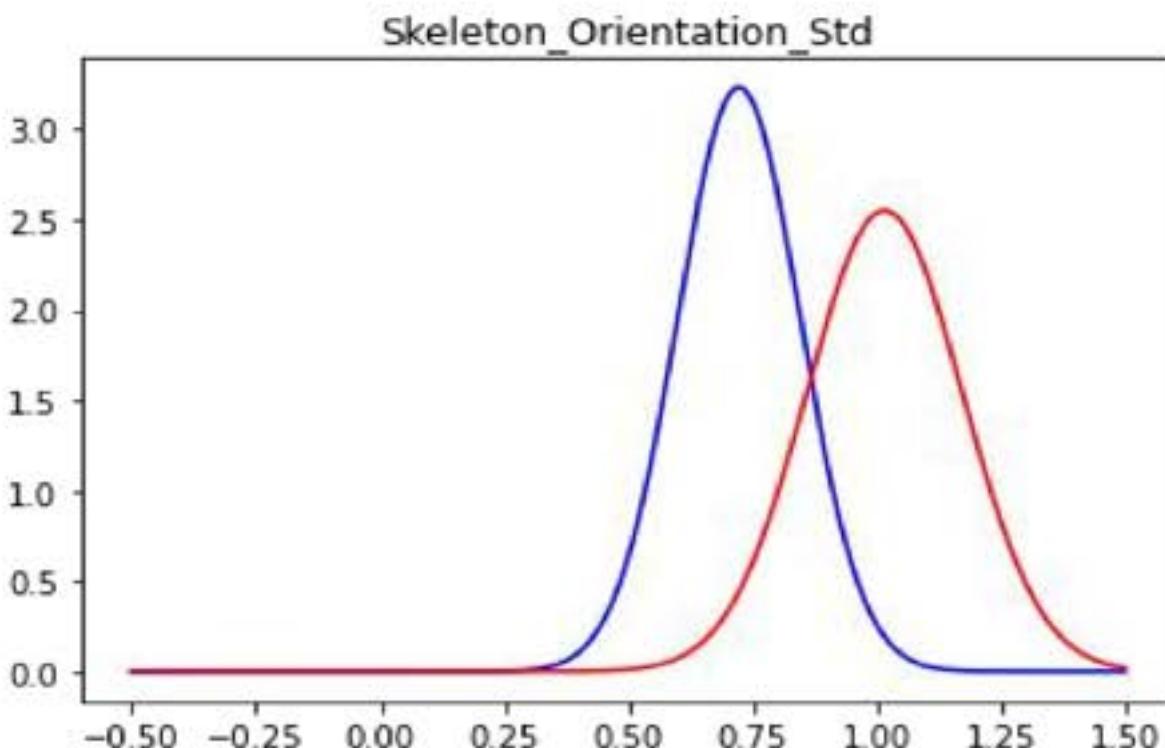
Software does not wear out only hardware wears out.
1. Software gets deteriorated ; A software frequently gets updated and new versions of the software are available. people shift to the new versions leaving behind the old versions therefore the software gets deteriorated.



→ It first executes the statements under do and then checks the condition, if it's true it runs the do part again else it exits the loop.



Example – Handwriting Quality Analysis



Given a value of data, it may not be possible to decisively arrive at the outcome.

The data value may have different probabilities associated with classes. This is the foundation of Bayesian classifier.

3 Axioms of Probability

$$0 \leq P(A) \leq 1$$

$$1 = \sum_{k=0}^n P(A_k)$$

$$P(A_1) + P(A_2) - P(A_1 \cap A_2) = P(A_1 \cup A_2)$$

Conditional Probability, Bayes Theorem

$$P(A_1|A_2) = \frac{P(A_1, A_2)}{P(A_2)}$$

$$\Rightarrow P(A_1, A_2) = P(A_1|A_2) * P(A_2)$$

$$\Rightarrow P(A_2, A_1) = P(A_2|A_1) * P(A_1)$$

$$P(A_2, A_1) = P(A_1, A_2)$$

$$P(A_2|A_1) = \frac{P(A_1, A_2)}{P(A_1)} = \frac{P(A_1|A_2) * P(A_2)}{P(A_1)}$$

← Bayes Theorem

Bayes Classifier

Software does not wear out only hardware wears out
, software gets deteriorated ; A software frequently gets updated and new versions of the software are available. people shift to the new versions leaving behind the old versions therefore the software gets deteriorated.

$\mathbf{X} \rightarrow$ features; $y \rightarrow$ classes

Class Conditional Density

Prior probability

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

Posterior probability

$\mathbf{X} =$ {area,
convex area,
bounding box area,
mean intensity,
extent,
solidity,
eccentricity,
orientation}

FEATURES

$y = \{\text{good}, \text{bad}\}$

CLASSES

Bayes Classifier Terminologies

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

Posterior probability

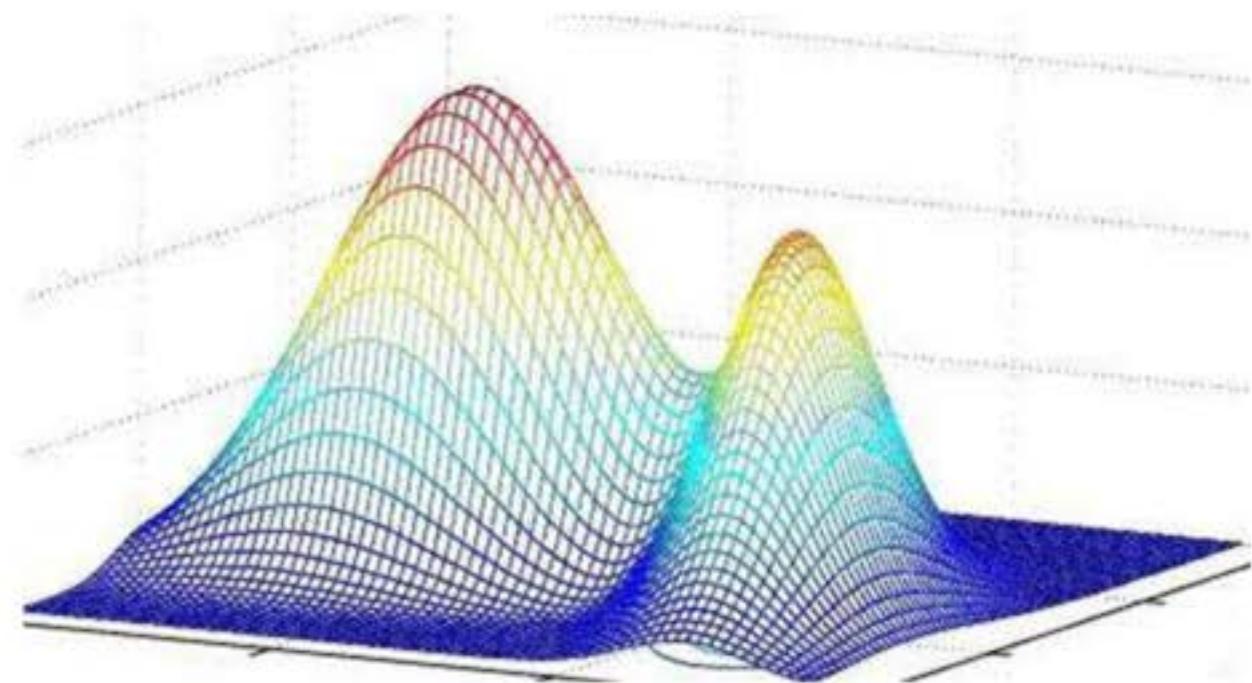
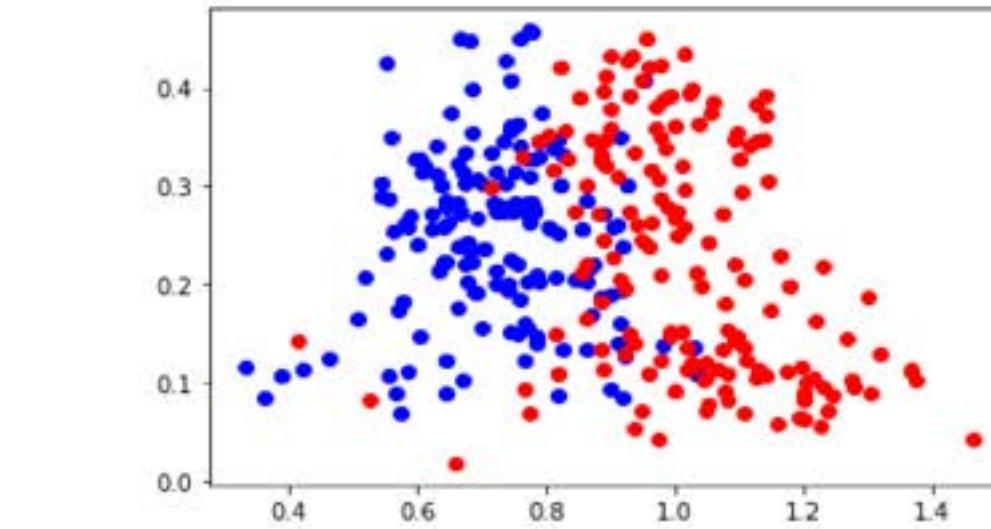
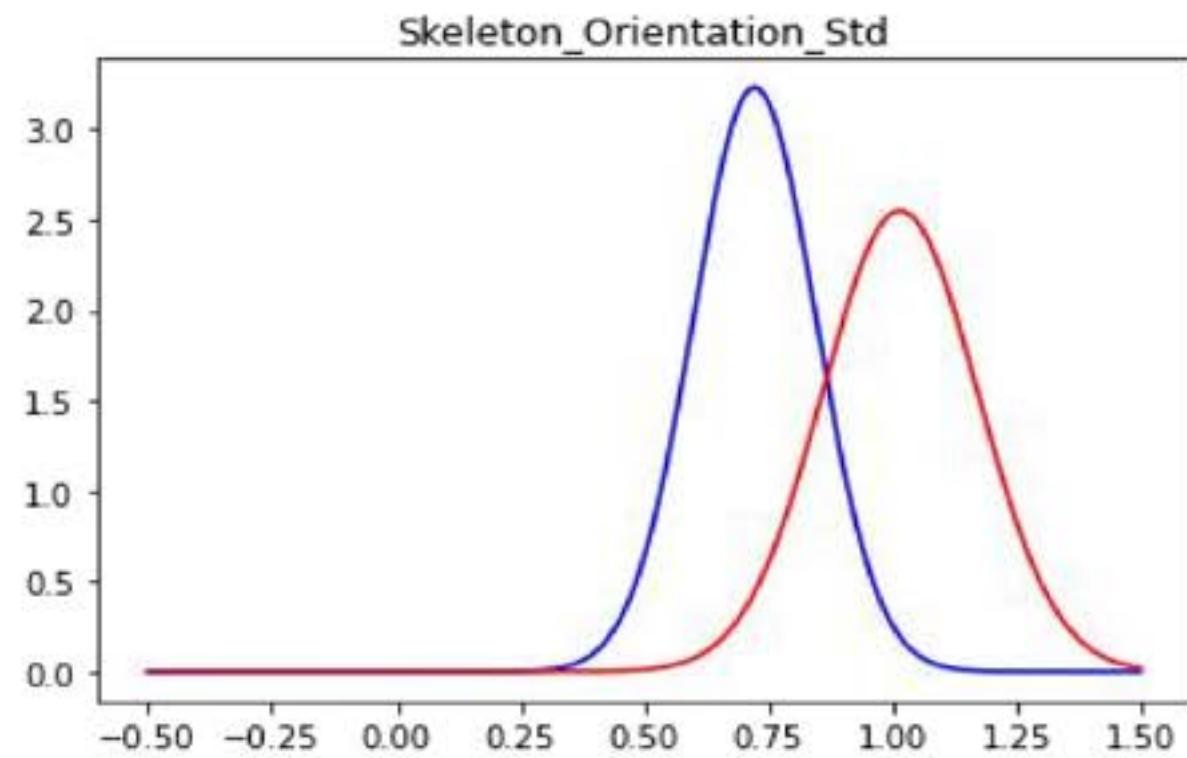
Class Conditional Density

Prior probability

$$y_{MAP} = \underset{y_j}{\operatorname{argmax}} \{P(y_j | \mathbf{X})\}$$

y_j are different classes

Distribution functions



Source: DOI: 10.13053/cys-24-3-3326

Another Example

Tid	Refund	Marital Status	Taxable Income	Evade
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

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

$\mathbf{X} \rightarrow [\text{age}, \text{income}, \text{student}, \text{credit_rating}]$
 $y \rightarrow \text{buys_computer } \{\text{yes,no}\}$

$\mathbf{X} \rightarrow [\text{refund}, \text{marital status}, \text{income}]$

$y \rightarrow \text{evade } \{\text{yes,no}\}$

$$P(y | \mathbf{X}) = \frac{P(\mathbf{X} | y) P(y)}{P(\mathbf{X})}$$

Difficulty for density estimation

Key differentiator. Estimation of this determines the outcome.

$$P(y | \mathbf{X}) = \frac{P(\mathbf{X} | y) P(y)}{P(\mathbf{X})}$$

This is a fixed term for all classes; so may be ignored as has no real impact

Assuming sample (training) set is representative of actual scenario, this may be calculated as the ratio between class representation and numbers of samples available in the set

Requires large amount of data for its estimation. Most scenarios we don't have that amount of training data.

Density Estimation

- 2 Techniques – Naïve Bayes & Belief Networks
- Naïve Bayes
 - $\mathbf{X} = \{X_1, X_2, \dots, X_n\}$
 - Assumes X_i is independent of X_j when $i \neq j$
 - estimation along each dimension possible with lower data samples

$$P(\mathbf{X}) = P(X_1) P(X_2) \dots P(X_n)$$

$$P(y | \mathbf{X}) = \frac{P(\mathbf{X} | y) P(y)}{Const} = \frac{P(X_1 | y) P(X_2 | y) \dots P(X_n | y) P(y)}{Const}$$

Exercise – Discrete case

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

X1 = [>40 , medium, yes, fair]



Microsoft Excel
Worksheet

$C * P(y = \text{yes} | \mathbf{X1})$

$\rightarrow P(\mathbf{X1} | y = \text{yes}) P(y = \text{yes})$

$\rightarrow P(x_1 = >40 | y = \text{yes}) * P(x_2 = \text{medium} | y = \text{yes}) * P(x_3 = \text{yes} | y = \text{yes}) * P(x_4 = \text{fair} | y = \text{yes}) * P(y = \text{yes})$

$\rightarrow 3/9 * 4/9 * 6/9 * 6/9 * 9/14$

X = {age, income, student, credit_rating}

Exercise – Discrete case what if ...?

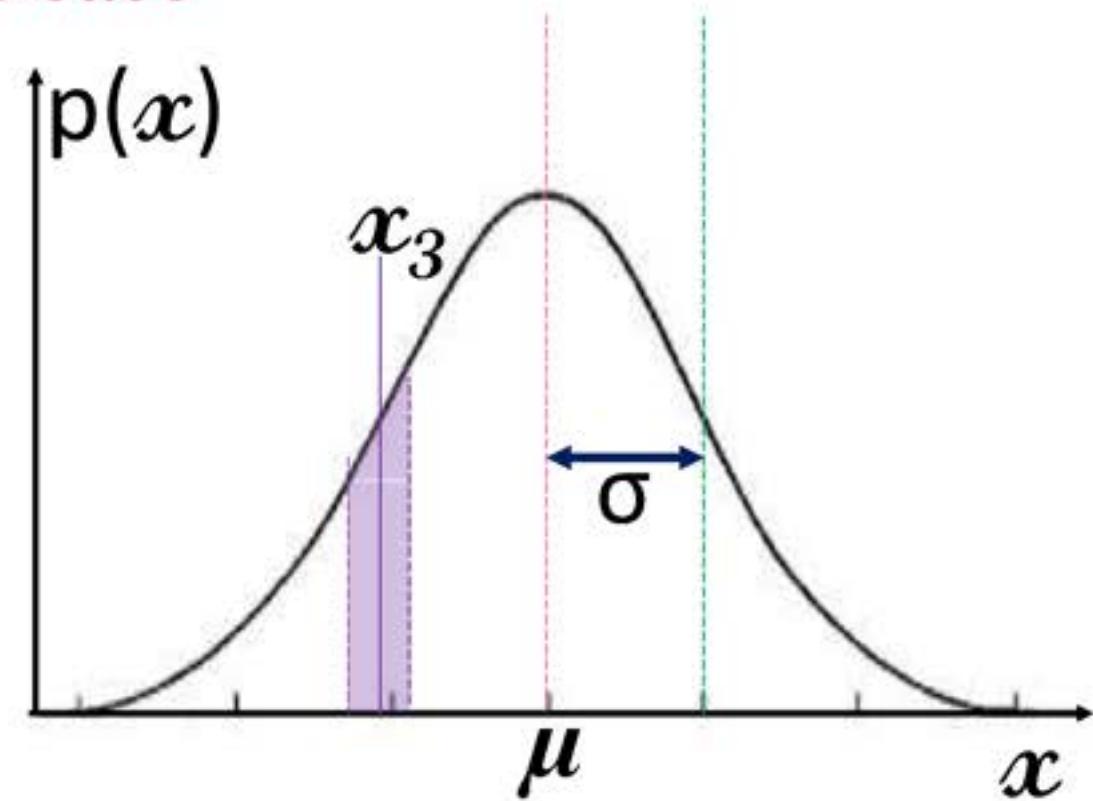
age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes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

$$\mathbf{X_1} = [30-40, \text{medium}, \text{yes}, \text{fair}]$$

- Use prior probability for the data point
- Use m-estimate formulation $\rightarrow P(x_i / y_j) = (n_c + mp) / (n + m)$

Density estimation for continuous case

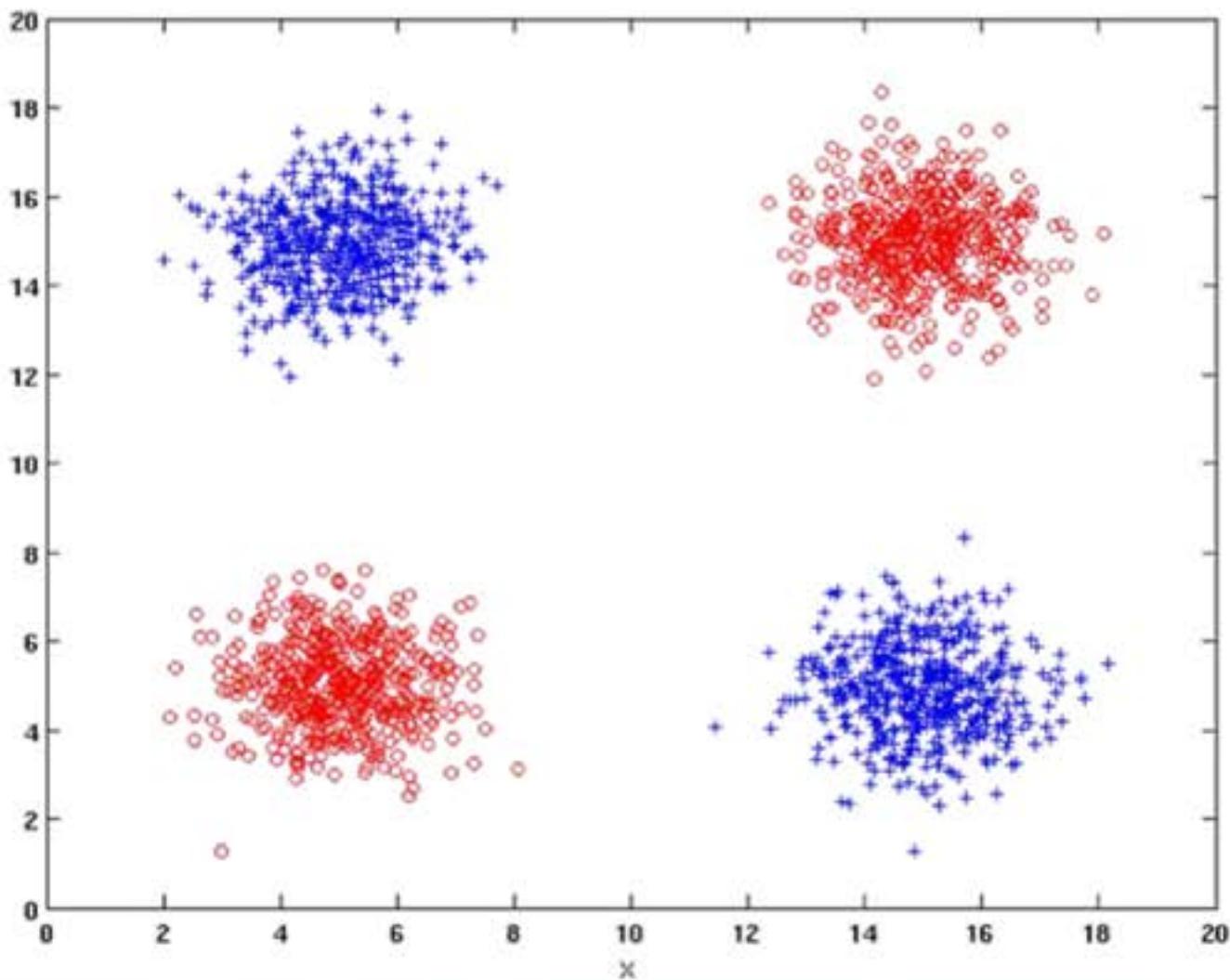
Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Home Assignment:

Find the class for $\mathbf{X} = [\text{Refund} = \text{No},$
 $\text{Marital status} = \text{married},$
 $\text{Taxable Income} = 120\text{K}]$

Naïve-Bayes – Independence is the key



Naïve-Bayes – Summary

- Robust to outlier observations
- Robust to irrelevant attributes
- Redundant and correlated attributes will violate class conditional assumption
 - Use other techniques such as Bayesian Belief Networks (BBN)

Content Reference: Introduction to Data Mining by Tan, Kumar & Steinbach

Thank you !!!!!



Machine Learning (19CSE305)

Error Surface & Parameter Optimization

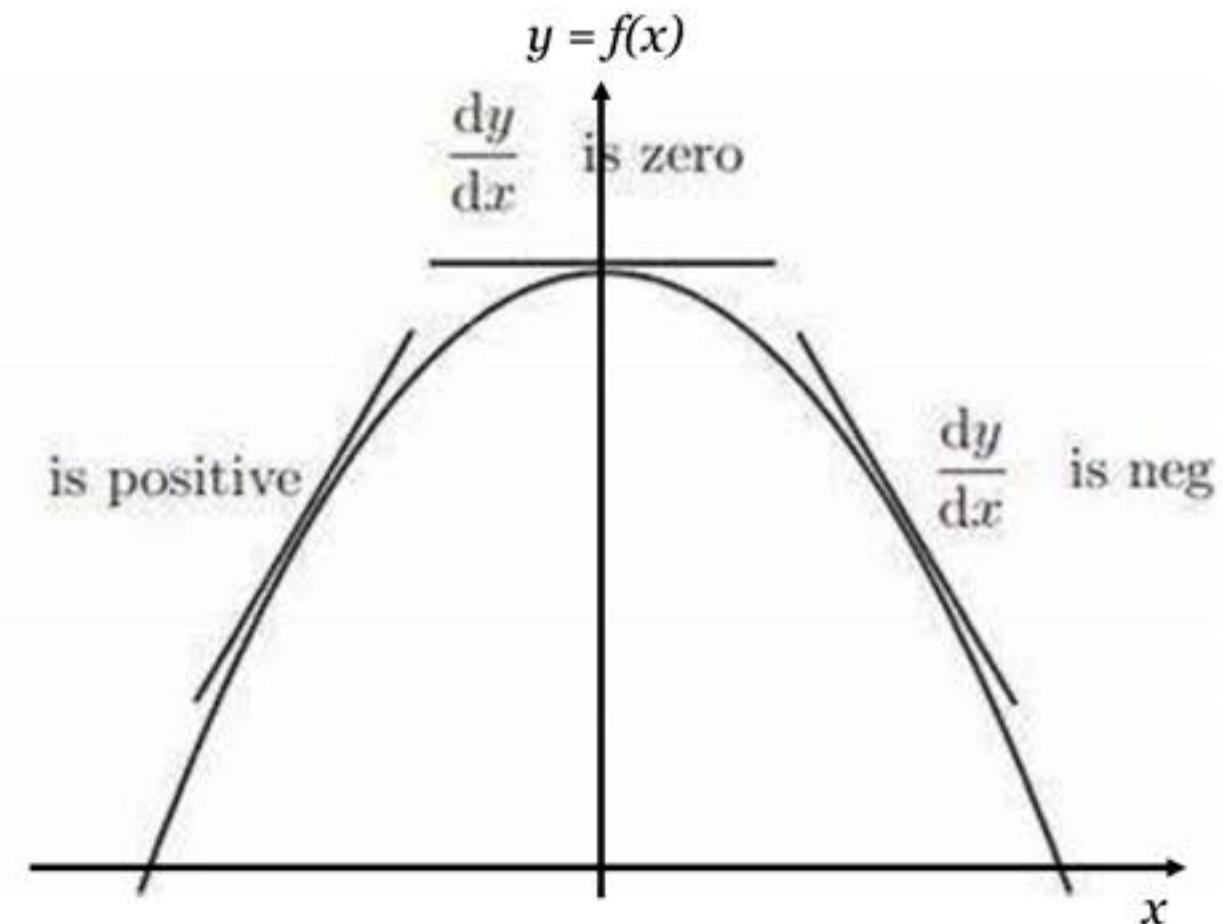
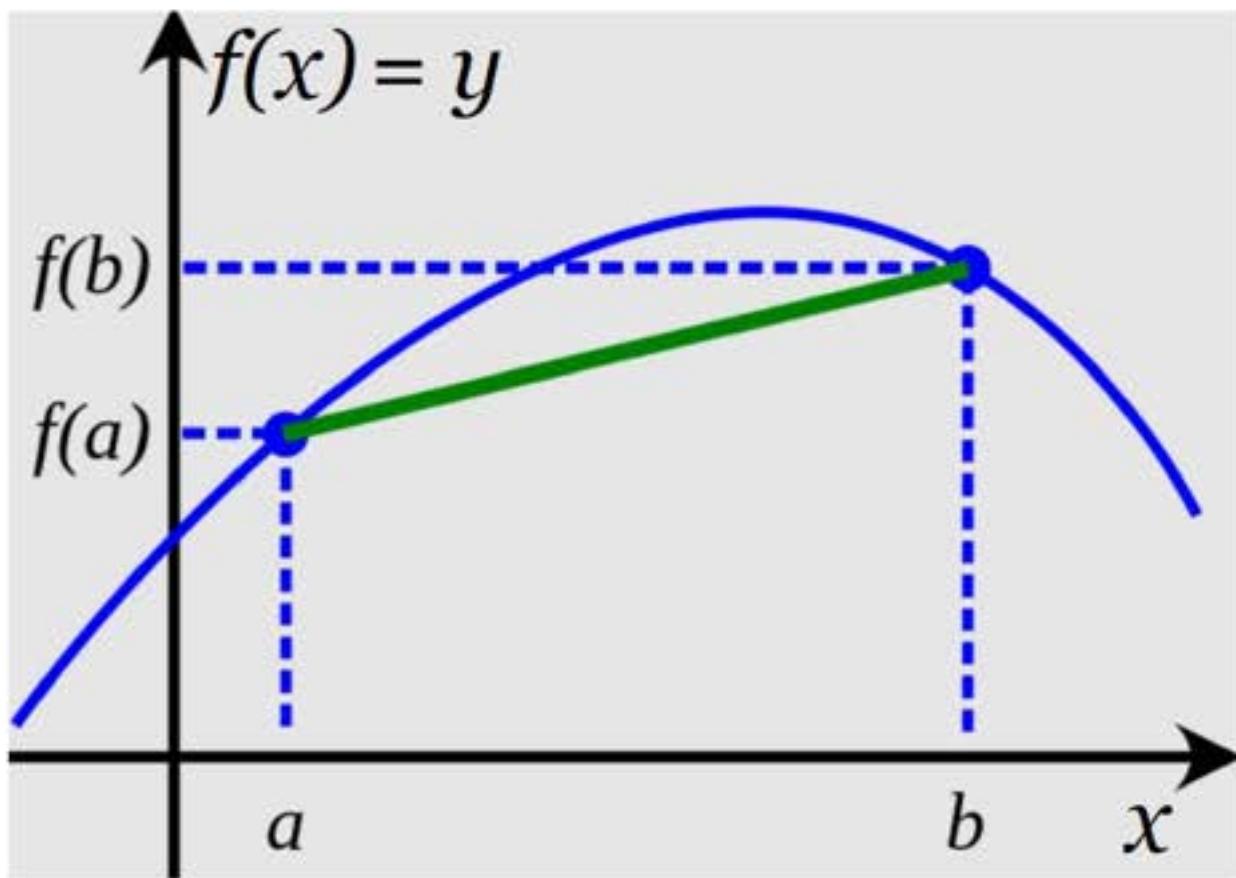


Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

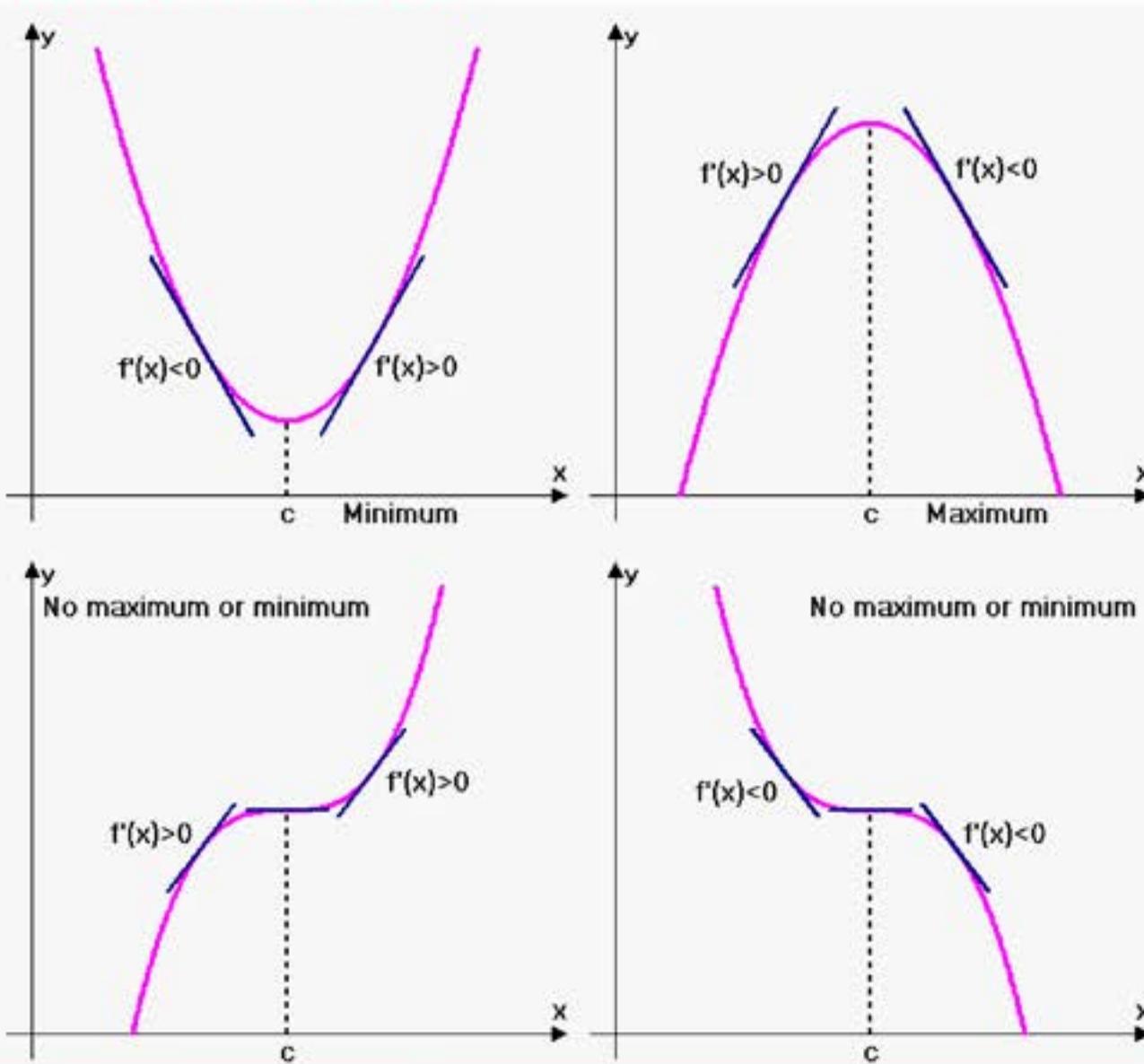
Topics

- Parameter space and error surface
- Optimal points
- Algorithms
 - Brute force
 - Gradient Descent
- Quiz

Functions and Derivatives



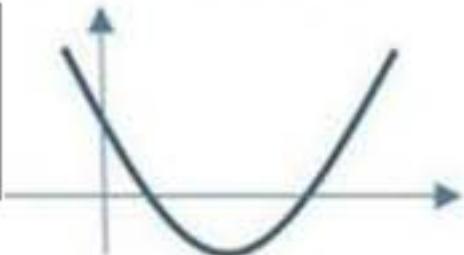
Functions and Derivatives



Source: Internet

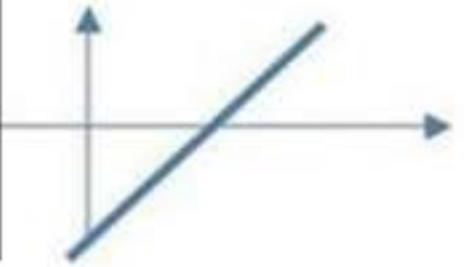
Functions and Derivatives

Function



$$y = (x - 1) * (x - 3)$$
$$= x^2 - 4x + 3$$

1st Derivative

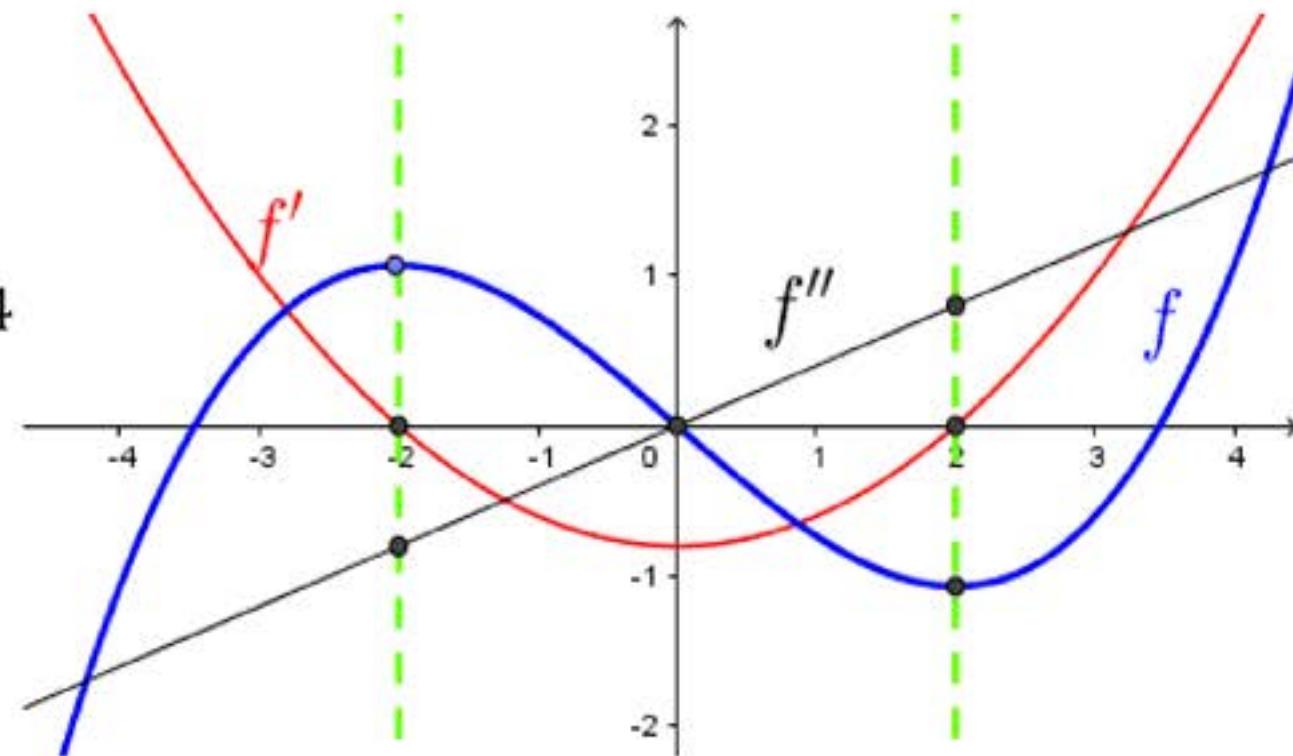


$$\frac{dy}{dx} = y' = 2x - 4$$

2nd Derivative

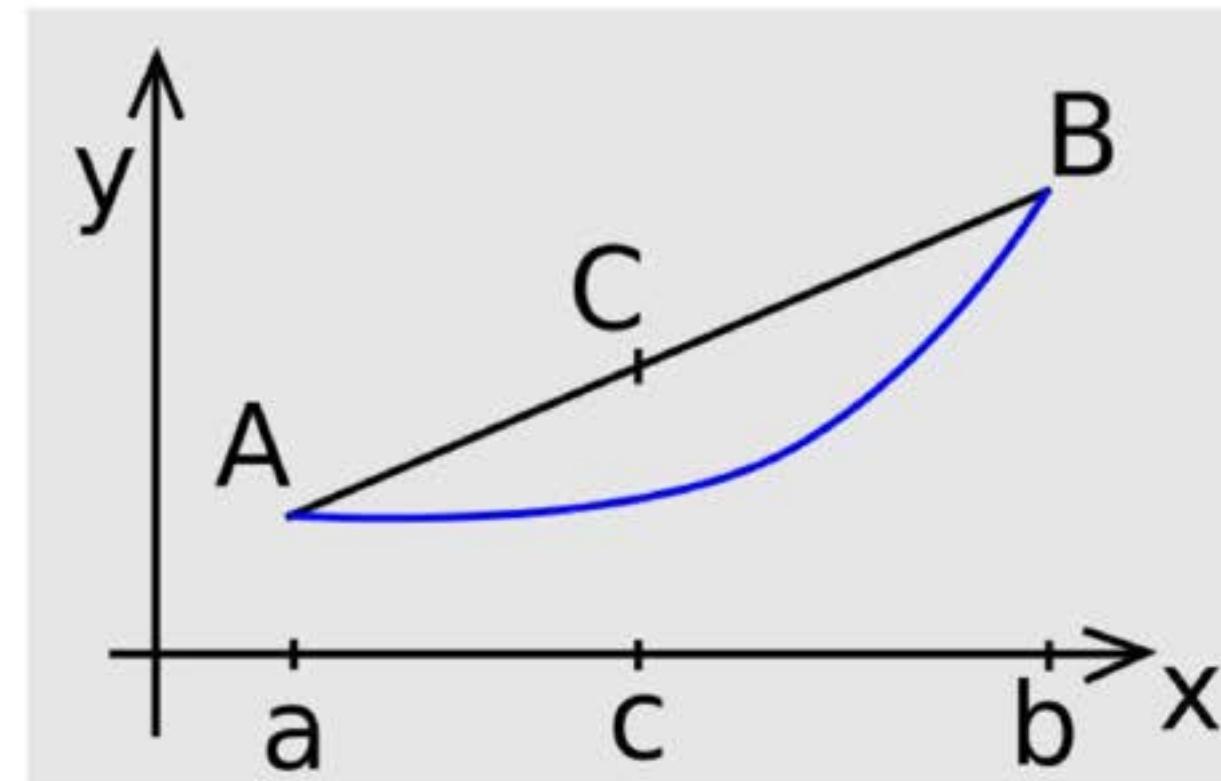
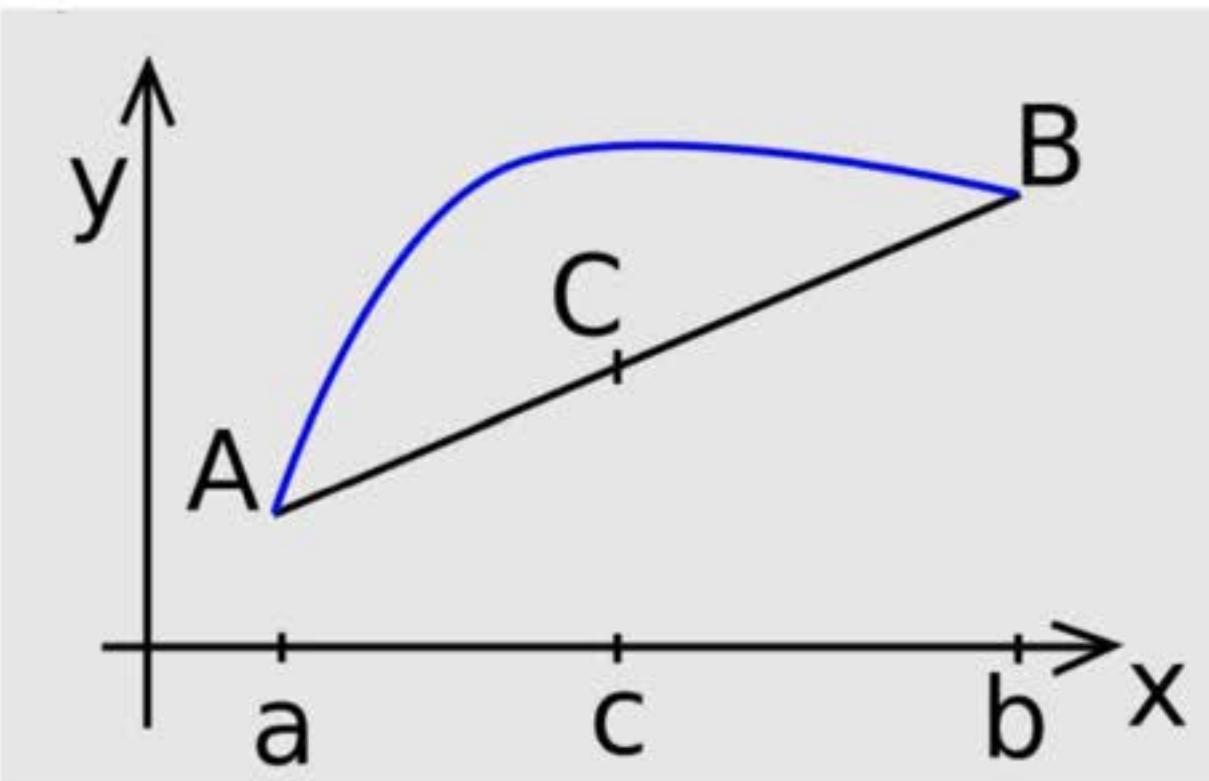


$$\frac{dy'}{dx} = \frac{d^2y}{dx^2} = 2$$

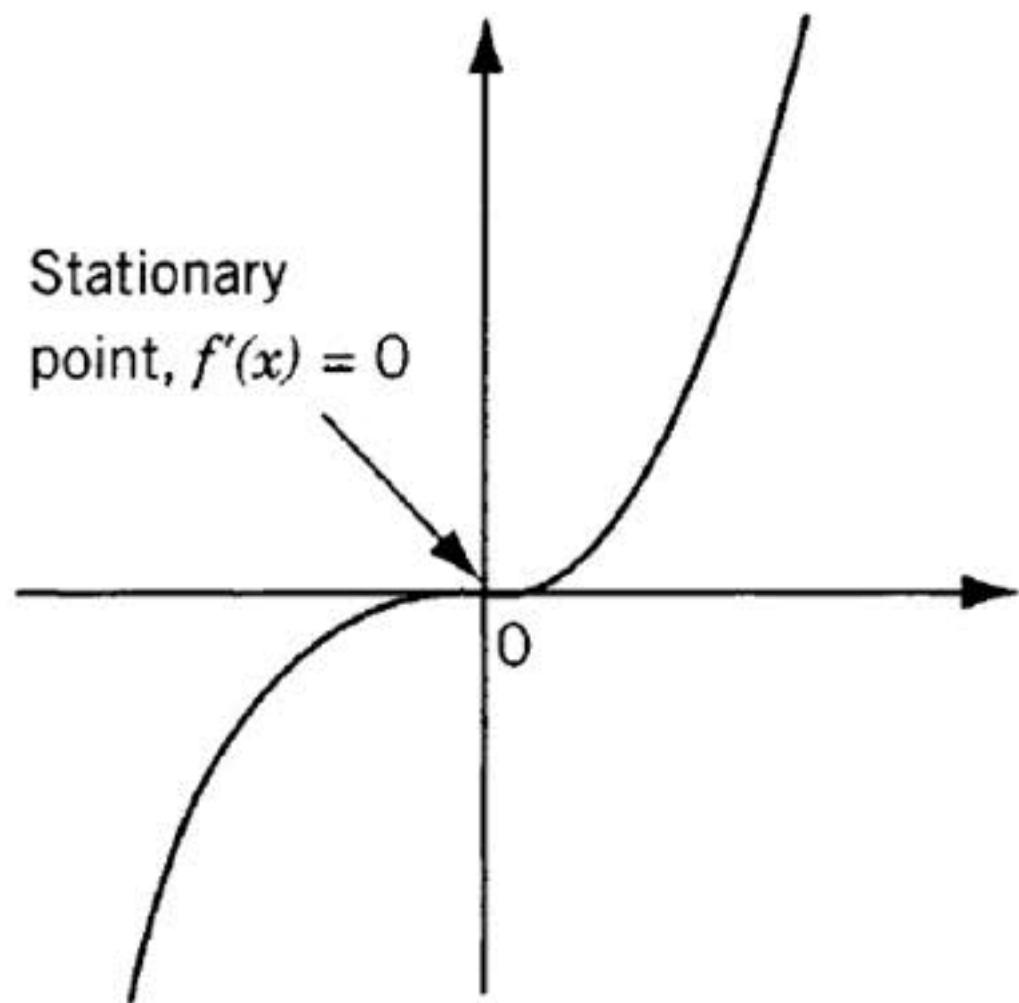
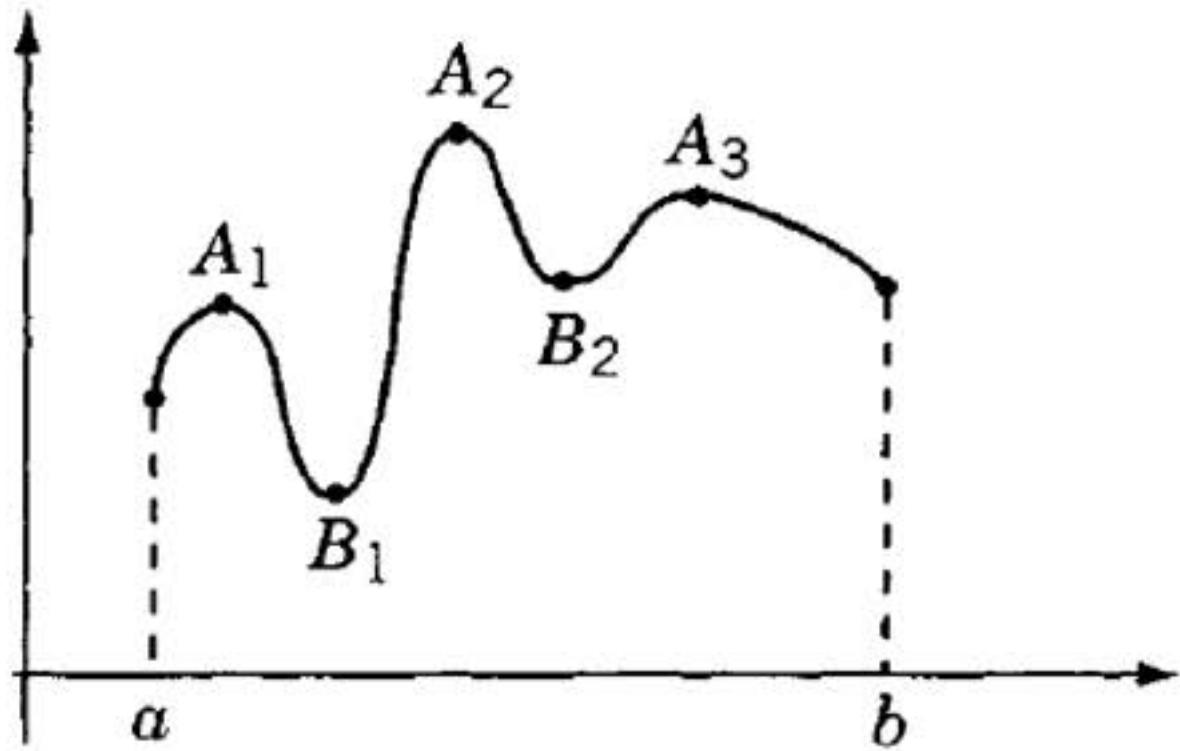


Source: www.analyzemath.com/

Concave and Convex Functions



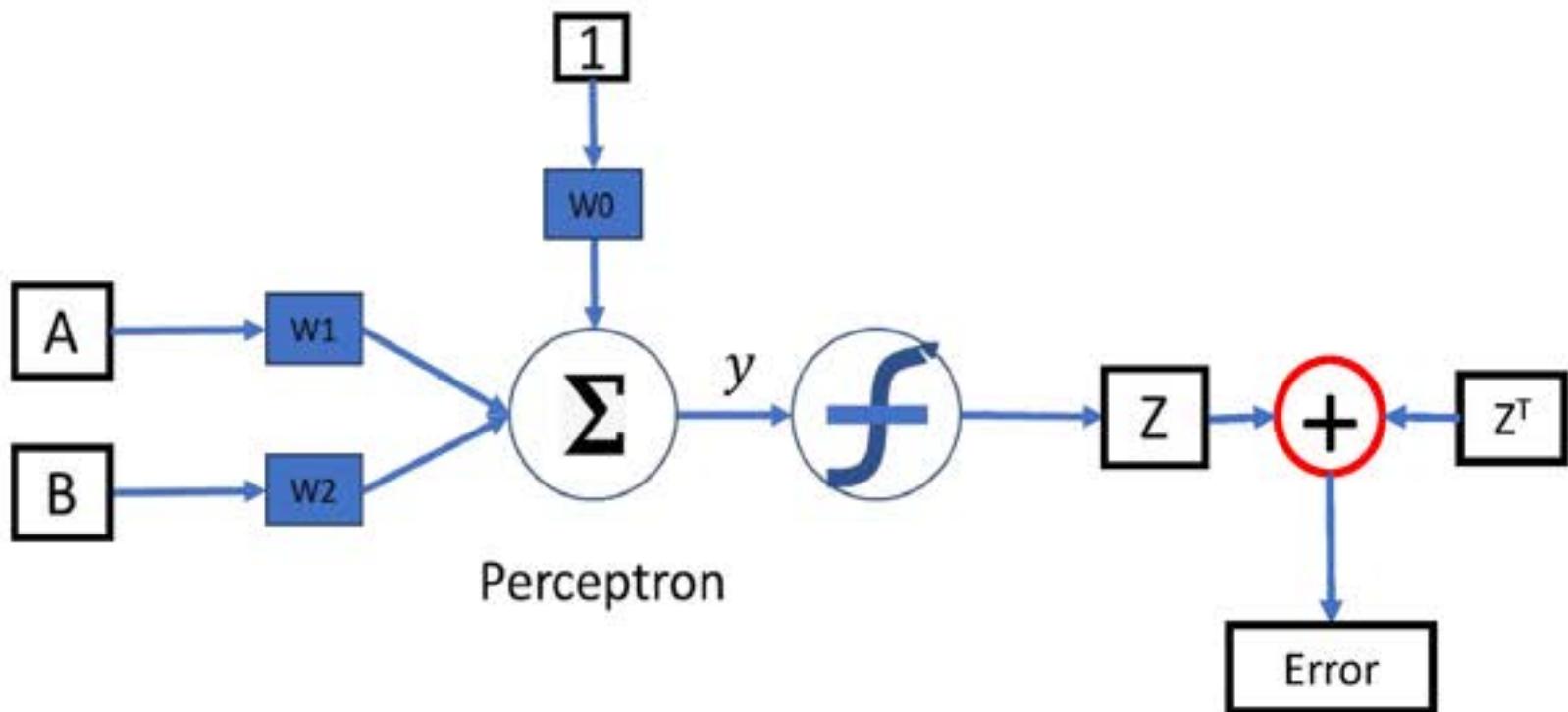
Local & Global – Minimum, Maximum; Saddle point



Source: Engineering Optimization, S S Rao

Estimation Error

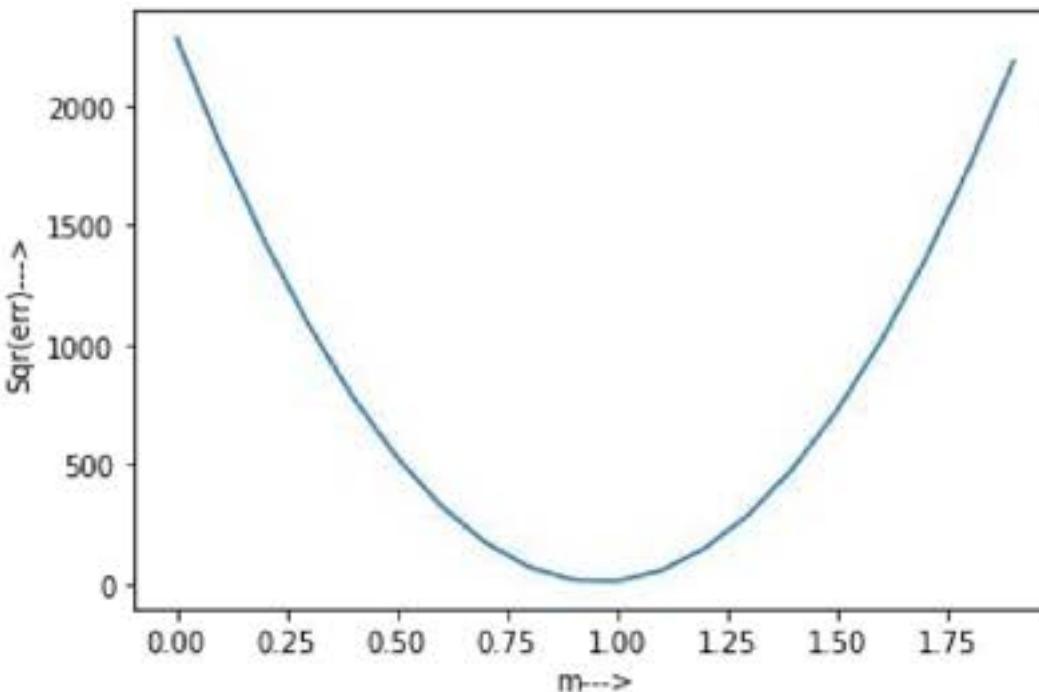
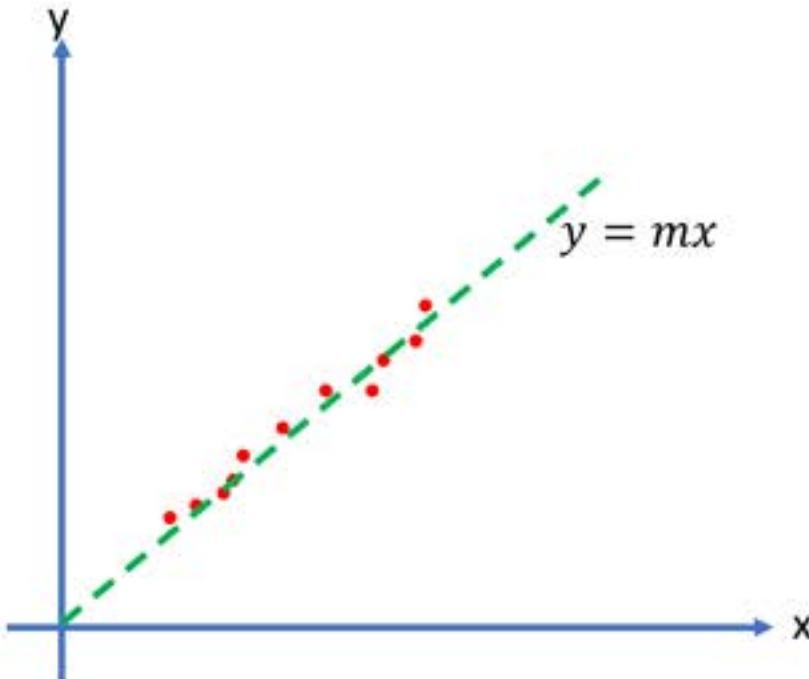
- $X \rightarrow$ Input vector
- $Z \rightarrow$ Estimation Output
- $Z^T \rightarrow$ Target Output
- $E \rightarrow$ Error



Aim of learning is to find the right set of parameters (weights in this case) such that error is minimal.

This is an optimization problem.

Single Parameter

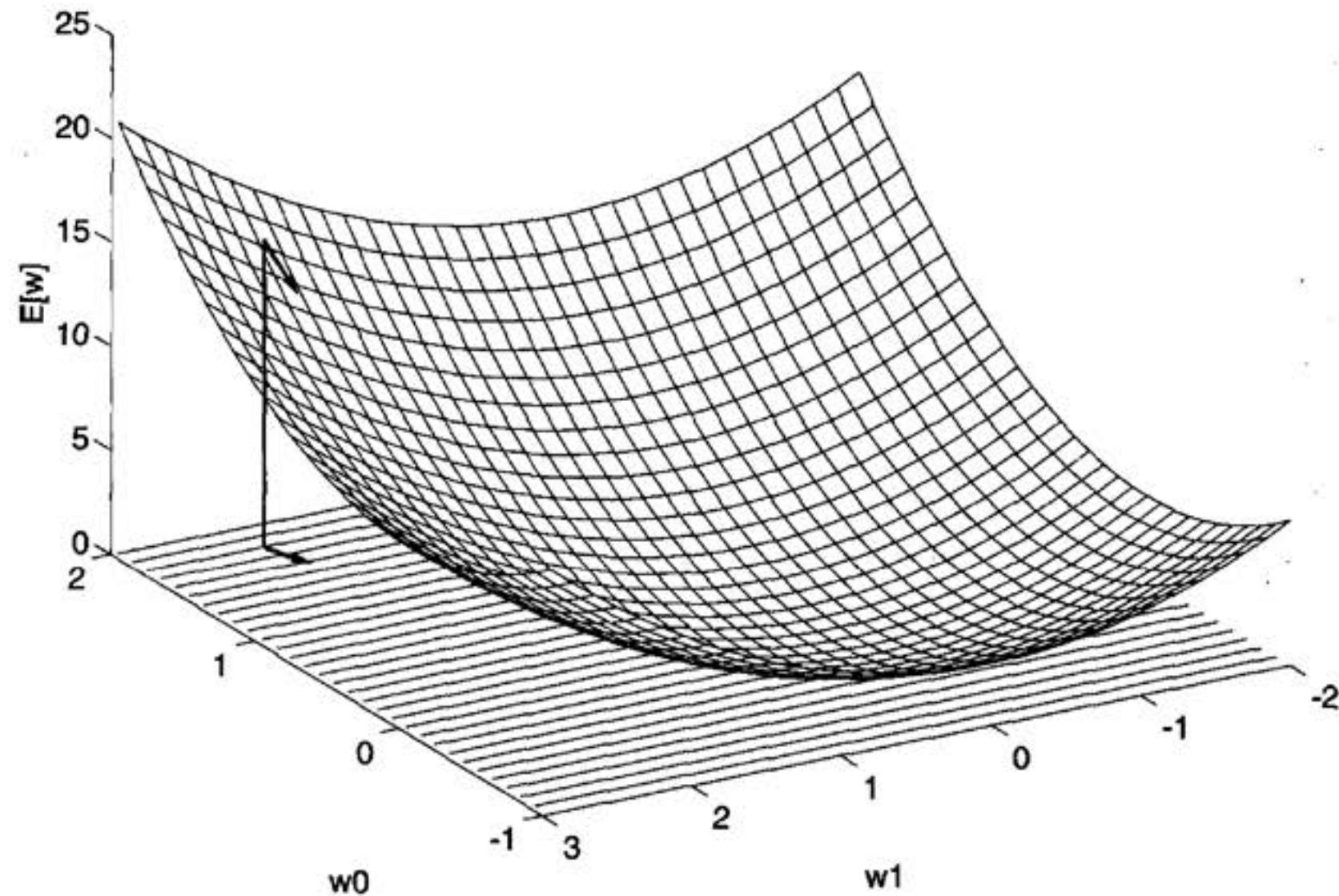


m is the parameter that we try to optimize such that error is less

How does the error graph look like? How can we find the point of minimal error?

When $y = mx + c$; m & c become 2 parameters to be optimized for.

Error space for parameters in \mathbb{R}^2



Brute Force or Exhaustive Search

- Estimate error for all variations of all parameters (weights & bias)
- Find the global minimum value
- Search is performed over the entire range of set limits
- Huge computational cost for the exercise; however, guaranteed outcome

Derivation of Gradient Descent Algorithm

$$o(\vec{x}) = \vec{w} \cdot \vec{x}$$

$$E(\vec{w}) \equiv \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$$

$$\vec{w} \leftarrow \vec{w} + \Delta \vec{w}$$

$$\Delta \vec{w} = -\eta \nabla E(\vec{w})$$

$$\nabla E(\vec{w}) \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \dots, \frac{\partial E}{\partial w_n} \right]$$

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

$$\begin{aligned}\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i} \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2 \\&= \frac{1}{2} \sum_{d \in D} \frac{\partial}{\partial w_i} (t_d - o_d)^2 \\&= \frac{1}{2} \sum_{d \in D} 2(t_d - o_d) \frac{\partial}{\partial w_i} (t_d - o_d) \\&= \sum_{d \in D} (t_d - o_d) \frac{\partial}{\partial w_i} (t_d - \vec{w} \cdot \vec{x}_d) \\ \frac{\partial E}{\partial w_i} &= \sum_{d \in D} (t_d - o_d) (-x_{id}) \\ \Delta w_i &= \eta \sum_{d \in D} (t_d - o_d) x_{id}\end{aligned}$$

Notes on Gradient Descent Algorithm

- Error is summed over all inputs and then the weights are updated
- Linear (pass through) activation function is used $\rightarrow f(x) = x$
- Assumes a convex error space
- If there are local minima, the search may get stuck and never come-out
- Since error is summed over all inputs, the convergence may be slow
- Incremental or stochastic gradient descent is a variation of the algorithm
 - Weight update done with each input
 - Sometimes helps in overcoming the local minima
- The same principle can be applied with other activation functions as well. However, mathematical proof of convergence may be difficult.

Other optimization techniques

- Genetic algorithm based approaches
- Evolutionary computation techniques
- Tabu Search
- Simulated Annealing
- Hill Climbing techniques
- Ant colony optimization
- Particle swarm optimization
- Random forest optimization

Thank you !!!!!



Machine Learning (19CSE305)

Linear regression



**Dr. Peeta Basa Pati
Ms. Priyanka V**
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Linear regression
- Logistic Regression

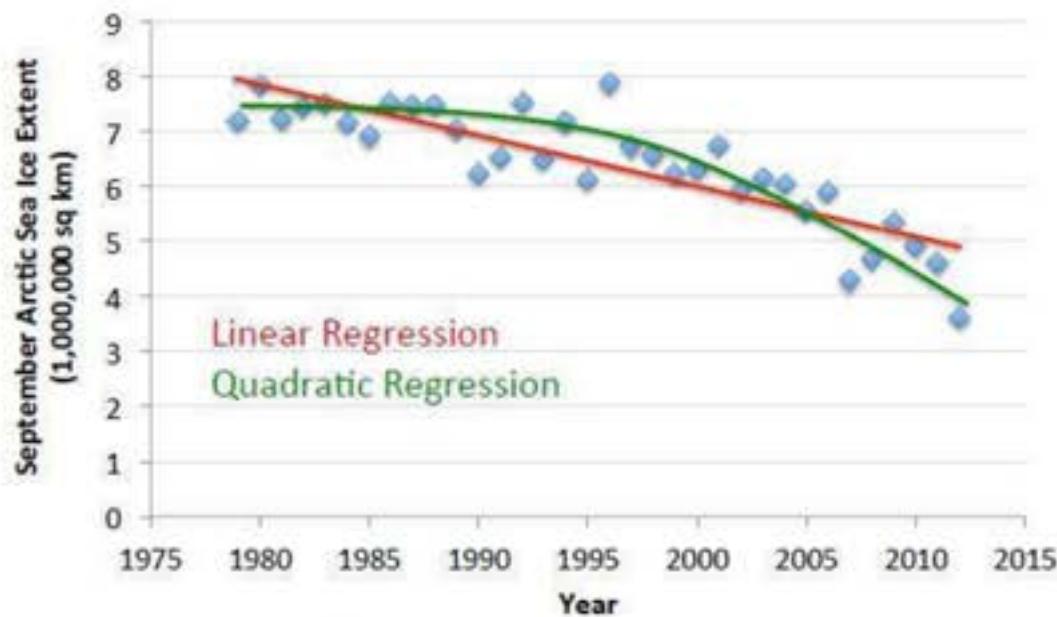
Regression

- model the relationship between one or more *independent* or **predictor** variables and a *dependent* or **response** variable

Regression

Given:

- Data $X = \{x^{(1)}, \dots, x^{(n)}\}$ where $x^{(i)} \in \mathbb{R}^d$
- Corresponding labels $y = \{y^{(1)}, \dots, y^{(n)}\}$ where $y^{(i)} \in \mathbb{R}$

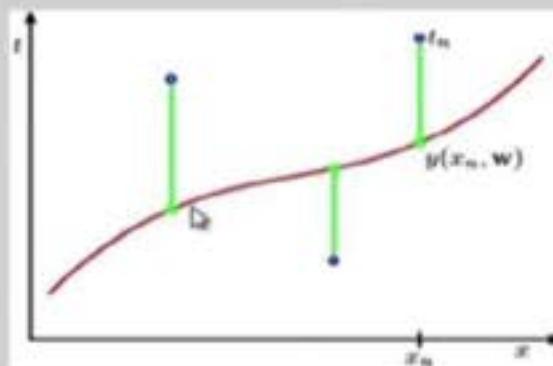
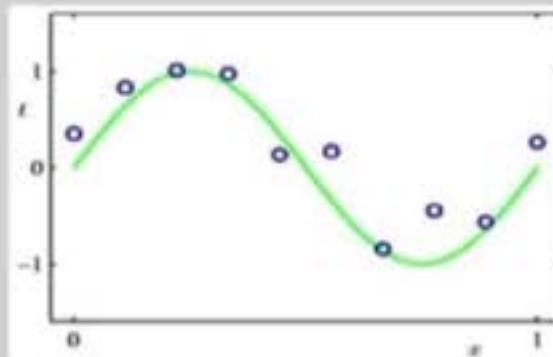


Data from G. Witt. Journal of Statistics Education, Volume 21, Number 1 (2013)

2

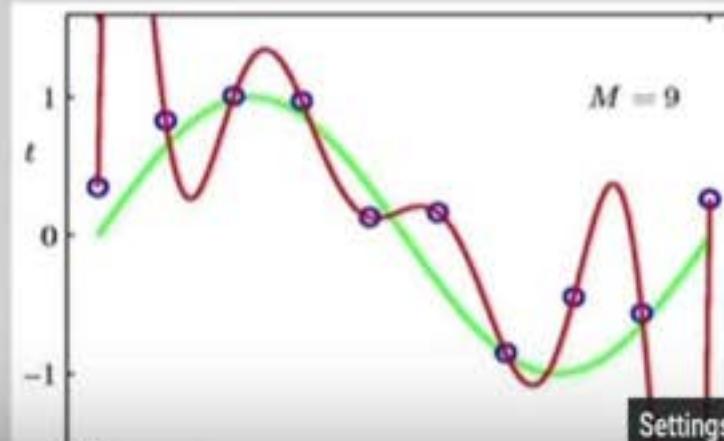
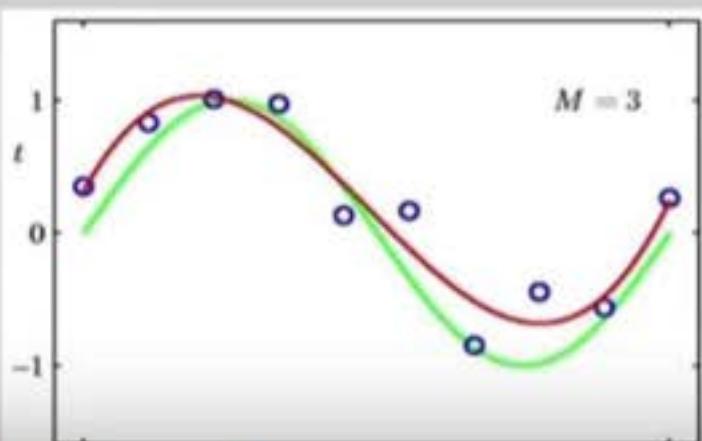
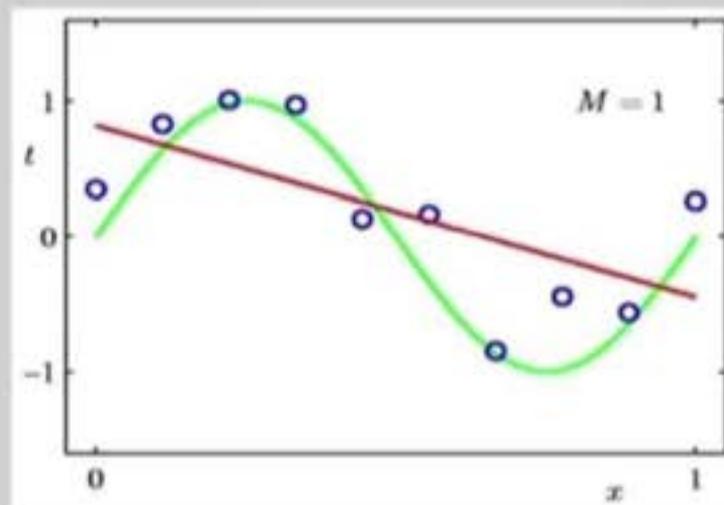
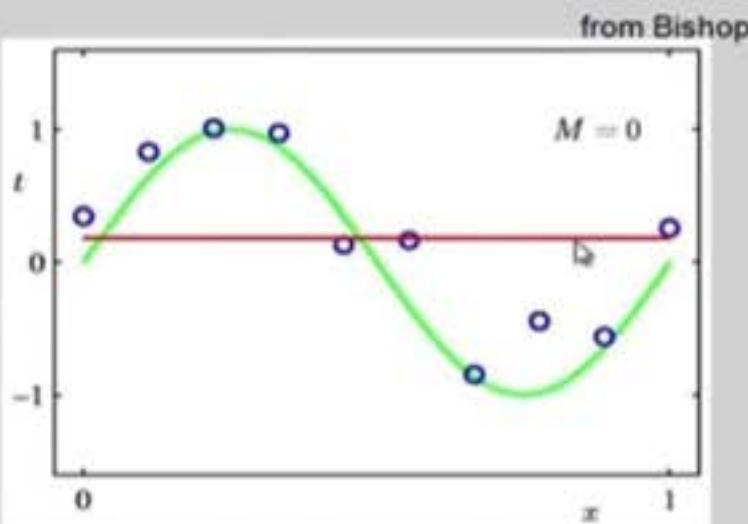
A Simple Example: Fitting a Polynomial

- The green curve is the true function (which is not a polynomial)
- We may use a loss function that measures the squared error in the prediction of $y(x)$ from x .

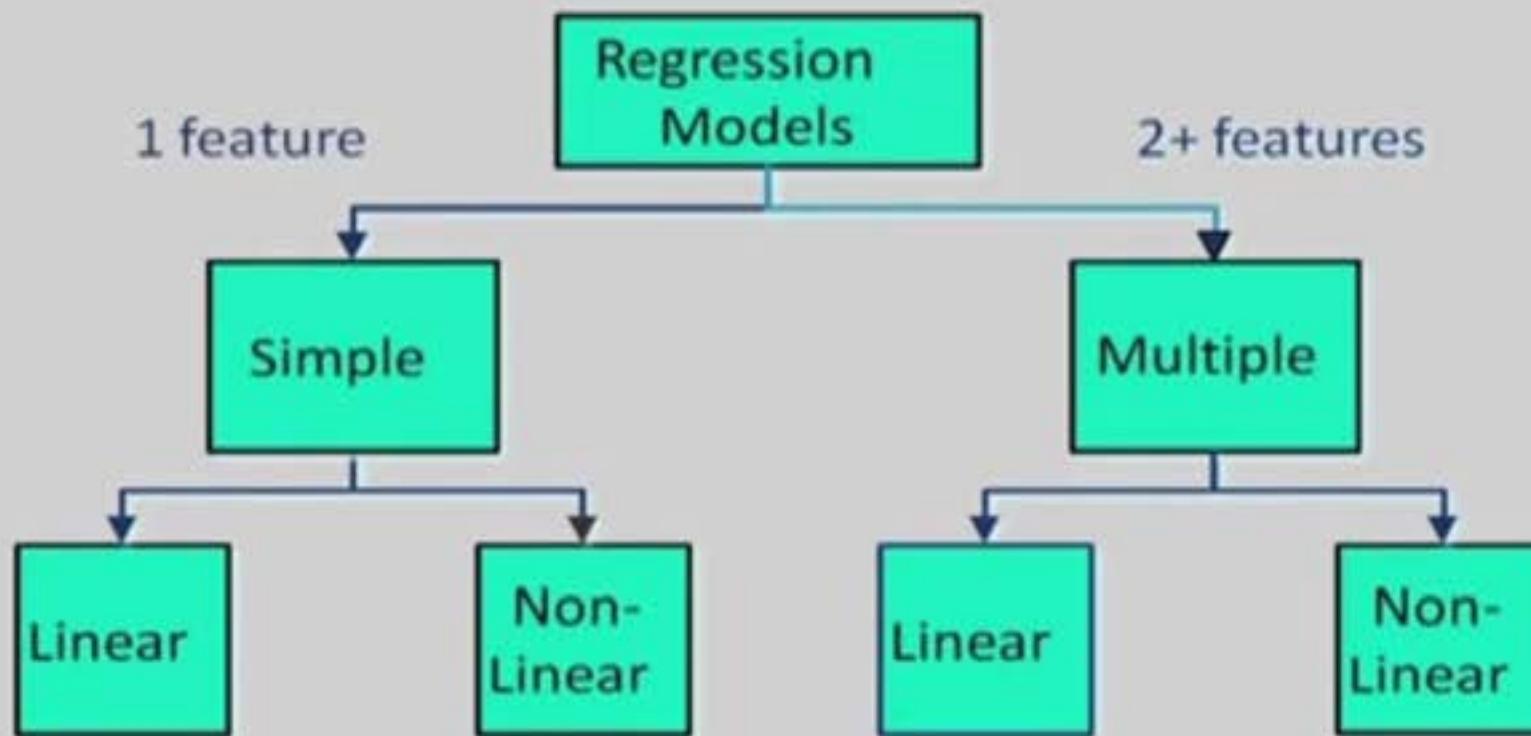


from Bishop's book on Machine Learning

Some fits to the data: which is best?

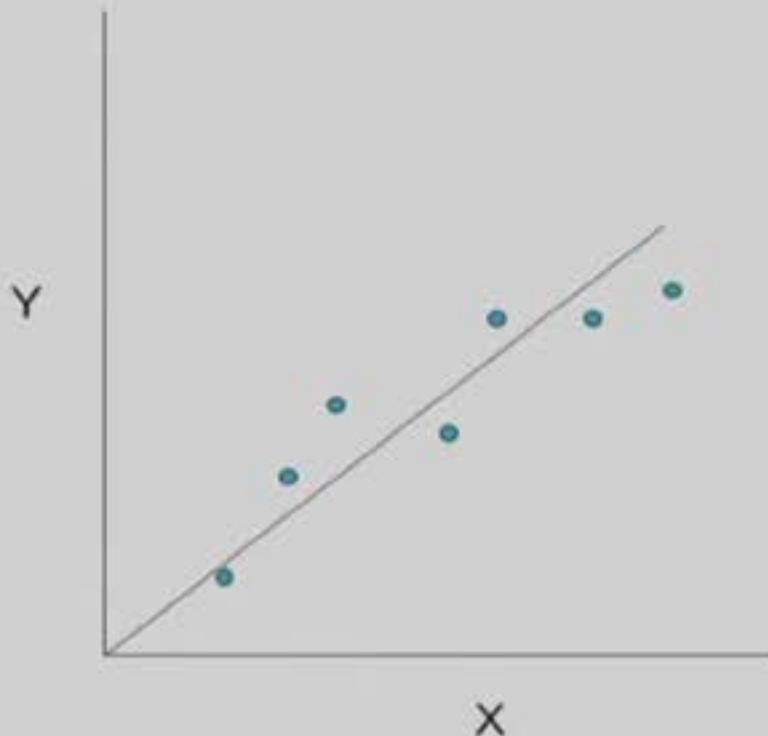


Types of Regression Models



Linear regression

- Given an input x compute an output y
- For example:
 - Predict height from age
 - Predict house price from house area
 - Predict distance from wall from sensors



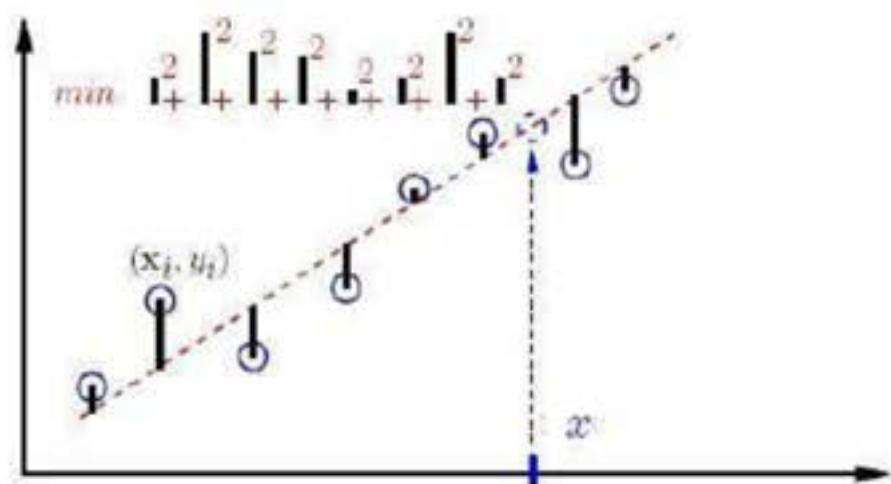
Linear regression Model

$$y = \theta_0 + \theta_1 x_1$$

θ_0 Y intercept

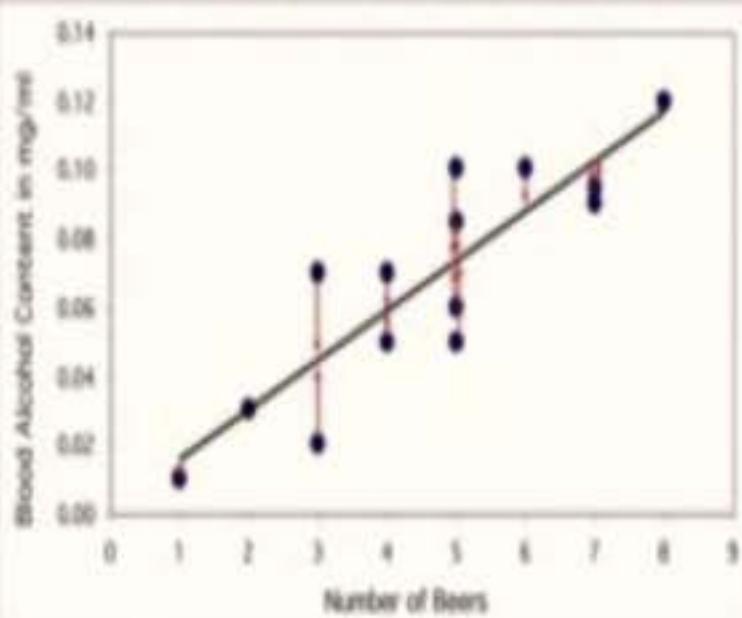
θ_1 Slope

- Fit model by minimizing sum of squared errors



The regression line

The least-squares regression line is the unique line such that the sum of the squared vertical (y) distances between the data points and the line is the smallest possible.



- Simple Linear regression: involves a response variable y and a single predictor variable x

$$y = \theta_0 + \theta_1 x$$

where θ_0 (y -intercept) and θ_1 (slope) are regression coefficients

- Method of least squares: estimates the best-fitting straight line

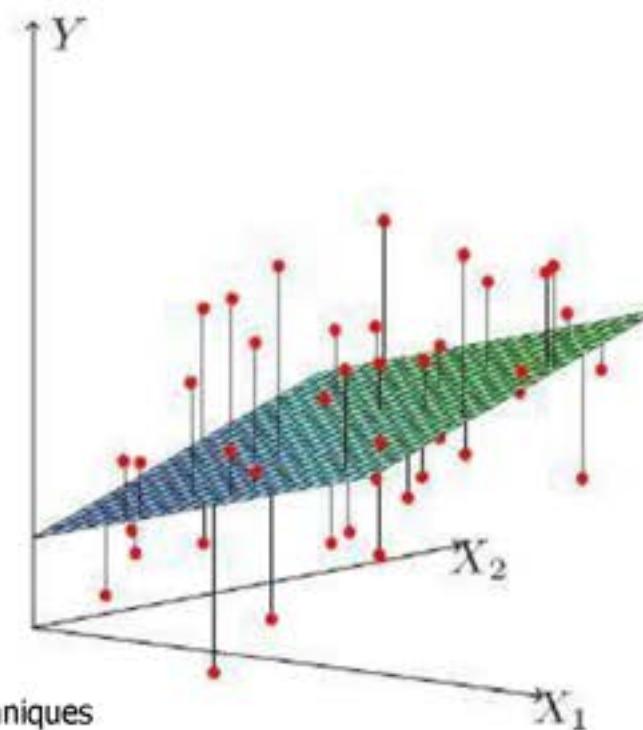
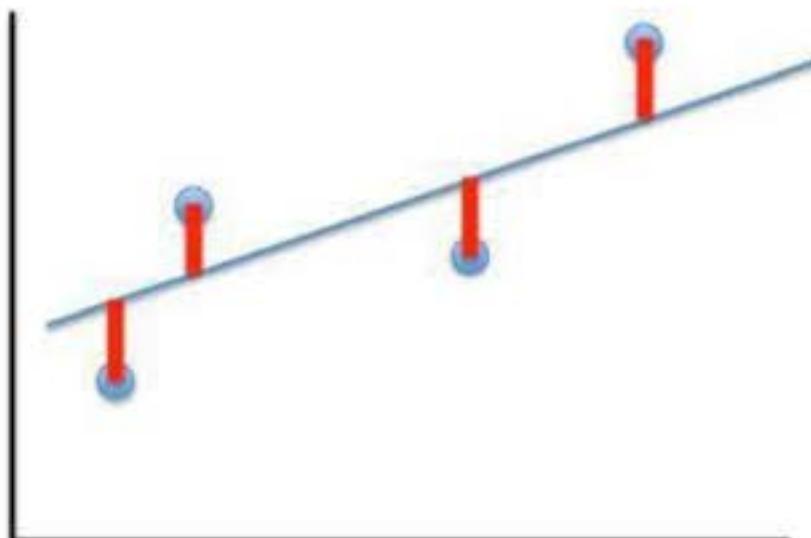
$$\theta_1 = \frac{\sum_{i=1}^{|D|} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{|D|} (x_i - \bar{x})^2}$$
$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

Least Squares Linear Regression

- Cost Function

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^n \left(h_{\theta}(x^{(i)}) - y^{(i)} \right)^2$$

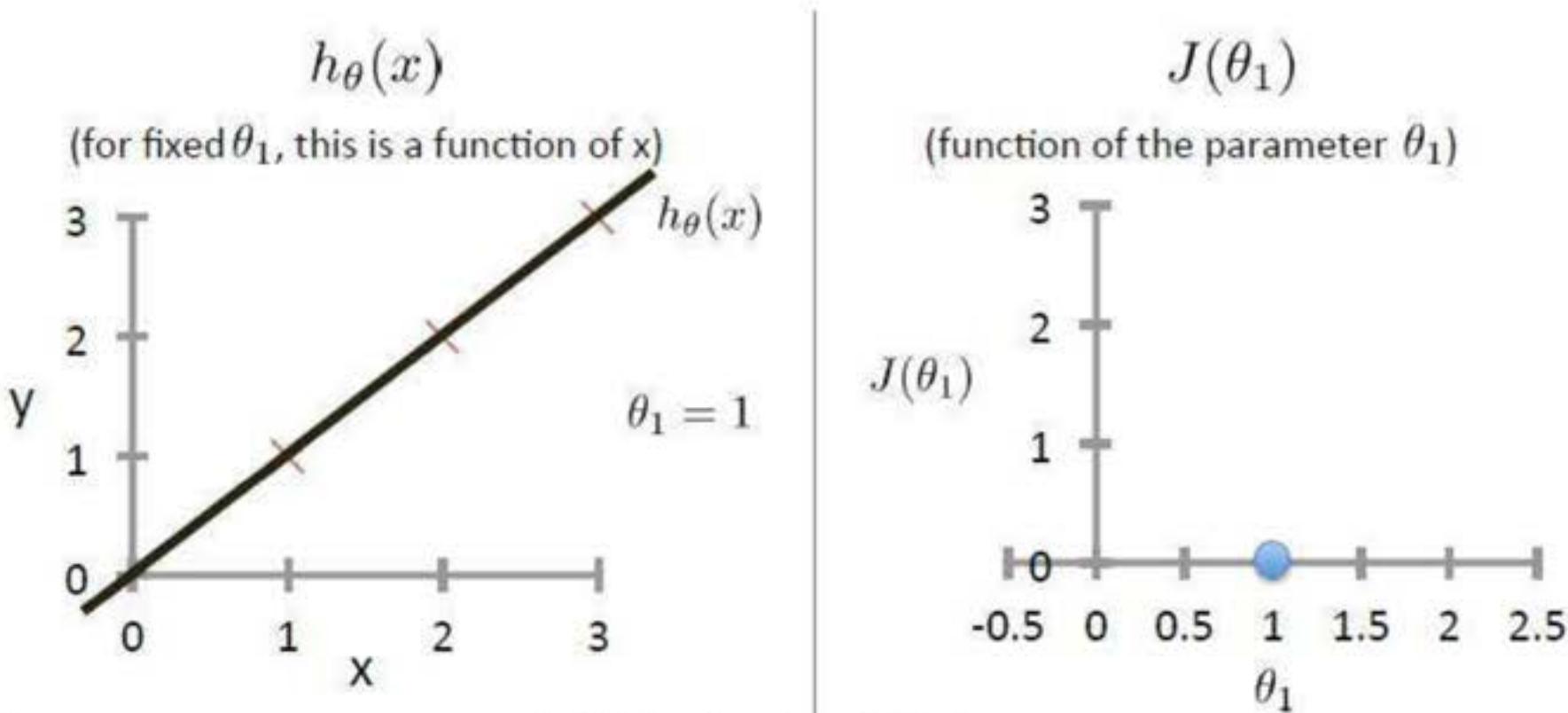
- Fit by solving $\min_{\theta} J(\theta)$



Intuition Behind Cost Function

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^n \left(h_{\theta}(x^{(i)}) - y^{(i)} \right)^2$$

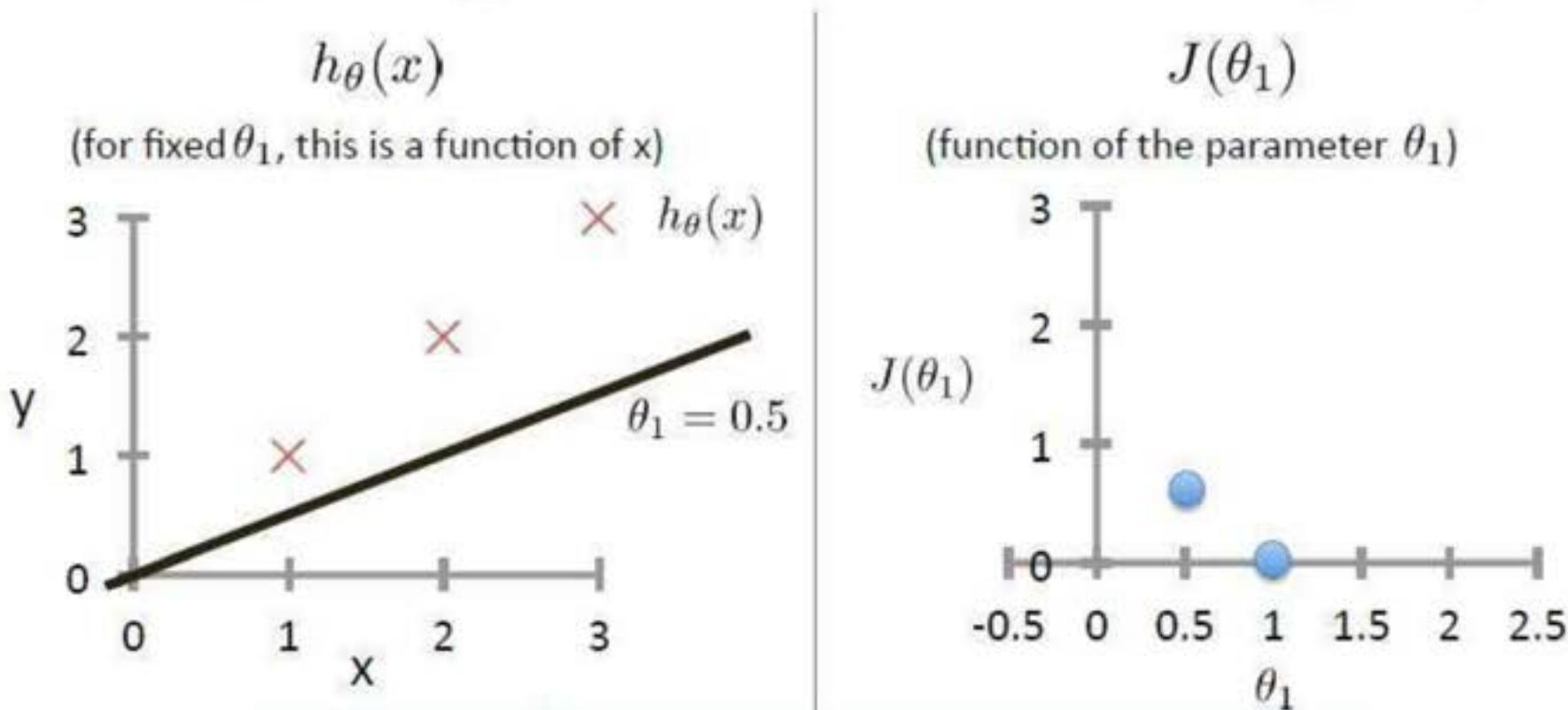
For insight on $J()$, let's assume $x \in \mathbb{R}$ so $\theta = [\theta_0, \theta_1]$



Intuition Behind Cost Function

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^n \left(h_{\theta}(x^{(i)}) - y^{(i)} \right)^2$$

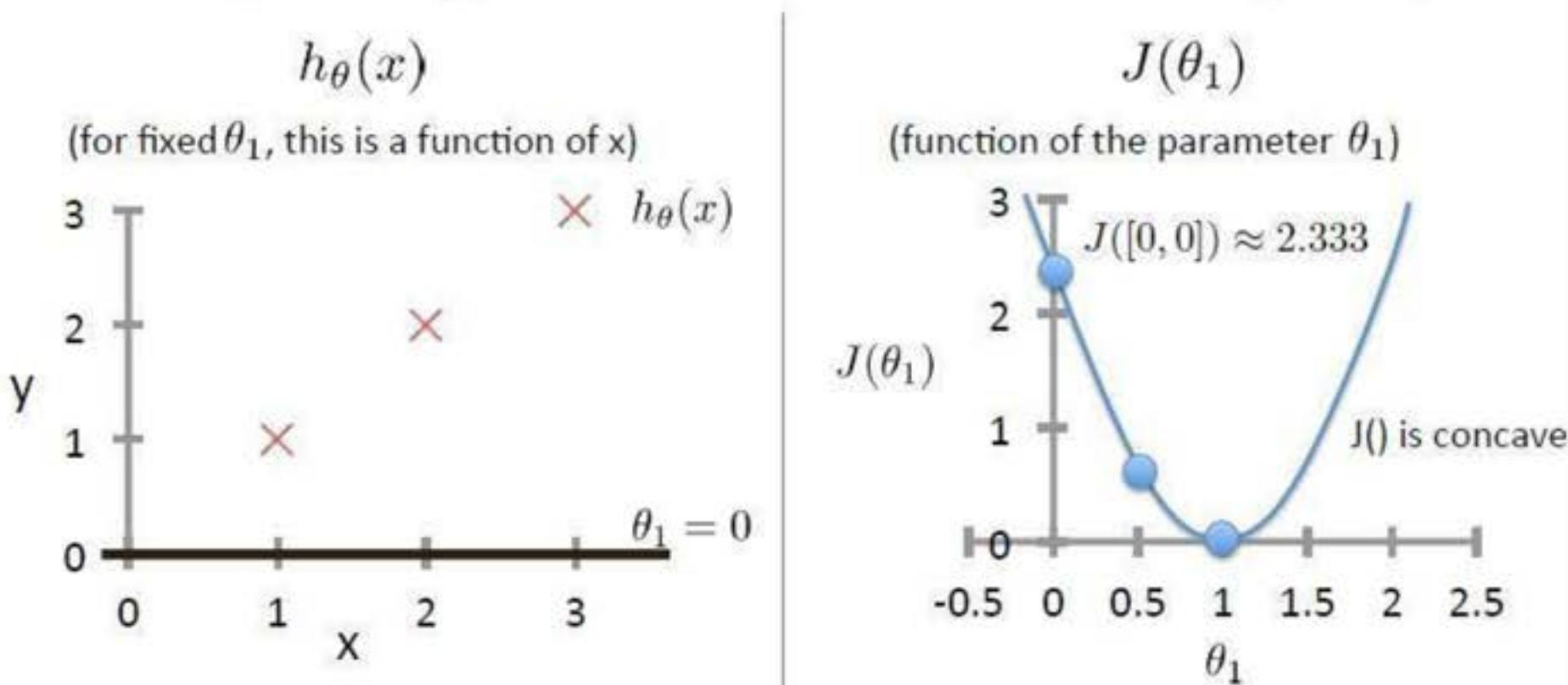
For insight on $J()$, let's assume $x \in \mathbb{R}$ so $\theta = [\theta_0, \theta_1]$



Intuition Behind Cost Function

$$J(\theta) = \frac{1}{2n} \sum_{i=1}^n \left(h_{\theta}(x^{(i)}) - y^{(i)} \right)^2$$

For insight on $J()$, let's assume $x \in \mathbb{R}$ so $\theta = [\theta_0, \theta_1]$



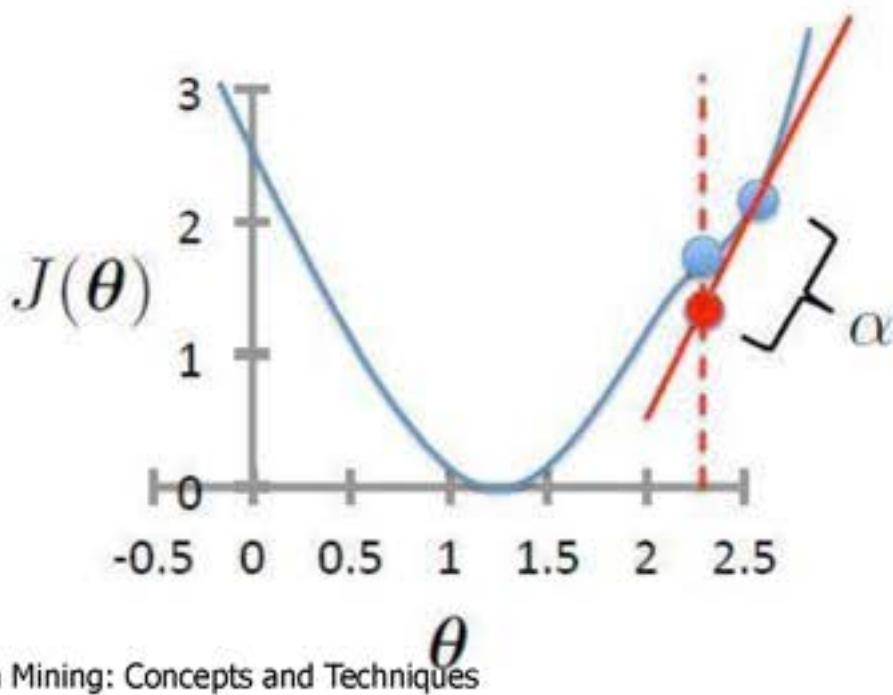
Gradient Descent

- Initialize θ
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

simultaneous update
for $j = 0 \dots d$

learning rate (small)
e.g., $\alpha = 0.05$



Gradient Descent for Linear Regression

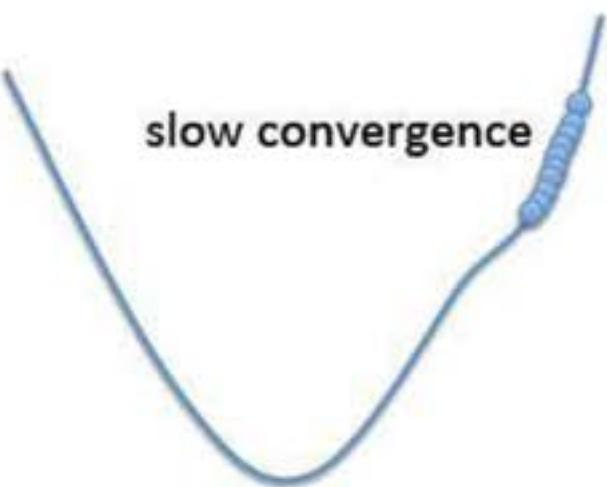
- Initialize θ
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{1}{n} \sum_{i=1}^n \left(h_{\theta}(\mathbf{x}^{(i)}) - y^{(i)} \right) x_j^{(i)}$$

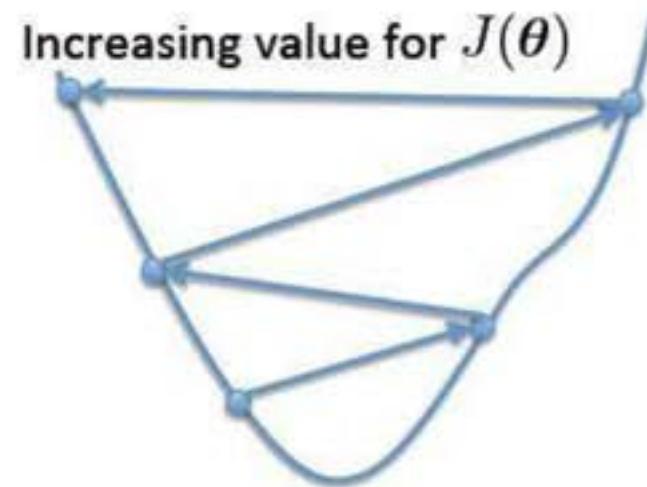
simultaneous
update
for $j = 0 \dots d$

Choosing α

α too small



α too large



- May overshoot the minimum
- May fail to converge
- May even diverge

To see if gradient descent is working, print out $J(\theta)$ each iteration

- The value should decrease at each iteration
- If it doesn't, adjust α

x	y
2	4
3	5
5	7
7	10
9	15

x	y	x- \bar{x}	y- \bar{y}	$(x-\bar{x})^2$	$(x-\bar{x})*(y-\bar{y})$
2	4	-3.2	-4.2	10.24	13.44
3	5	-2.2	-3.2	4.84	7.04
5	7	-0.2	-1.2	0.04	0.24
7	10	1.8	1.8	3.24	3.24
9	15	3.8	6.8	14.44	25.84
5.2	8.2			$\Sigma 32.8$	$\Sigma 49.8$

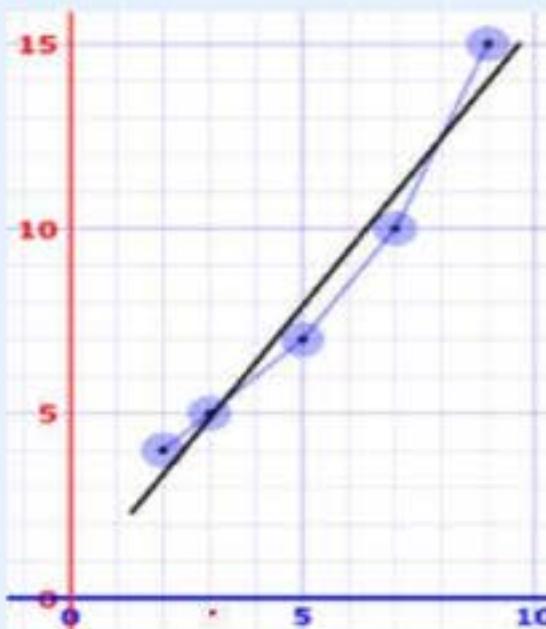
$$\theta_0 = 0.305$$

$$\theta_1 = 1.518$$

$$Y = 1.518x + 0.305$$

x	y	$y = 1.518x + 0.305$	error
2	4	3.34	-0.66
3	5	4.86	-0.14
5	7	7.89	0.89
7	10	10.93	0.93
9	15	13.97	-1.03

Here are the (x,y) points and the line $y = 1.518x + 0.305$ on a graph:



Nice fit!

Sam hears the weather forecast which says "we expect 8 hours of sun tomorrow", so he uses the above equation to estimate that he will sell

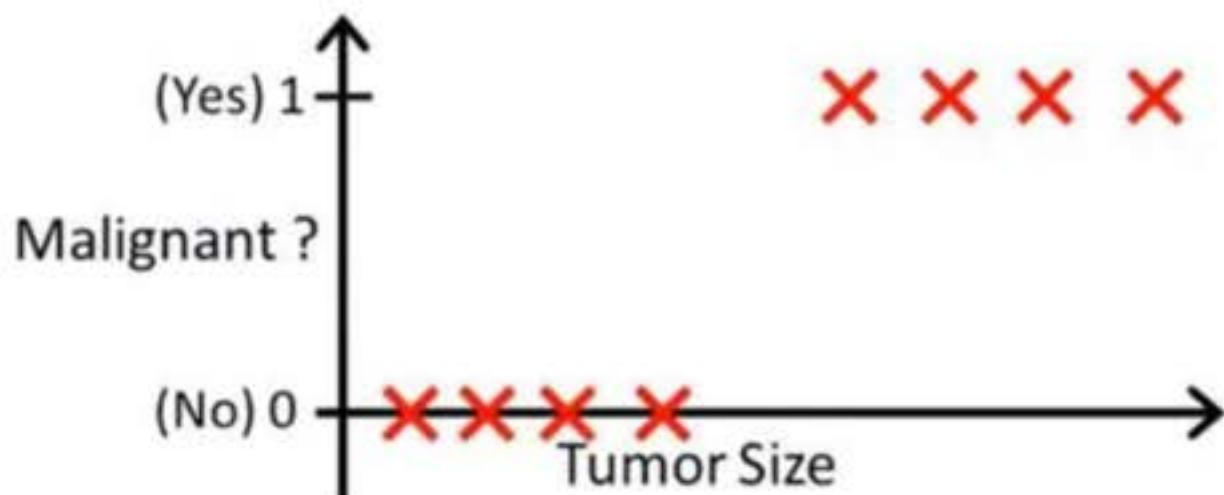
$$y = 1.518 \times 8 + 0.305 = 12.45 \text{ Ice Creams}$$

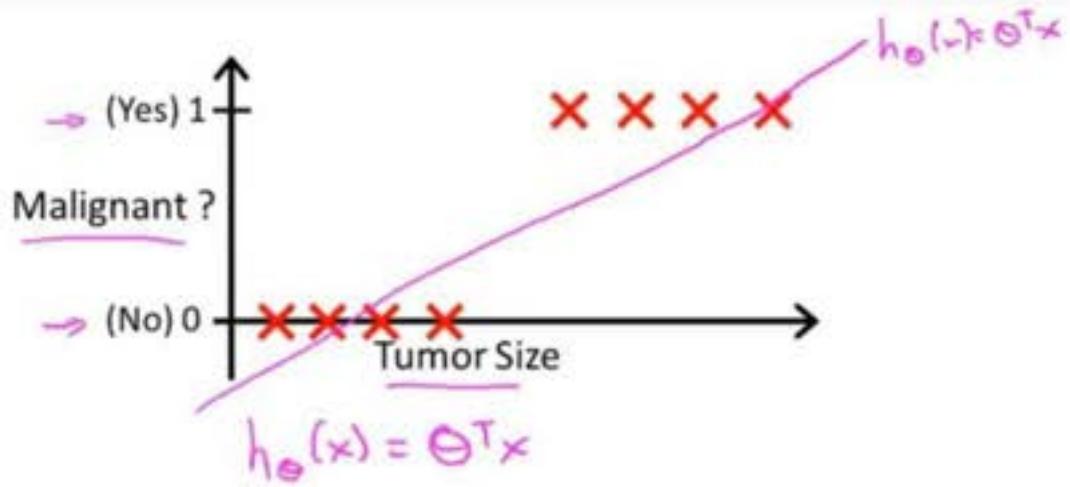
Logistic regression

- Machine learning classification algorithm used to predict the **probability of categorical dependent variable** to belong to a class.
- If one independent variable- **simple logistic regression**
- If more than one independent variable- **multiple logistic regression**
- Logistic regression predicts the probability of occurrence of an event by fitting data to a logistic sigmoid /logit function.
- logistic regression transforms its output using the logistic sigmoid /logit function to return a probability value which can then be mapped to two or more discrete classes.

Linear Regression	Logistic Regression
Dependent(y) and independent variables are continuous(quantitative)	Dependent variable(y) is categorical(qualitative). Independent variable can be continuous or categorical
Used for regression(predicting values of y)	Used for classification
Eg: predict the student's test score on a scale of 0 - 100.	Eg: predict whether the student passed or failed
Linear regression predictions are continuous (numbers in a range)	Logistic regression predictions are discrete (only specific values or categories are allowed).

Logistic Regression





→ Threshold classifier output $h_\theta(x)$ at 0.5:

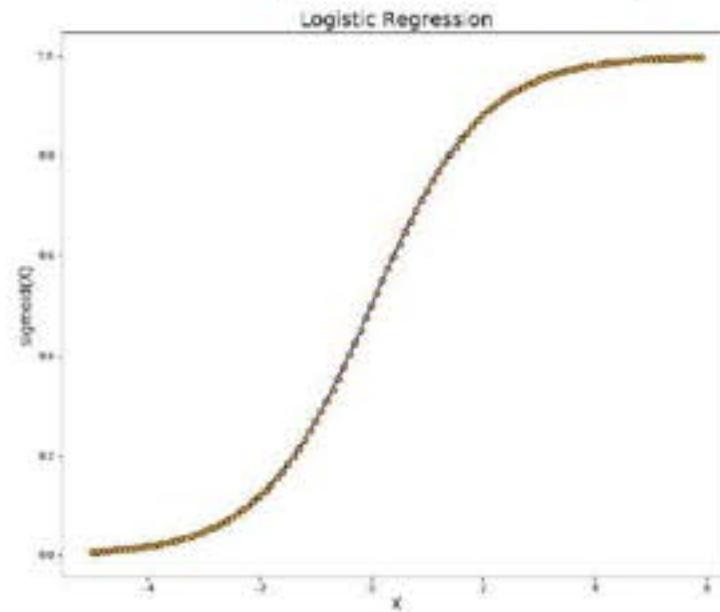
If $h_\theta(x) \geq 0.5$, predict "y = 1"

If $h_\theta(x) < 0.5$, predict "y = 0"

Logistic Regression(Handling Categorical Target Feature)

Logistic function

$$\text{logistic}(x) = \frac{1}{1 + e^{-x}}$$



To build a logistic regression model, we threshold the output of the basic linear regression model using the logistic function.

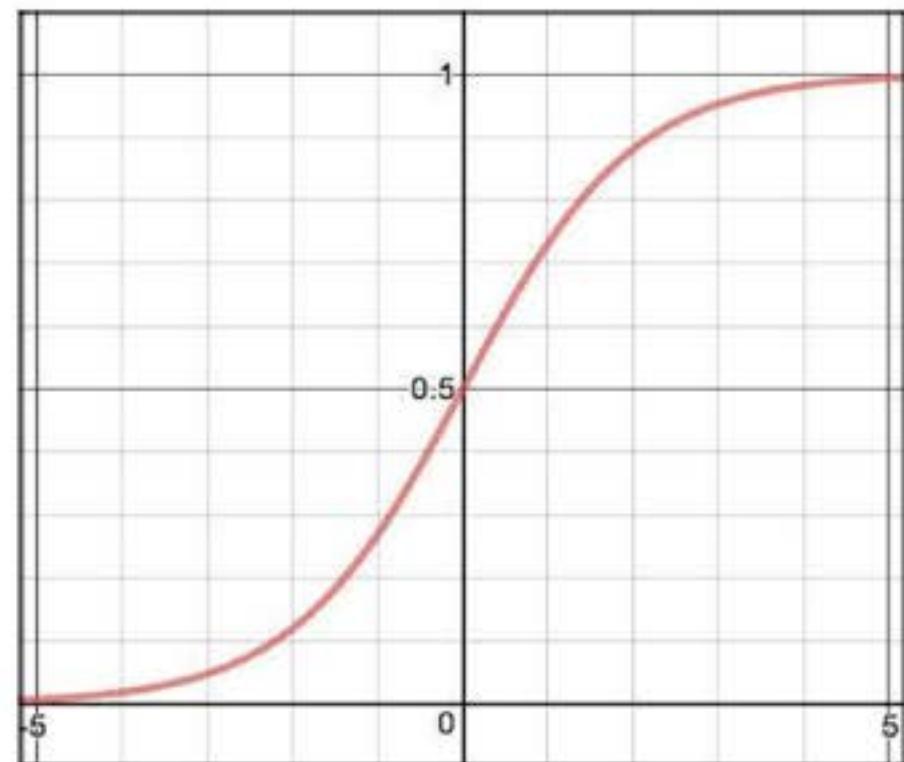
$$h_{\theta}(x) = \frac{1}{1 + e^{-\theta^T x}}$$

Decision Boundary

- Our current prediction function returns a probability score between 0 and 1. In order to map this to a discrete class (true/false, cat/dog), we select a threshold value or tipping point above which we will classify values into class 1 and below which we classify values into class 2.

$$p \geq 0.5, \text{class} = 1$$

$$p < 0.5, \text{class} = 0$$



If 'Z' goes to infinity, Y(predicted) will become 1 and if 'Z' goes to negative infinity, Y(predicted) will become 0.

Thank you !!!!



Decision Tree

Classification & Regression

Peeta Basa Pati, Ph. D.

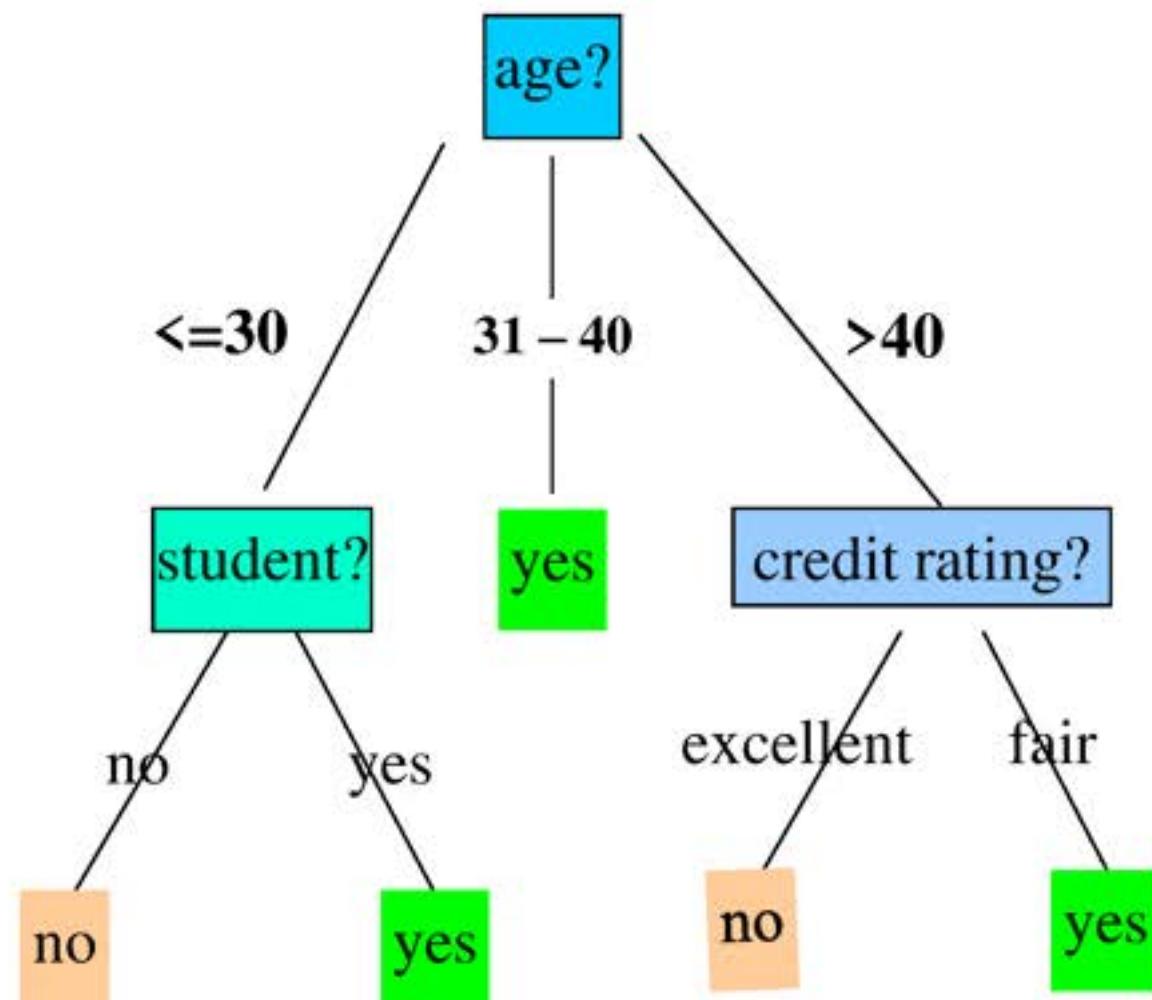
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Agenda

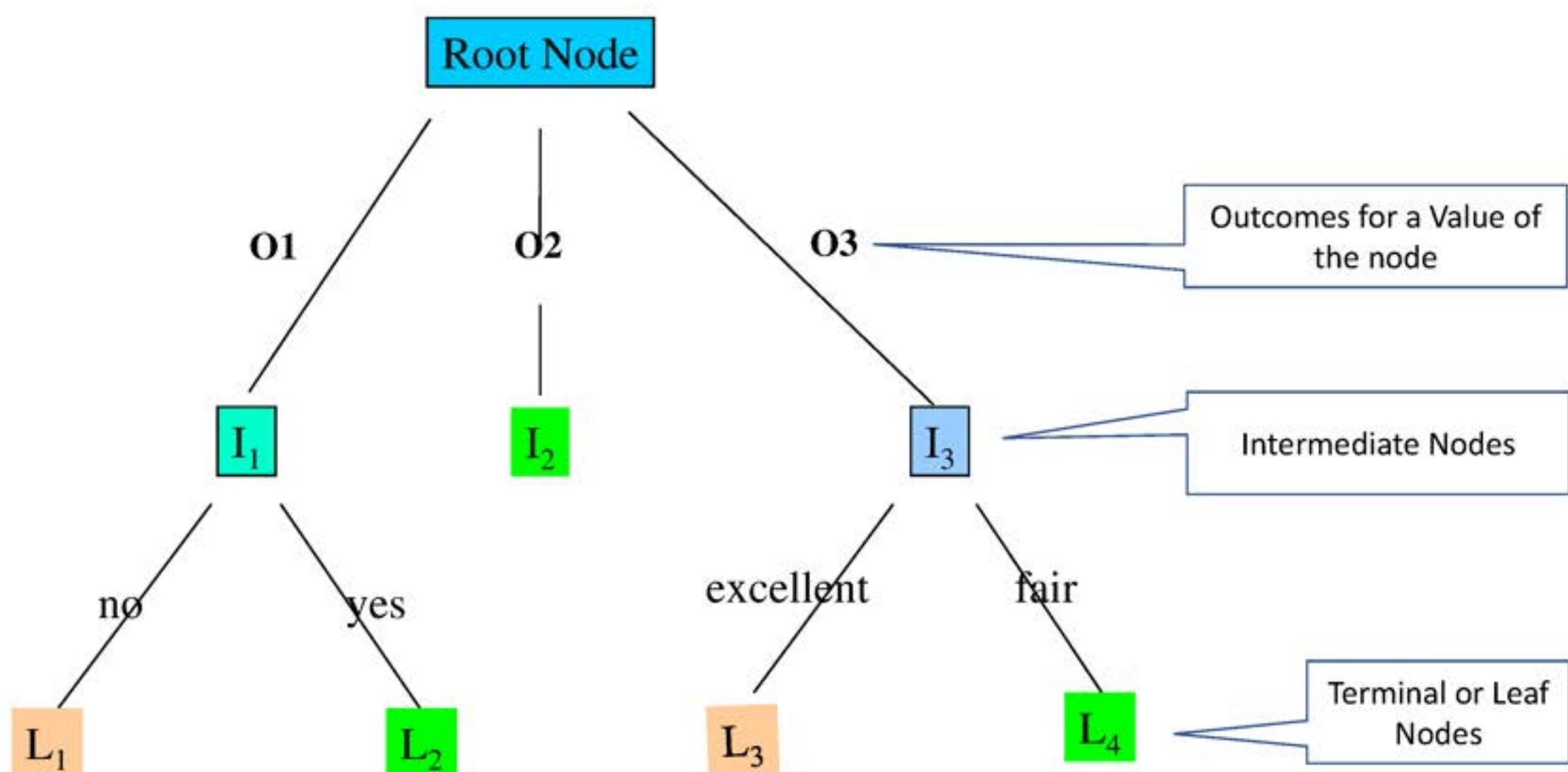
- Decision Tree
 - Anatomy / structure – nodes
 - Impurity measures – Entropy, Information Gain, Gini
 - Pruning – Pre/Post-pruning
- Random Forest
- Regression Trees
- Demos

Decision Tree

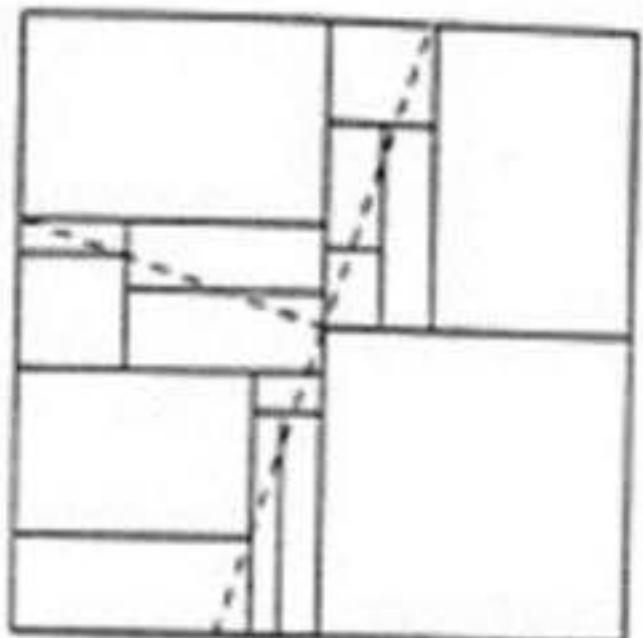
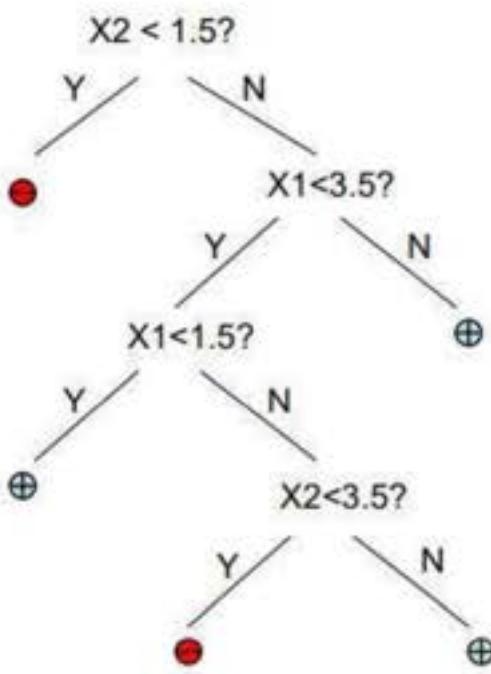
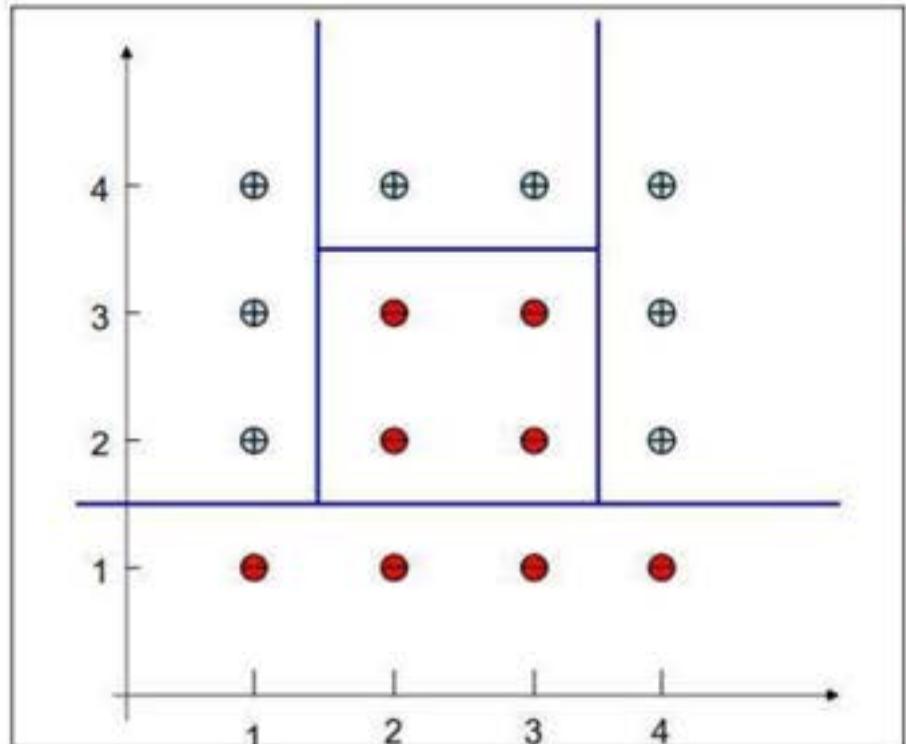
age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes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



Anatomy of a Tree



Geometrical View



Decision Tree Definition

- A decision tree is a tree in which each branch node represents a choice between a number of alternatives, and each leaf node represents a decision.
- A type of supervised learning algorithm.

DEFINITION

Dataset $D = \{t_1, t_2, \dots, t_n\}$, where t_i denotes a tuple

Each tuple t_i consists of a set of attributes $F = \{f_1, f_2, \dots, f_m\}$

Each tuple t_i belongs to one from the set of classes $C = \{c_1, c_2, \dots, c_k\}$

A decision tree T is a tree associated with D with following properties:

- Root & each internal node is labeled with an attribute f_i
- Edges are predicates that are applied to the attribute node
- Each leaf node is labeled with class c_j

Some Questions

- What should be the root node?
- What should be the decision node?
- When should I stop splitting?

How to Choose an attribute at a node?

Attribute Selection

Buys = Yes {9}
Buys = No {5}

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

YES

No

Fair

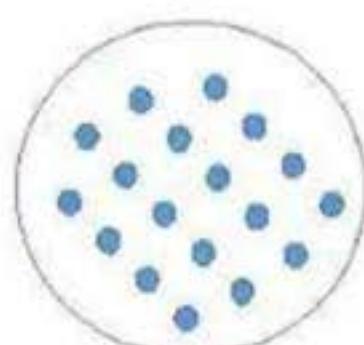
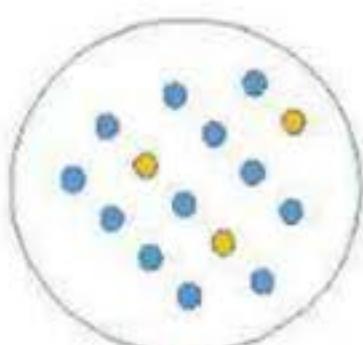
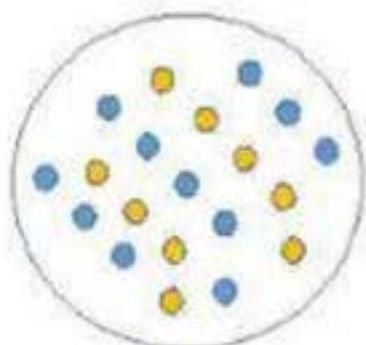
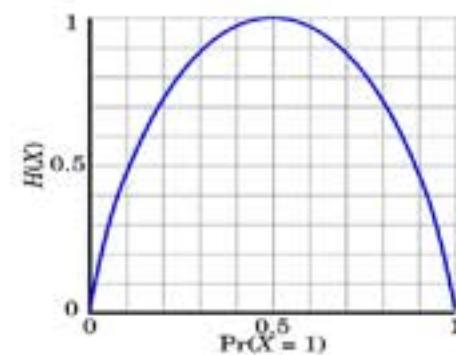
Excellent

Buys = Yes {9}
Buys = No {5}

Entropy

$$H[X] = - \sum_{i=1}^n P(x_i) \log P(x_i)$$

Higher the entropy higher impurity of data



- **Entropy is 0 if all the members of S belong to the same class.**
- **Entropy is 1 when the collection contains an equal no. of +ve and -ve examples.**
- **Entropy is between 0 and 1 if the collection contains unequal no. of +ve and -ve examples.**

Entropy: Exercise



Count of Events → 10

Heads = 10 → Prob (Heads) = 1

Tails = 0 → Prob (Tails) = 0

$$\begin{aligned}\text{Entropy} &= -(P(H) * \log(P(H)) + P(T) * \log(P(T))) \\ &= -(1.0 * \log(1.0) + 0.0 * \log(0.0)) = 0\end{aligned}$$

log base 2 is used here!!



$P(H) = 0.8; P(T) = 0.2$

$$\text{Entropy} = -(P(H) * \log(P(H)) + P(T) * \log(P(T))) = -(0.8 * \log(0.8) + 0.2 * \log(0.2)) = -(-0.25 - 0.46) = 0.71$$



$P(H) = 0.5; P(T) = 0.5$

$$\text{Entropy} = -(P(H) * \log(P(H)) + P(T) * \log(P(T))) = -(0.5 * \log(0.5) + 0.5 * \log(0.5)) = -(-0.5 - 0.5) = 1$$

Information Gain

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Higher the information gain lower is the uncertainty of decision.

Information Gain is always greater than or equal to Zero. Only in a scenario where the event probabilities don't change, it's zero.

It's biased towards multivalued attributes.

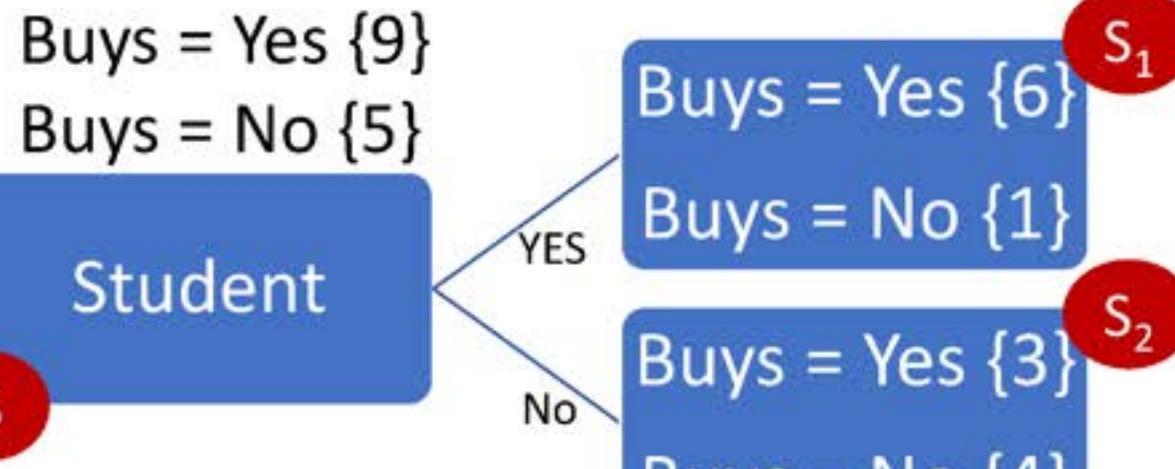
Information Gain: Example

$$\begin{aligned}H(S) &= -(P(\text{Buys} = \text{Yes}) * \log(P(\text{Buys} = \text{Yes}))) \\&\quad + P(\text{Buys} = \text{No}) * \log(P(\text{Buys} = \text{No}))) \\&= -\left(\frac{9}{14} * \log\left(\frac{9}{14}\right) + \frac{5}{14} * \log\left(\frac{5}{14}\right)\right) \\&= -(0.64 * -0.64 + 0.36 * -1.47) = 0.94\end{aligned}$$

$$\begin{aligned}H(S_1) &= -(P(\text{Buys} = \text{Yes}) * \log(P(\text{Buys} = \text{Yes}))) \\&\quad + P(\text{Buys} = \text{No}) * \log(P(\text{Buys} = \text{No}))) \\&= -\left(\frac{6}{7} * \log\left(\frac{6}{7}\right) + \frac{1}{7} * \log\left(\frac{1}{7}\right)\right) \\&= -(0.86 * -0.22 + 0.14 * -2.84) = 0.59\end{aligned}$$

$$\begin{aligned}H(S_2) &= -(P(\text{Buys} = \text{Yes}) * \log(P(\text{Buys} = \text{Yes}))) \\&\quad + P(\text{Buys} = \text{No}) * \log(P(\text{Buys} = \text{No}))) \\&= -\left(\frac{3}{7} * \log\left(\frac{3}{7}\right) + \frac{4}{7} * \log\left(\frac{4}{7}\right)\right) \\&= -(0.43 * -1.22 + 0.57 * -0.81) = 0.98\end{aligned}$$

$$\begin{aligned}\text{Information Gain}(S, A) &= H(S) - \left(\frac{|S_1|}{|S|} * H(S_1) + \frac{|S_2|}{|S|} * H(S_2)\right) \\&= 0.94 - (0.5 * 0.59 + 0.5 * 0.98) \\&= 0.94 - (0.29 + 0.49) = 0.94 - 0.78 = 0.16\end{aligned}$$

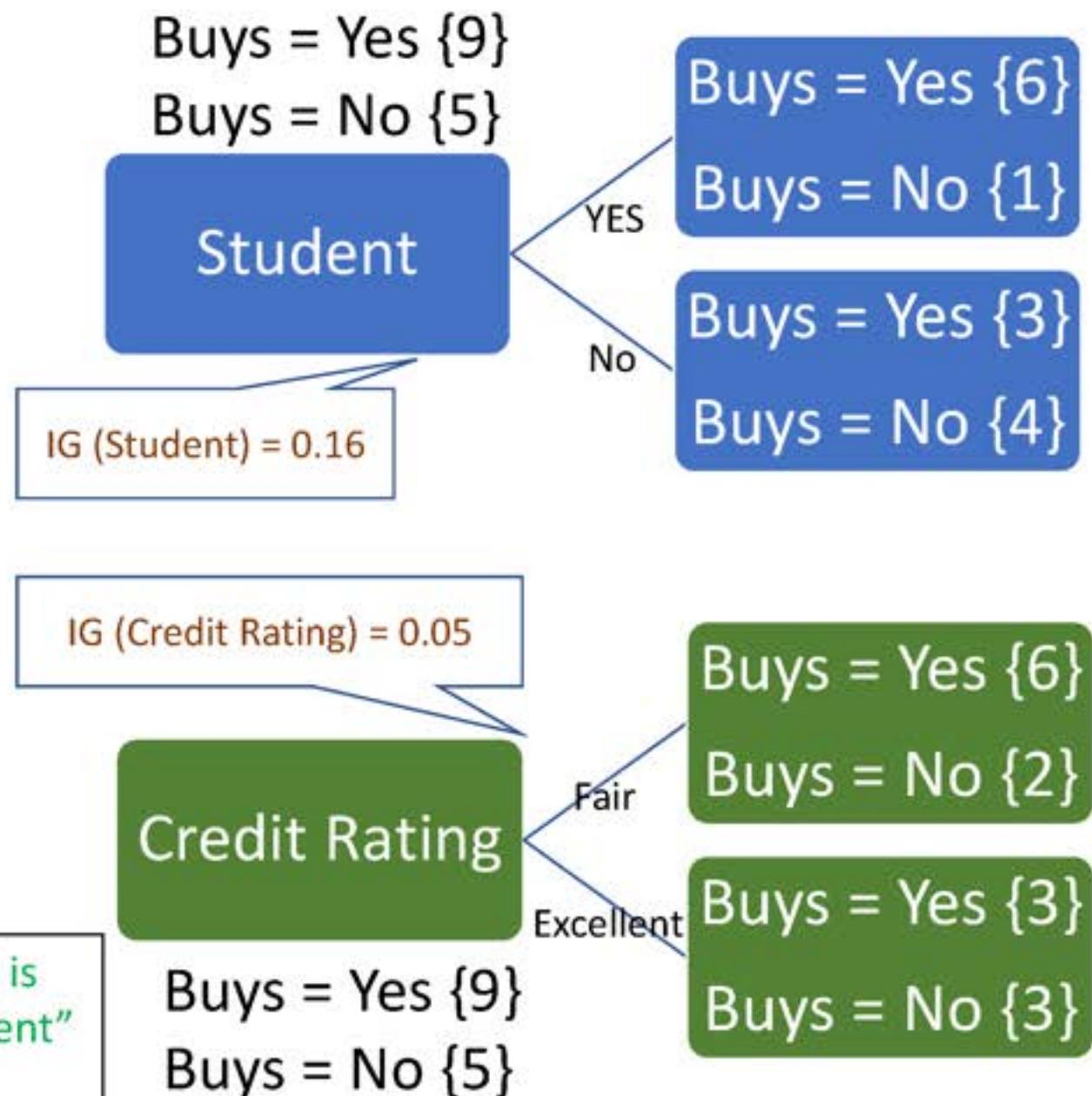


age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes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

Information Gain: Example

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

The Information Gain with "Student" attribute (0.16) is more than "Credit Rating" attribute (0.05). So, "Student" attribute is selected for splitting at this node.



Building Tree Progressively

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

$$Gain(age) = 0.25$$

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\ rating) = 0.048$$

Age has maximum information gain. So, age is selected as the best node to split as root node.

- The rules are:

IF $age = "<=30"$ AND $student = "no"$ THEN
 $buys_computer = "no"$

IF $age = "<=30"$ AND $student = "yes"$ THEN
 $buys_computer = "yes"$

IF $age = "31...40"$ THEN
 $buys_computer = "yes"$

IF $age = ">40"$ AND $credit_rating = "excellent"$ THEN
 $buys_computer = "no"$

IF $age = ">40"$ AND $credit_rating = "fair"$ THEN
 $buys_computer = "yes"$

Other Impurity Measures

Gini index

$$gini(D) = 1 - \sum_{j=1}^n p_j^2$$

$p_j \rightarrow$ relative frequency of class j in D

- Choose attribute with low gini index
- Difficulty when # of classes is large
- Favors tests that result in equal-sized partitions and purity in both partitions

Gain ratio

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$$

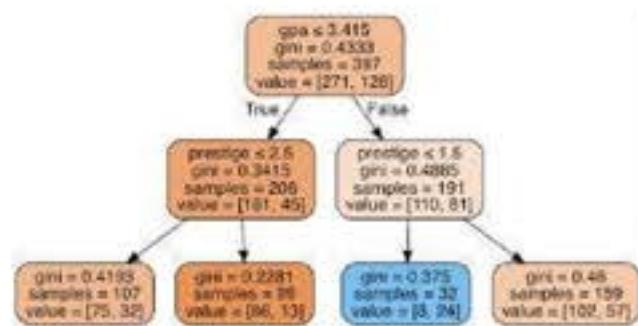
$$SplitInformation(S, A) \equiv - \sum_{i=1}^c \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

- Tends to prefer unbalanced splits in which one partition is much smaller than the others

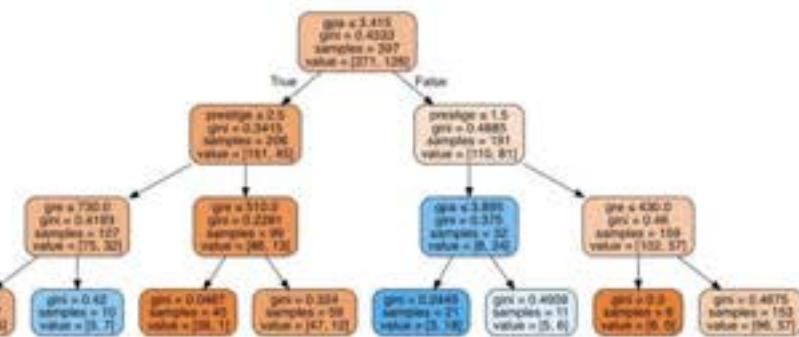
When to stop?

Depth of a Tree

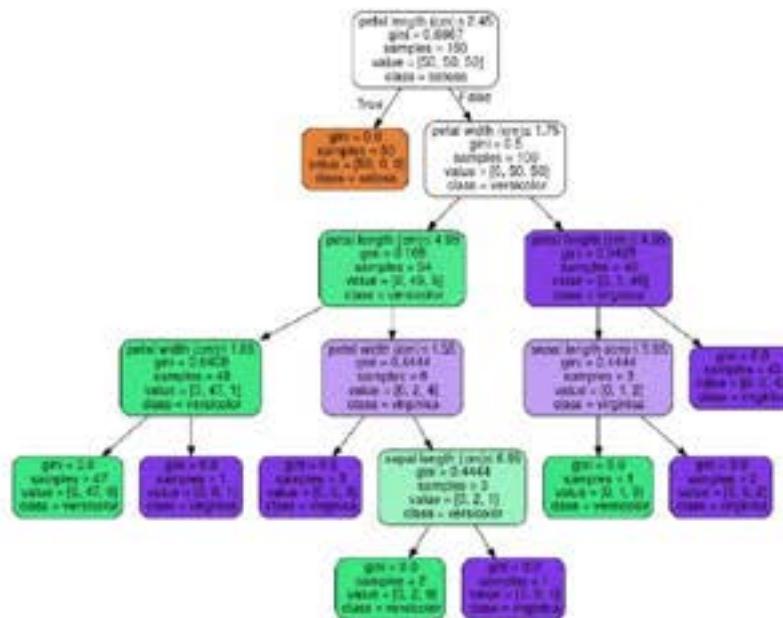
- Count of splits a tree makes before coming to a prediction
- Considered for the longest path to reach the leaf node



Depth = 2



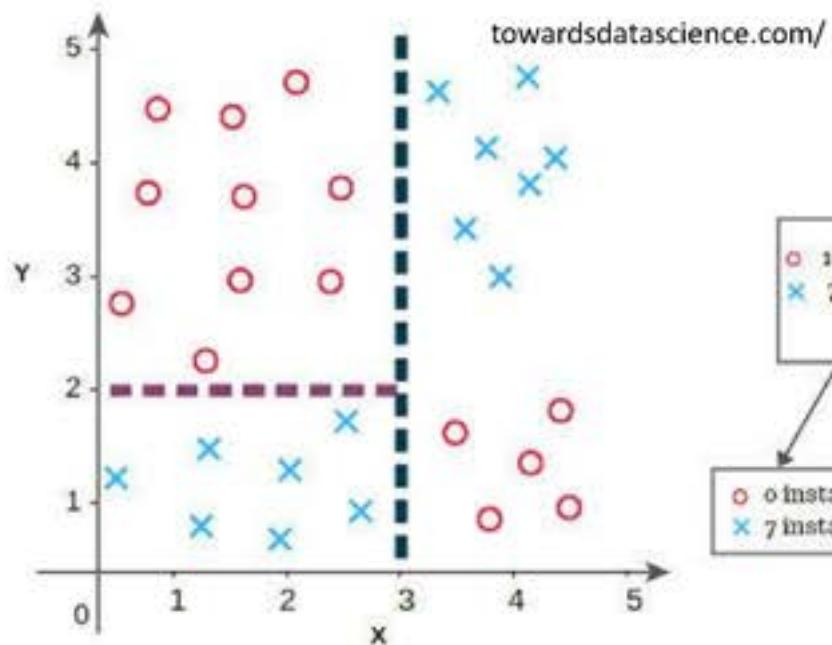
Depth = 3



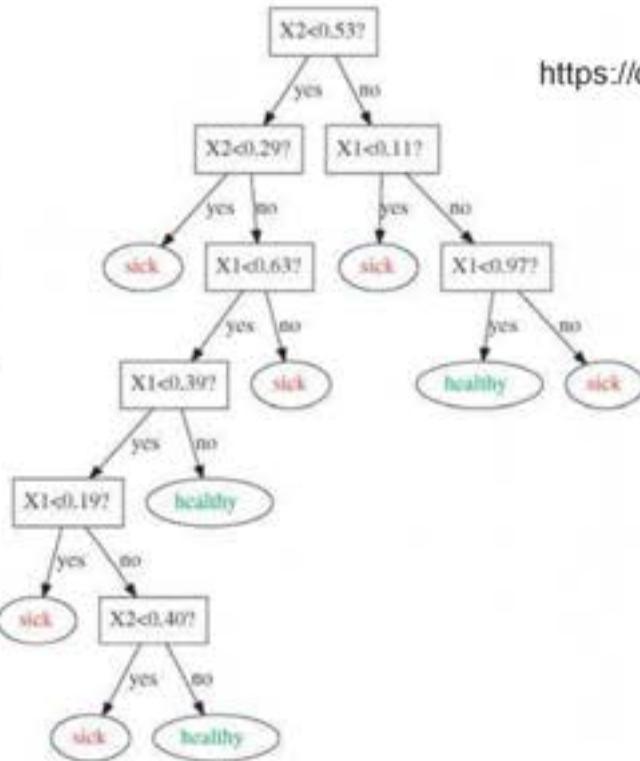
Depth = 4

Overfitting vs Underfitting

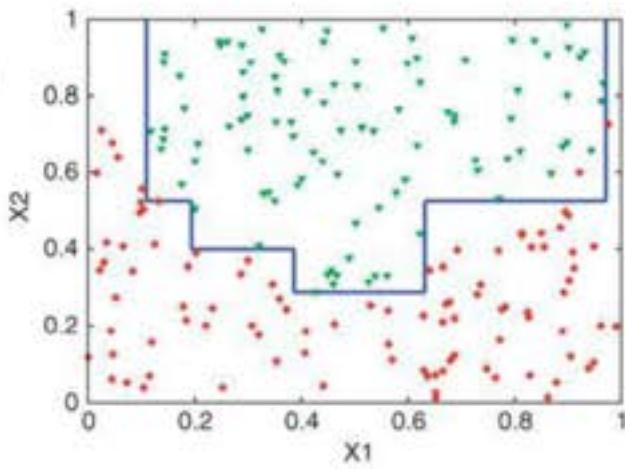
Very short tree – over generalization indicating an underfitting



Very long tree – overfitting

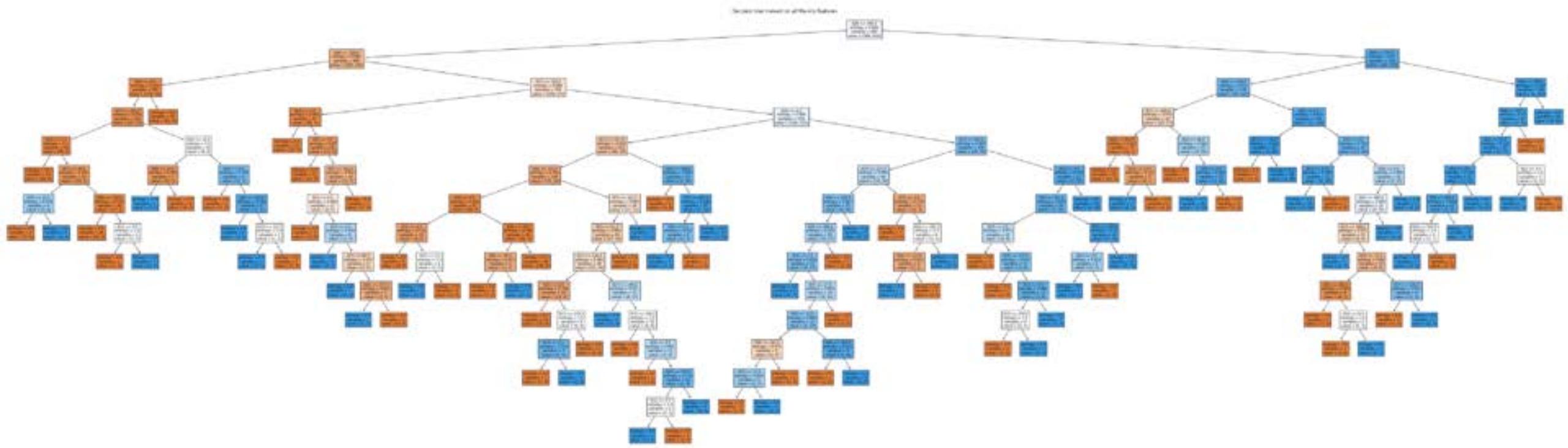


<https://doi.org/10.3929/ethz-a-010811999>



Inductive Bias: shorter trees are preferred over larger trees.

Very lengthy tree



Tree depth = 14

Image taken from demo data

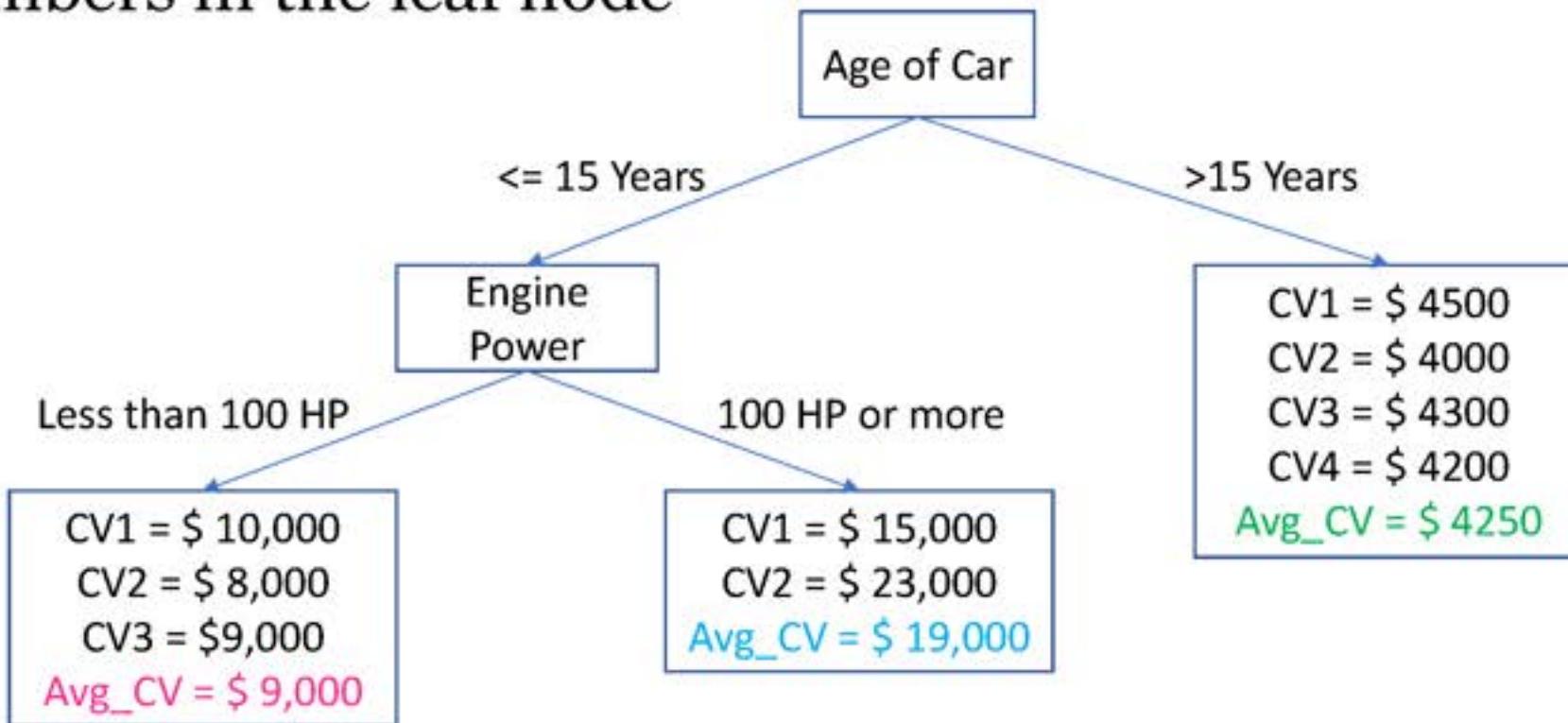
Pruning & effects

- Removal of a sub-tree which is either redundant or nor very useful
- Usually achieved by removal of the weakest rule at a split
- 2-types
 - Pre-pruning – achieved while building the tree
 - Also called as early stopping
 - Evaluate the pruning condition at each split & stop if met
 - Ex: $\text{Gain}(S,A) \leq \text{min-threshold}$ or $\text{depth} \geq \text{max-depth}$
 - Post-pruning – tree is fully grown and pruned in bottom-up approach
- Measures used
 - Reduced Error Pruning (REP) – uses a validation set to arrive at the min error condition
 - Cost Complexity Pruning – uses a combination of residual error & leaf node count
- Removes over-fitting, enhances generalization, reduces complexity

Regression Tree & Random Forest

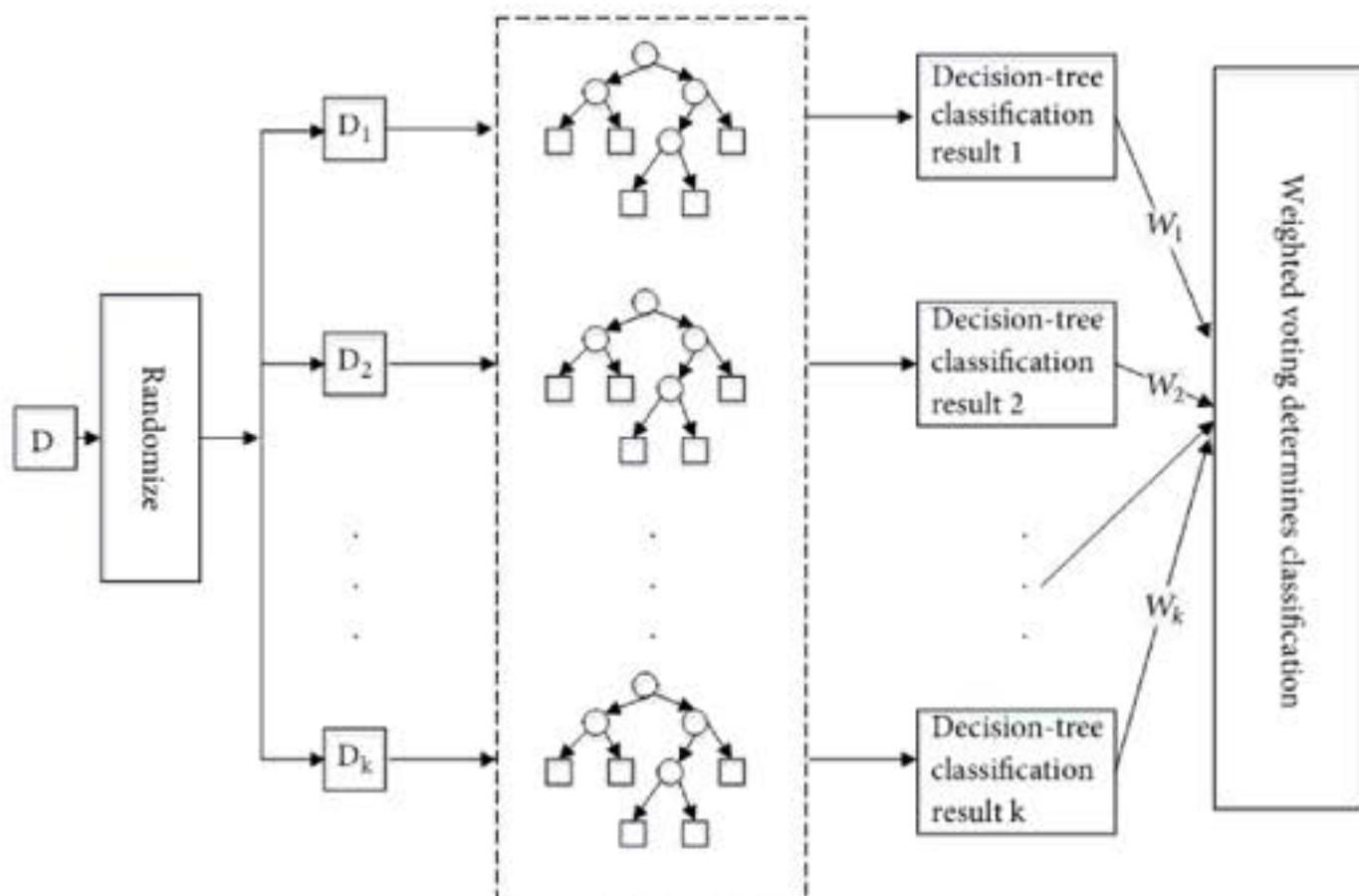
Regression Trees

- Conceptually same as DT but output is a value instead of a class label
- Impurity measure used – variance reduction
- The estimated value is arrived at by taking average of the values of members in the leaf node



Random Forest

- Ensemble of Trees to perform classification or regression task
- Bagging or Boosting Techniques
- Split of Data observations or attributes
- May be built with different hyperparameters as well
- Adv – better accuracy, lower overfitting
- Drawback – high computation, slower



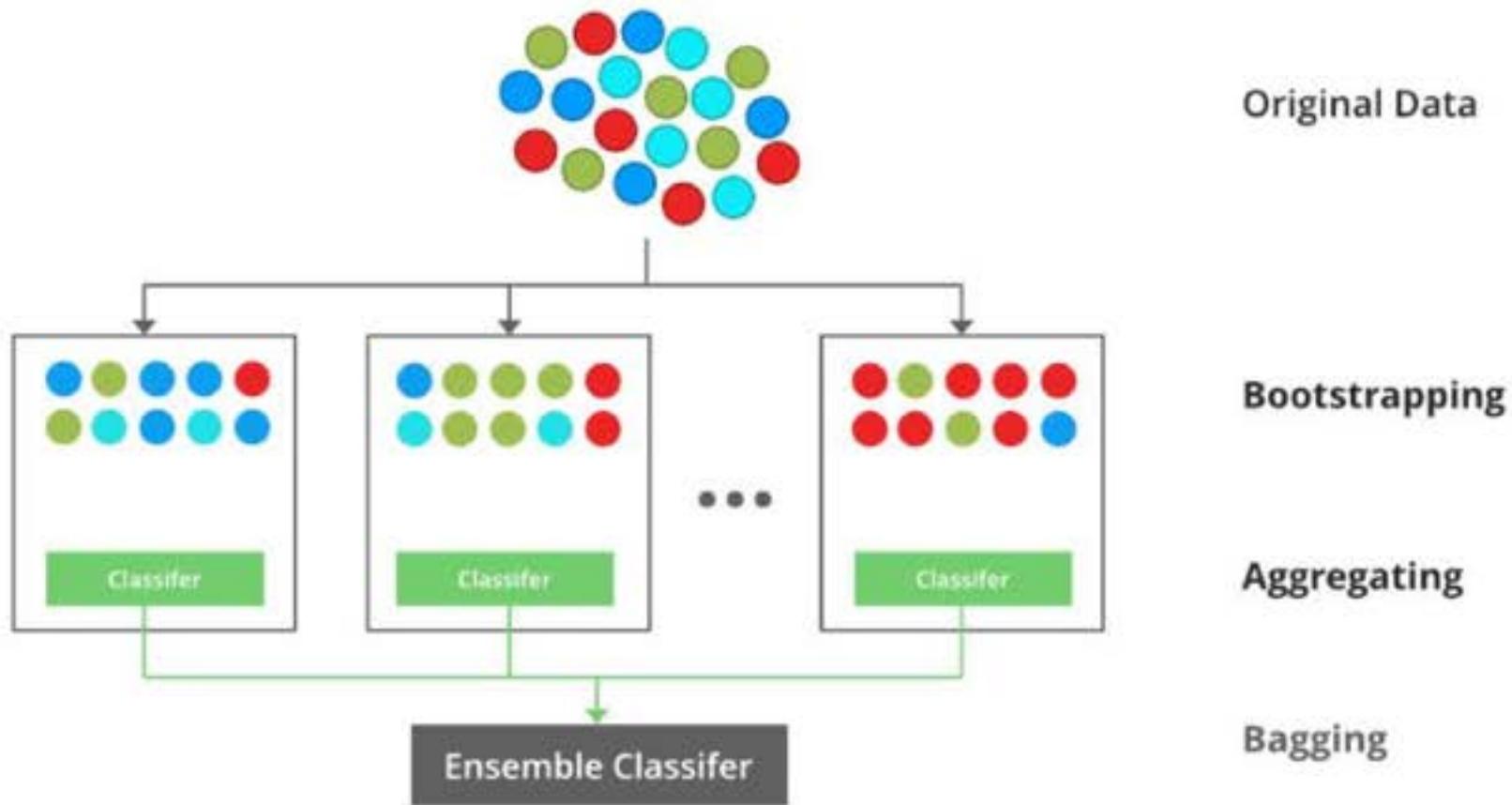
<https://www.hindawi.com/journals/mpe/2019/4140707/>

- helps improve machine learning results by combining several models.
- This approach allows the production of better predictive performance compared to a single model.
- Basic idea is to learn a set of classifiers (experts) and to allow them to vote.
- **Bagging** and **Boosting** are two types of **Ensemble Learning**.
- Ensemble learning helps improve machine learning results by combining several models.
- These two decrease the variance of a single estimate as they combine several estimates from different models. So the result may be a model with higher stability.

- **Bagging:** It is a homogeneous weak learners' model that learns from each other independently in parallel and combines them for determining the model average.
- **Boosting:** It is also a homogeneous weak learners' model but works differently from Bagging. In this model, learners learn sequentially and adaptively to improve model predictions of a learning algorithm.

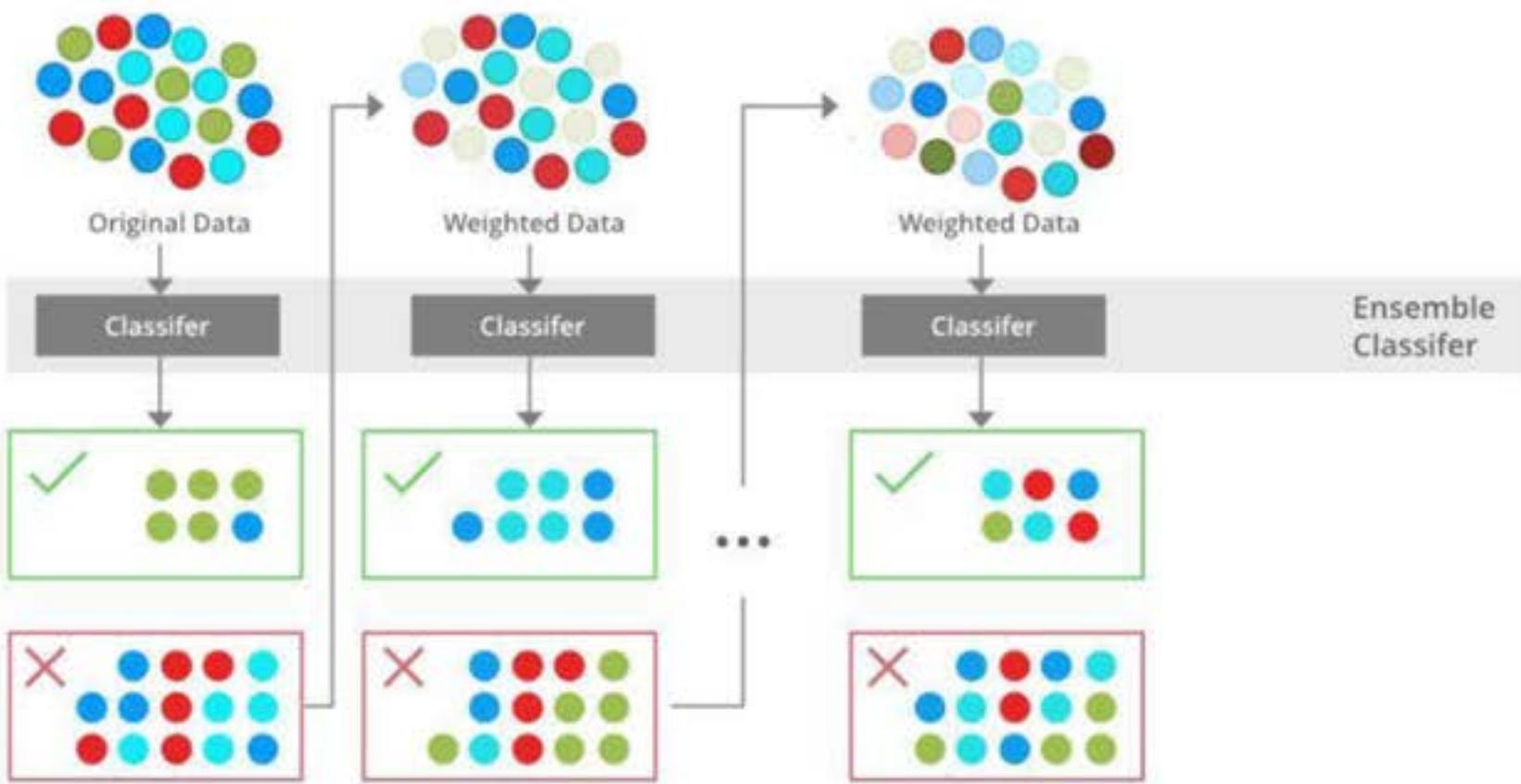
- **Bagging**
- **Bootstrap Aggregating**, also known as bagging, is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It decreases the variance and helps to avoid overfitting. It is usually applied to decision tree methods. Bagging is a special case of the model averaging approach.

- **Implementation Steps of Bagging**
- **Step 1:** Multiple subsets are created from the original data set with equal tuples, selecting observations with replacement.
- **Step 2:** A base model is created on each of these subsets.
- **Step 3:** Each model is learned in parallel with each training set and independent of each other.
- **Step 4:** The final predictions are determined by combining the predictions from all the models.
-



- **Boosting**
- Boosting is an ensemble modeling technique that attempts to build a strong classifier from the number of weak classifiers. It is done by building a model by using weak models in series. Firstly, a model is built from the training data. Then the second model is built which tries to correct the errors present in the first model. This procedure is continued and models are added until either the complete training data set is predicted correctly or the maximum number of models is added.

- Initialise the dataset and assign equal weight to each of the data point.
- Provide this as input to the model and identify the wrongly classified data points.
- Increase the weight of the wrongly classified data points and decrease the weights of correctly classified data points. And then normalize the weights of all data points.
- if (got required results)
 Goto step 5
else
 Goto step 2
- End



An illustration presenting the intuition behind the boosting algorithm, consisting of the parallel learners and weighted dataset.

Similarities Between Bagging and Boosting

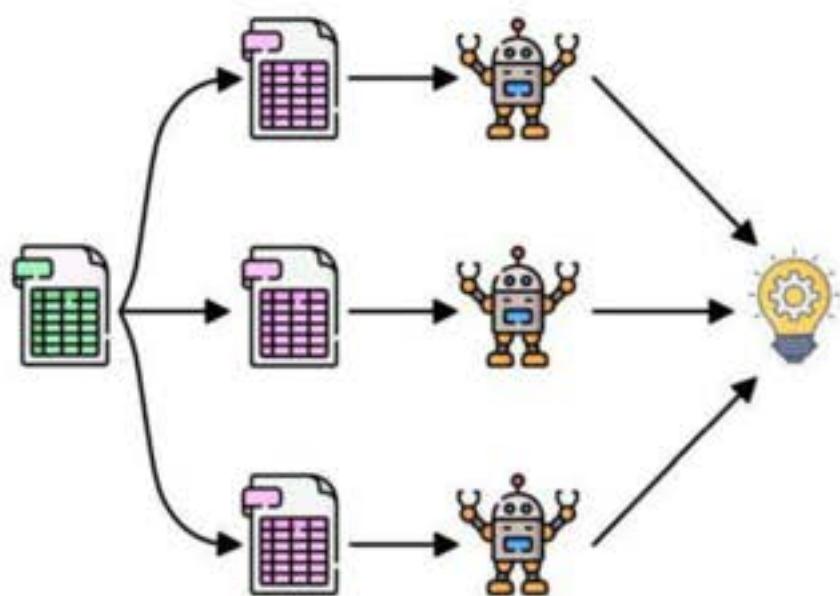
Bagging and Boosting, both being the commonly used methods, have a universal similarity of being classified as ensemble methods. Here we will explain the similarities between them.

1. Both are ensemble methods to get N learners from 1 learner.
2. Both generate several training data sets by random sampling.
3. Both make the final decision by averaging the N learners (or taking the majority of them i.e Majority Voting).
4. Both are good at reducing variance and provide higher stability.

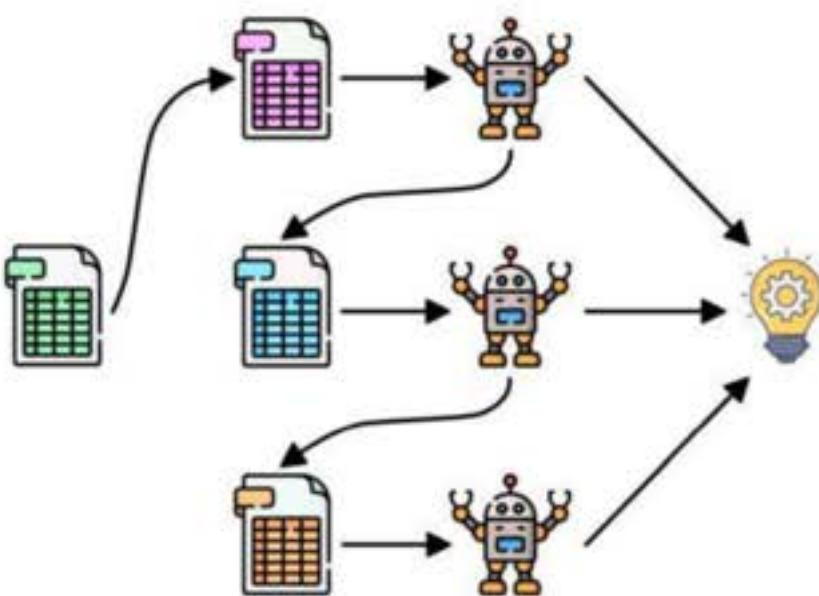


Random Forest: Bagging vs Boosting

Bagging



Boosting



Parallel

Sequential

Conclusion

- Decision Trees
 - Anatomy / structure
 - Impurity measures
 - Pruning
- Regression Trees
- Random Forest
- Demos

Thank you !!!!!



Machine Learning (19CSE305)

Markov Models, HMM



**Dr. Peeta Basa Pati
Ms. Priyanka V**
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- States and transition probabilities
- Markov Models
- HMM's

A very good reference book on this module:

Pattern Recognition – An Introduction by Susheela Devi & Narasimha Murthy

Images used in these slides are taken from multiple sources. All authors are acknowledged.

Intro

- Classifiers so far – feature vectors are unrelated
- What if they are?
 - If it has rained yesterday, its likely to rain today
 - In IPL if a team has been winning, like to win in the next match
 - If a student has been scoring good grades, likely to score good grade this semester
 - Covid cases are decreasing for last 1 week, likely to decrease today
 - A restaurant has been serving good food in your last 3 visits, likely to serve good food now
 - A person has allergy during Diwali time last 3 years, likely to have allergy this time
- Past events determine the future in many scenarios
- We could leverage the underlying pattern to decide better

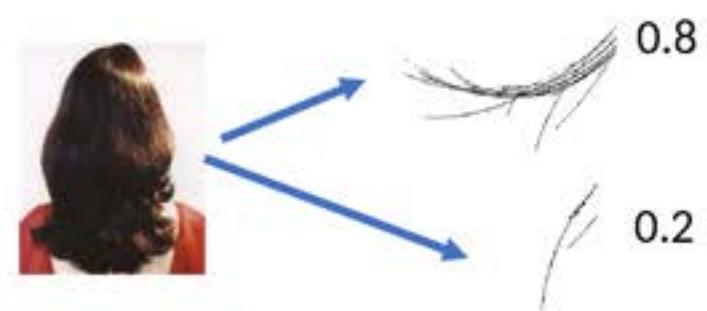
Observable & Hidden states

Assume that we measure a person's hair length and try to predict the gender.

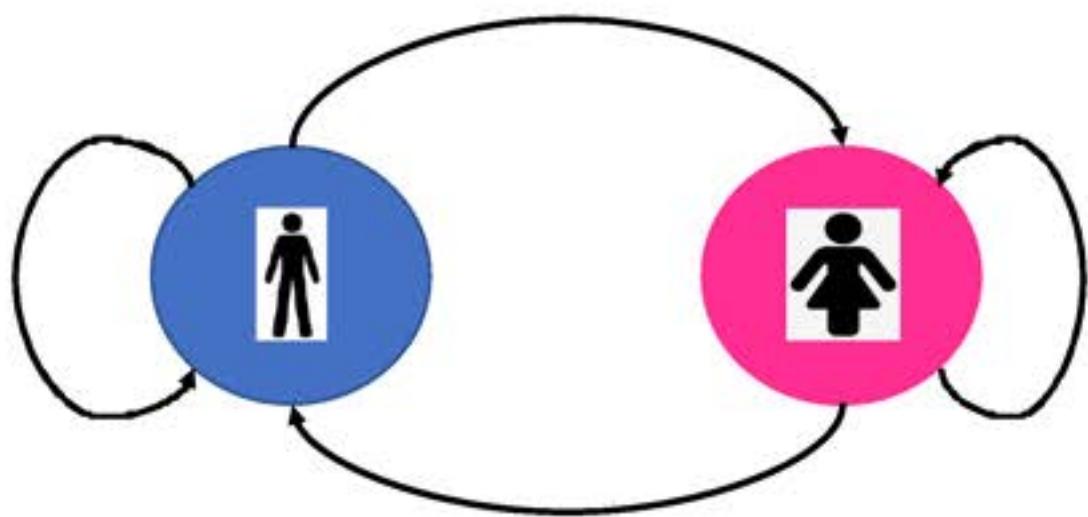
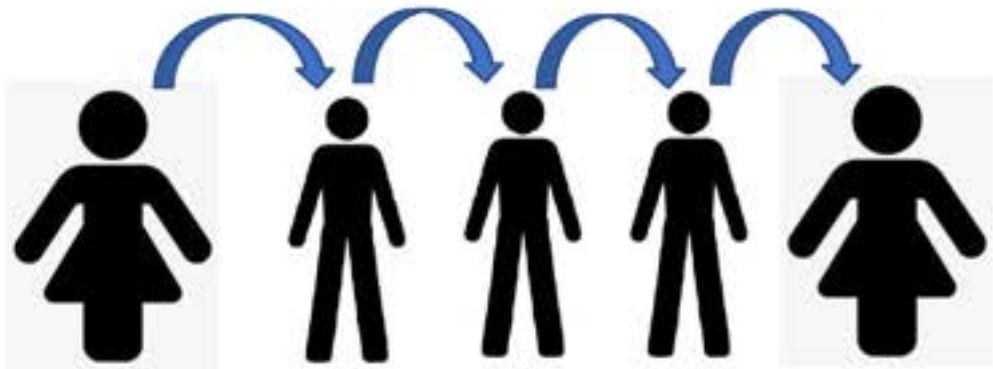


Prior probabilities are probabilities of states. Ex: probability of a person being male or female. Male = 0.55; Female = 0.45

Class conditional densities are observations associated with a class. Ex: hair being short given the person is a female. Male = {0.95, 0.05}; Female = {0.2, 0.8}



State Transition Probabilities



	Female	Male
Female	p_{ff}	p_{fm}
Male	p_{mf}	p_{mm}

Bayes Classifier

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

Posterior probability

Class Conditional Density

Prior probability

$$\underset{\text{Maximum a posteriori probability}}{\max \{P(y_j | \mathbf{X})\}} \quad (= \arg \max_{y_j} P(y_j | \mathbf{X}))$$

y_j are different classes

Markov Process

$$\{s_1, s_2, \square, s_N\}$$

Set of possible states: Male/ Female, Rain / Dry

$$s_{i1}, s_{i2}, \square, s_{ik}, \square$$

Sequence of states for i^{th} observation

$$P(s_{ik} | s_{i1}, s_{i2}, \square, s_{ik-1}) = P(s_{ik} | s_{ik-1})$$

Markov process (first order)
probability of a state depends only on the previous state.

$$\pi_i = P(s_i)$$

Initial / Prior Probability

$$a_{ij} = P(s_i | s_j)$$

State Transition Probability

Significance of i variation
Probabilities may change based on time or place

Sequence Probability

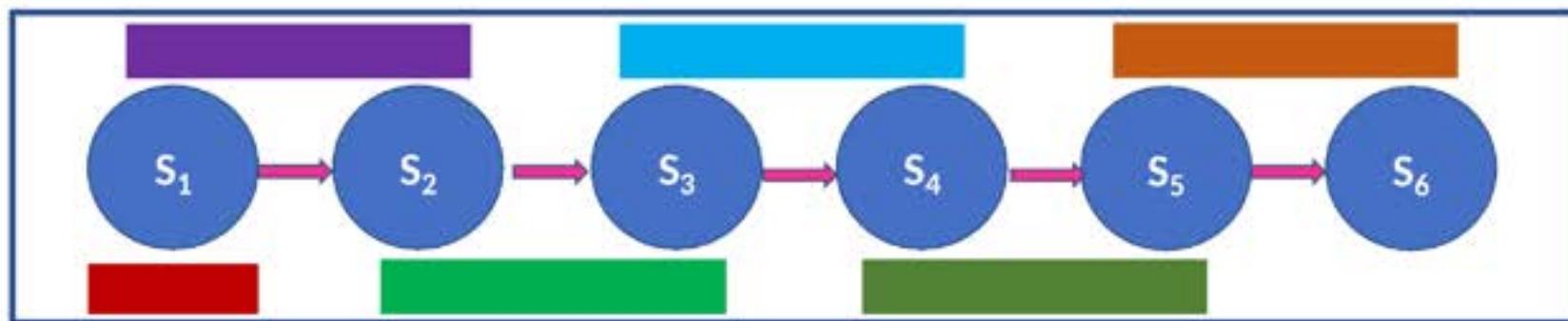
{ s_1, s_2, \dots, s_k }

Set of possible states: Male/ Female, Rain / Dry

$s_{i1}, s_{i2}, \dots, s_{ik}$

Sequence of states for i^{th} observation

$$P(s_{ik} | s_{i1}, s_{i2}, \dots, s_{ik-1}) = P(s_{ik} | s_{ik-1}) \quad \text{Markov process (first order)}$$



$$P(s_{i1}, s_{i2}, \dots, s_{ik}) = P(s_{ik} | s_{i1}, s_{i2}, \dots, s_{ik-1})P(s_{i1}, s_{i2}, \dots, s_{ik-1})$$

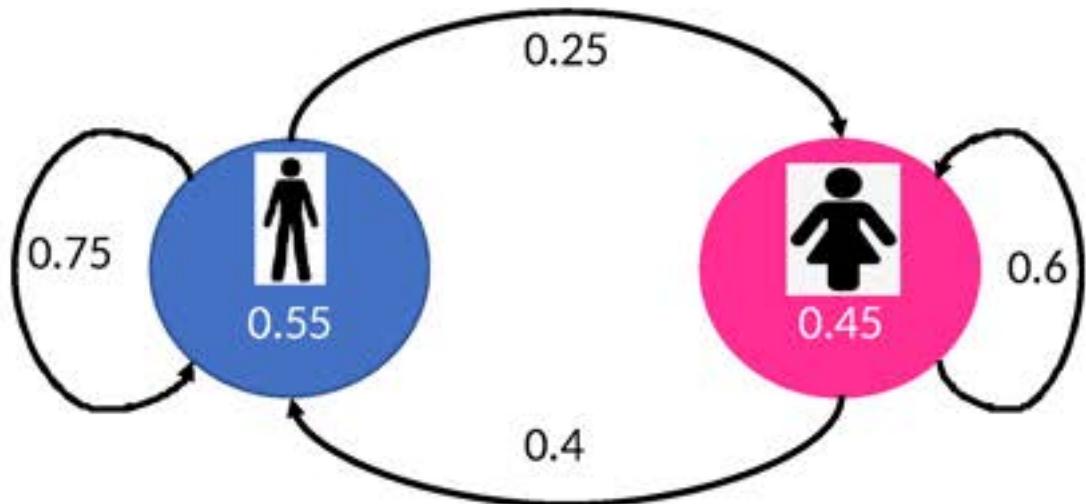
$$= P(s_{ik} | s_{ik-1})P(s_{i1}, s_{i2}, \dots, s_{ik-1}) = \square$$

$$= P(s_{ik} | s_{ik-1})P(s_{ik-1} | s_{ik-2}) \square P(s_{i2} | s_{i1})P(s_{i1})$$

Example

What is the Probability for {Girl, Boy}?

$$\begin{aligned} P(\{\text{Girl, Boy}\}) &= P(\text{Boy} \mid \text{Girl}) * P(\text{Girl}) \\ &= 0.4 * 0.45 = 0.18 \end{aligned}$$



$P(\{\text{Boy, Girl, Girl, Boy}\})??$

$$\begin{aligned} P(\{\text{Boy, Girl, Girl, Boy}\}) &= P(\text{Boy} \mid \{\text{Girl, Girl, Boy}\}) * P(\text{Girl} \mid \{\text{Girl, Boy}\}) * \\ &\quad P(\text{Girl} \mid \text{Boy}) * P(\text{Boy}) \\ &= P(\text{Boy} \mid \text{Girl}) * P(\text{Girl} \mid \text{Girl}) * P(\text{Girl} \mid \text{Boy}) * P(\text{Boy}) \\ &= 0.4 * 0.6 * 0.25 * 0.55 = 0.033 \end{aligned}$$

References

- <https://www.youtube.com/watch?v=kqSzLo9fenk>
- <https://cedar.buffalo.edu/~govind/CS661/Lec12.ppt>
- <https://nlp.stanford.edu/fsnlp/hmm-chap/blei-hmm-ch9.ppt>
- Pattern Recognition – An Introduction by Susheela Devi & MNM
- Pattern Recognition – Duda, Hart & Stork

Thank you !!!!



Machine Learning (19CSE305)

Confusion Matrix, ROC, F-Score



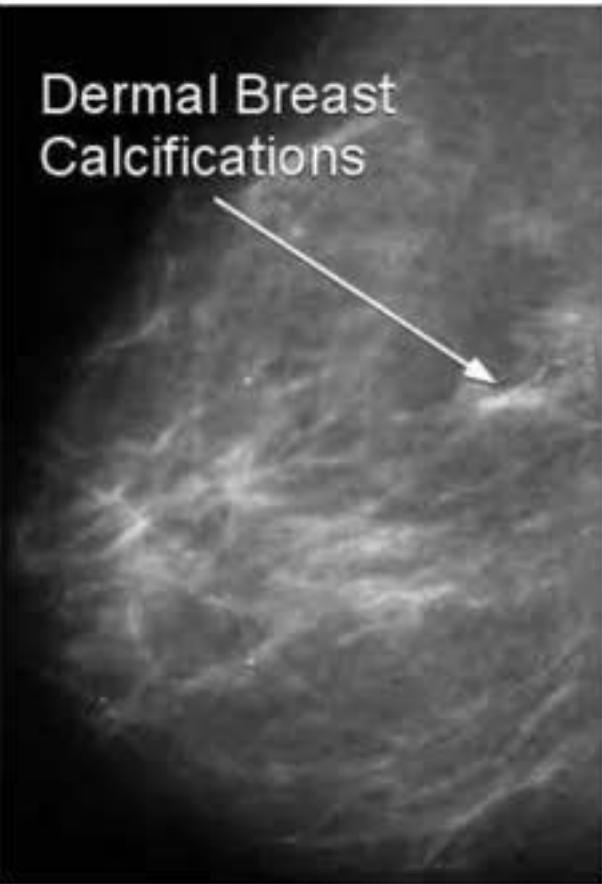
Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Confusion Matrix
- Scores to measure efficiency
- ROC & AUC

Example Cases

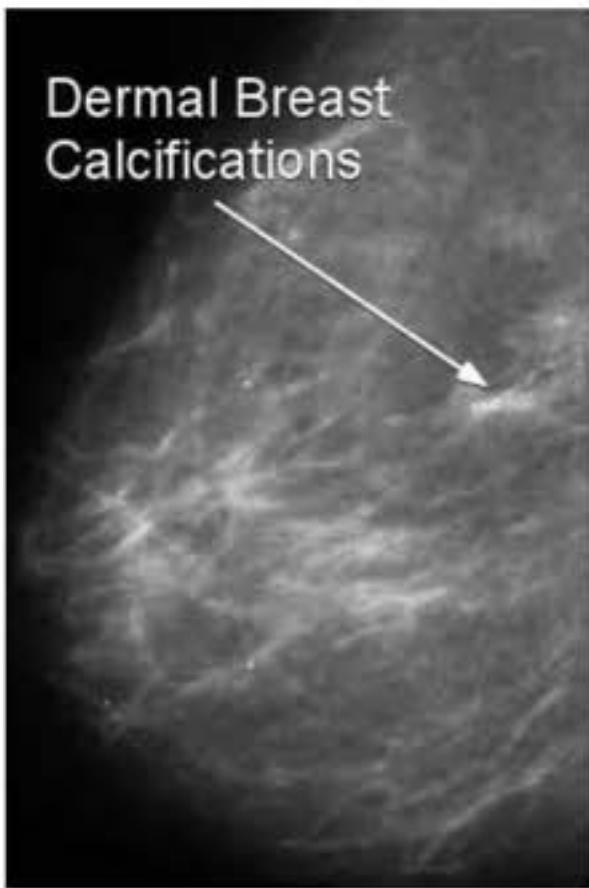
Dermal Breast
Calcifications



4 7 5
↓ ↓ ↓
4 7 5

Source: Internet

Mammogram Analysis



- Calcification Present → +ve class
- Calcification Absent → -ve class

One in twenty-eight Indian women is likely to develop breast cancer during her lifetime. -[Statistics of Breast Cancer In India | Cytecare Hospitals](#)

Scenario	Calcification - Actual	Calcification - Detection
1	Yes	Yes
2	Yes	No
3	No	Yes
4	No	No

		DETECTION	
		YES	NO
PRESENCE	YES	TRUE POSITIVE	FALSE NEGATIVE
	NO	FALSE POSITIVE	TRUE NEGATIVE

Are the costs same?

Other Scenarios

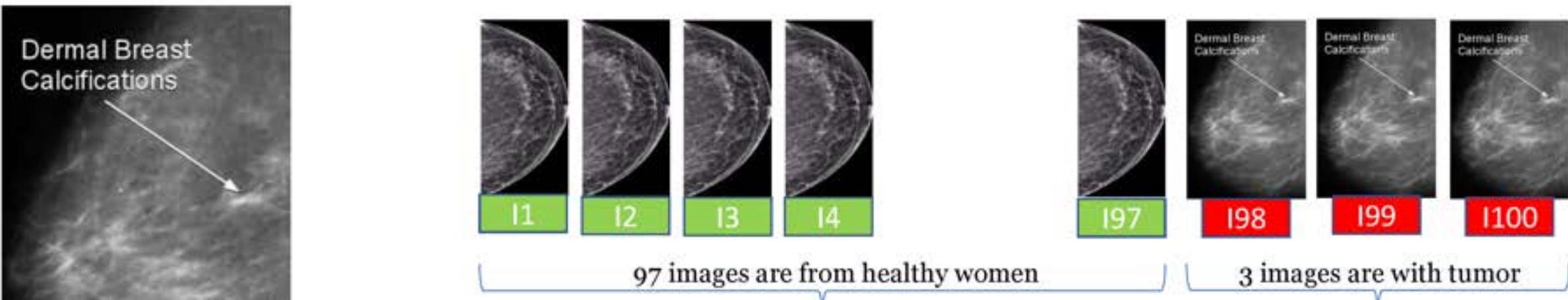
COVID Infection

		DETECTION	
		YES	NO
PRESENCE	YES	TRUE POSITIVE	FALSE NEGATIVE
	NO	FALSE POSITIVE	TRUE NEGATIVE

Spam Mail

		DETECTION	
		YES	NO
PRESENCE	YES	TRUE POSITIVE	FALSE NEGATIVE
	NO	FALSE POSITIVE	TRUE NEGATIVE

Concept of Accuracy



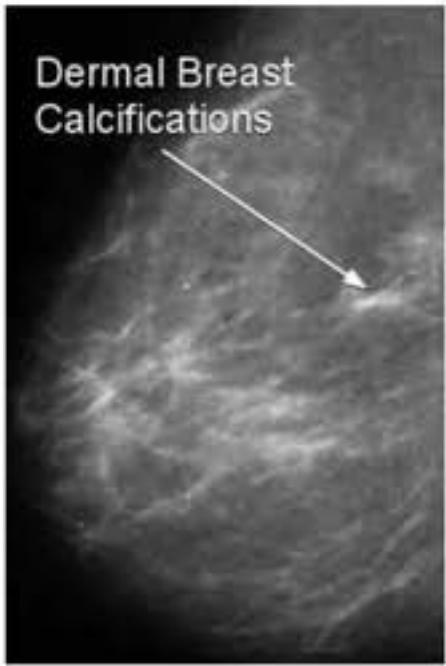
		DETECTION	
		YES	NO
PRESENCE	YES	TRUE POSITIVE (TP)	FALSE NEGATIVE (FN)
	NO	FALSE POSITIVE (FP)	TRUE NEGATIVE (TN)

- Declare all healthy → 97% accurate
- Declare all tumorous → 3% accurate

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

This notion of accuracy is fallacious since the costs associated with the decision are not same.

Precision & Recall



		DETECTION	
		YES	NO
PRESENCE	YES	27 (TP)	5 (FN)
	NO	18 (FP)	950 (TN)

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{27}{27+18} = 60\%$$

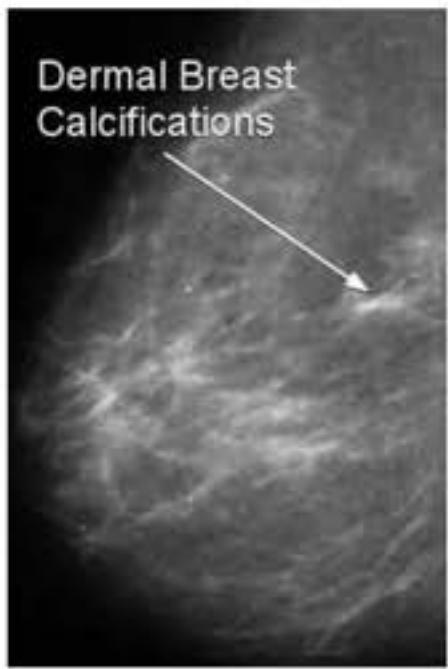
$$\text{Recall} = \frac{TP}{TP+FN} = \frac{27}{27+5} = 84\%$$

(Sensitivity)

$$\text{Specificity} = \frac{TN}{TN+FP} = \frac{950}{950+18} = 98\%$$

How do we measure these with a single number?

Precision & Recall



DETECTION	
PRESENCE	YES
YES	27 (TP)
NO	18 (FP)
TN	
950 (TN)	

Precision = 60%

Recall = 84%

How do we measure these with a single number?

We can take Average of the numbers →
Avg = (Precision + Recall) / 2 = 72%

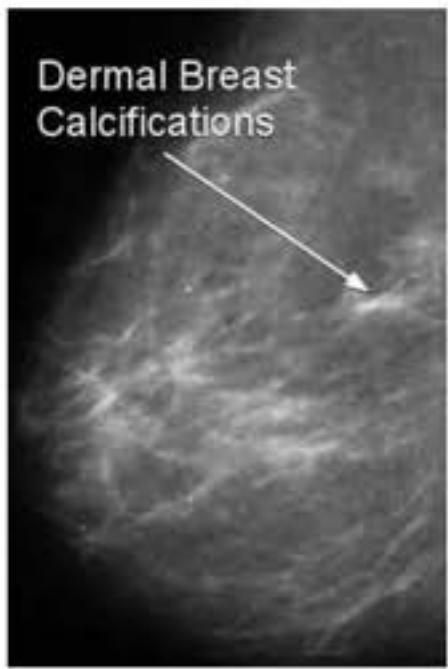
$$\begin{aligned}\text{Harmonic Mean} &= \frac{2XY}{X+Y} \\ &= \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \\ &= 70\%\end{aligned}$$



Also called as F1-Score

$$F_{\beta} = (1 + \beta^2) \frac{\text{Precision} * \text{Recall}}{(\beta^2 * \text{Precision}) + \text{Recall}}$$

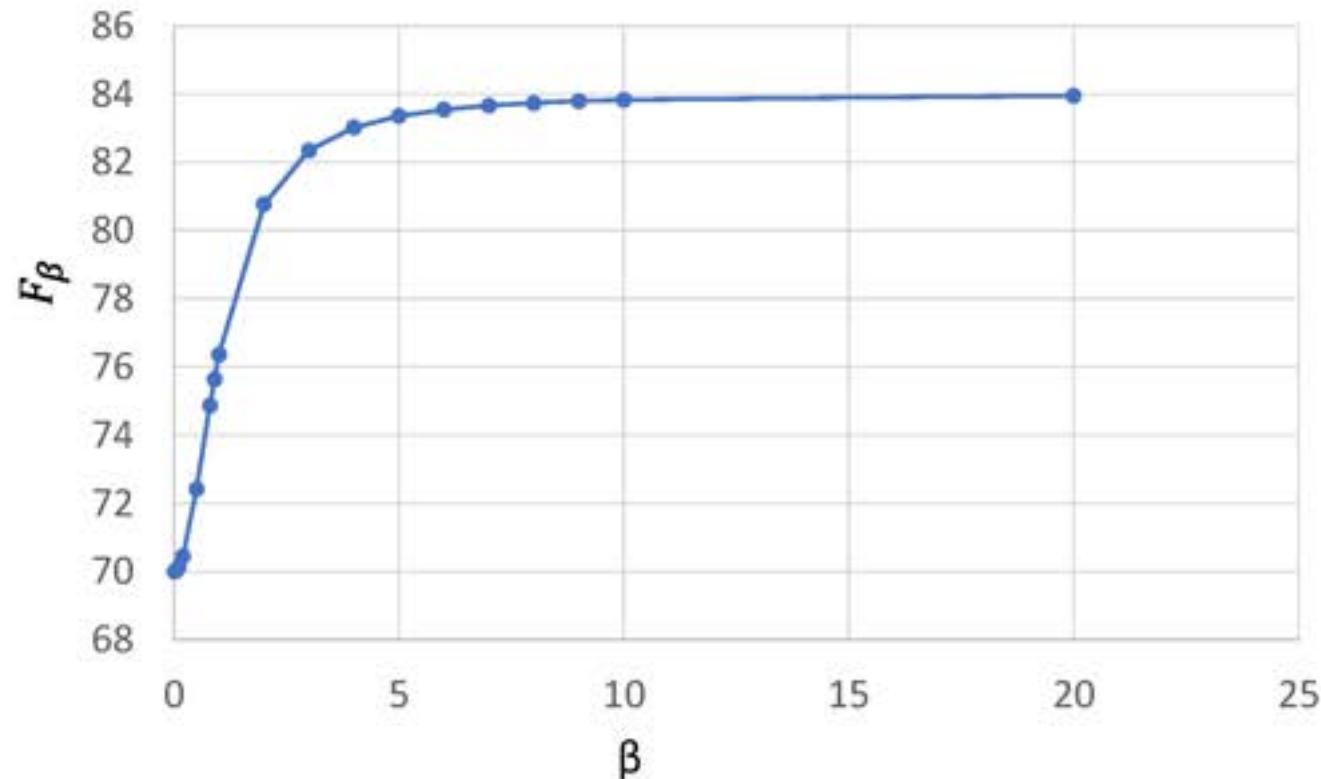
Precision & Recall



Precision = 60%

		DETECTION	
		YES	NO
PRESENCE	YES	27 (TP)	5 (FN)
	NO	18 (FP)	950 (TN)

Recall = 84%



$$F_\beta = \frac{Precision * Recall}{(\beta^2 * Precision) + Recall}$$

Error Types



DETECTION	
PRESENCE	YES
YES	27 (TP)
NO	18 (FP)
NO	950 (TN)
5 (FN)	

Type-I Error:

- Refers to error made in judgement leading to positive detection
- False Positives

Type-II Error:

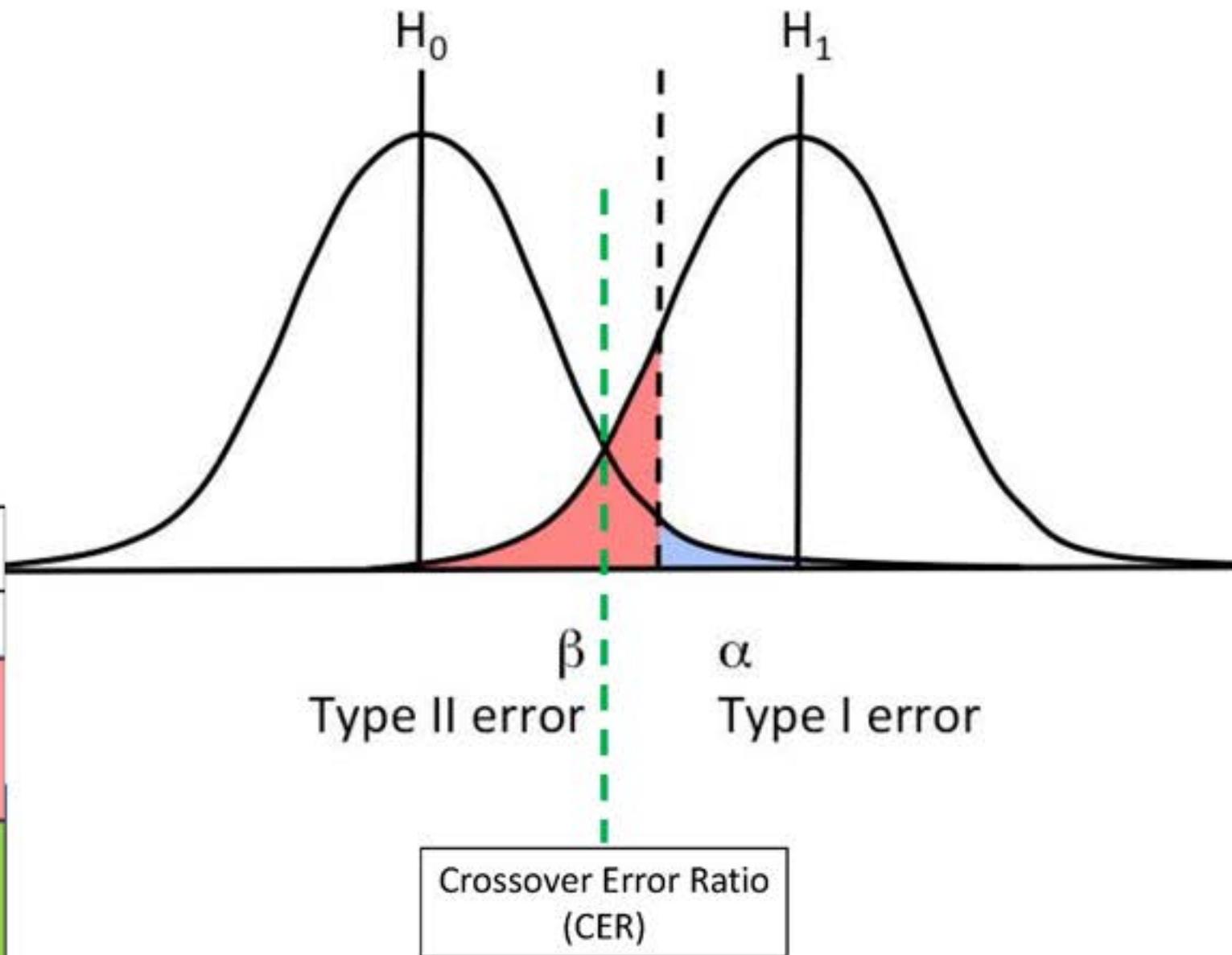
- Refers to error made in judgement leading to negative detection
- False Negatives

Error due to bias & error due to over-fitting

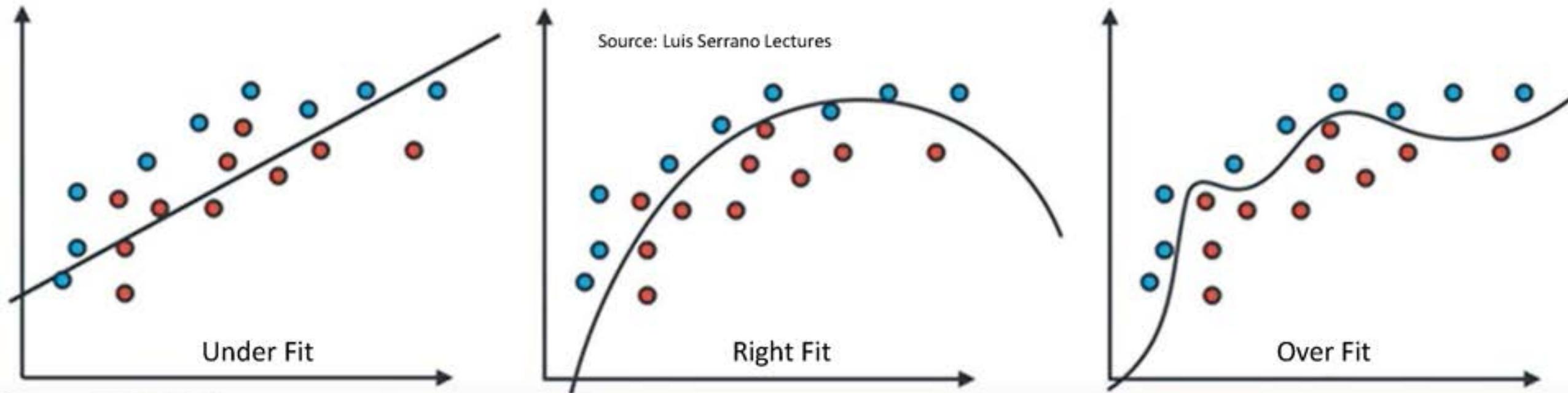
Error Types



DETECTION	
PRESENCE	YES
YES	27 (TP)
NO	18 (FP)
NO	950 (TN)



Model Complexity



Under Fit

Right Fit

Over Fit

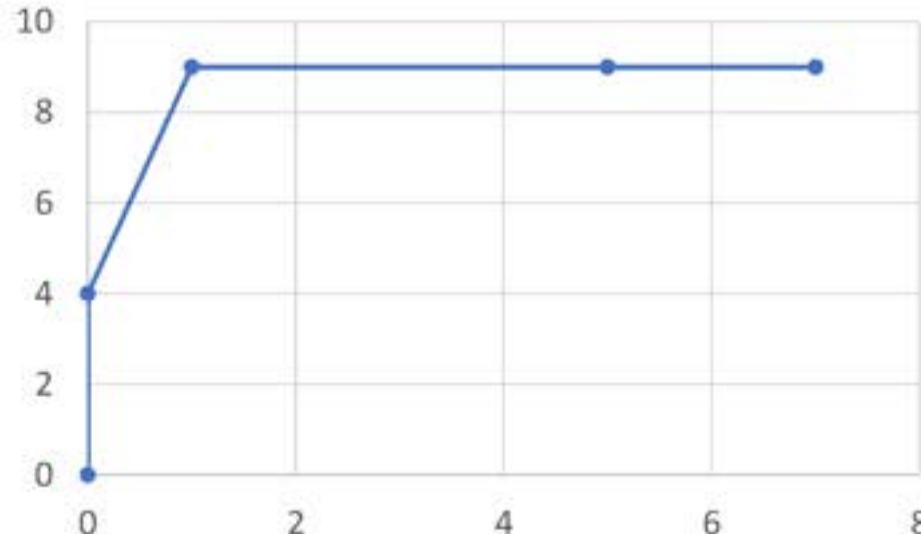
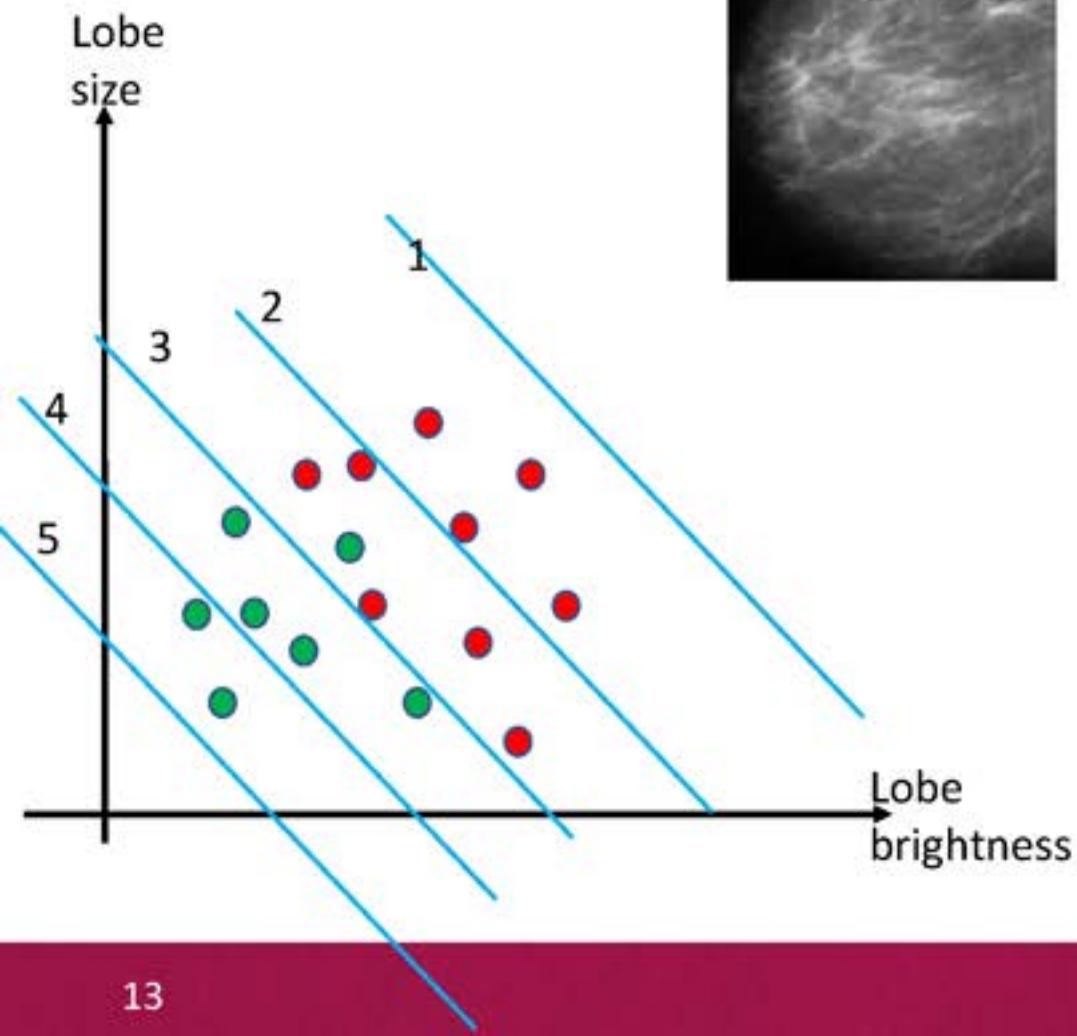


Under Fit – any bright spot (grey pixel value > 180) is malignant tissue

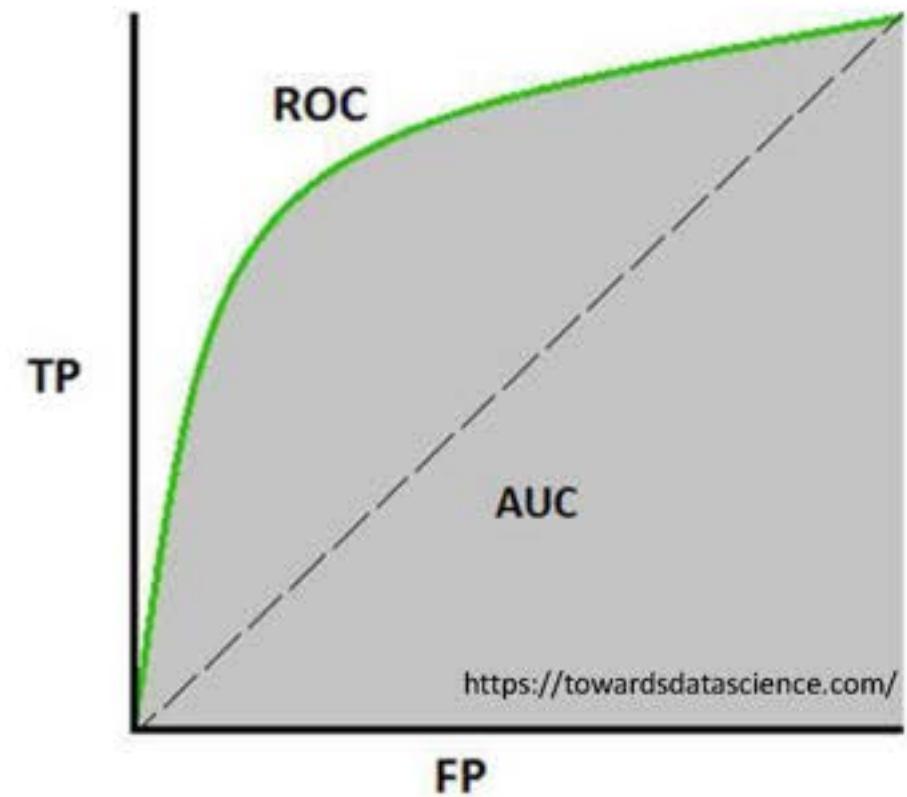
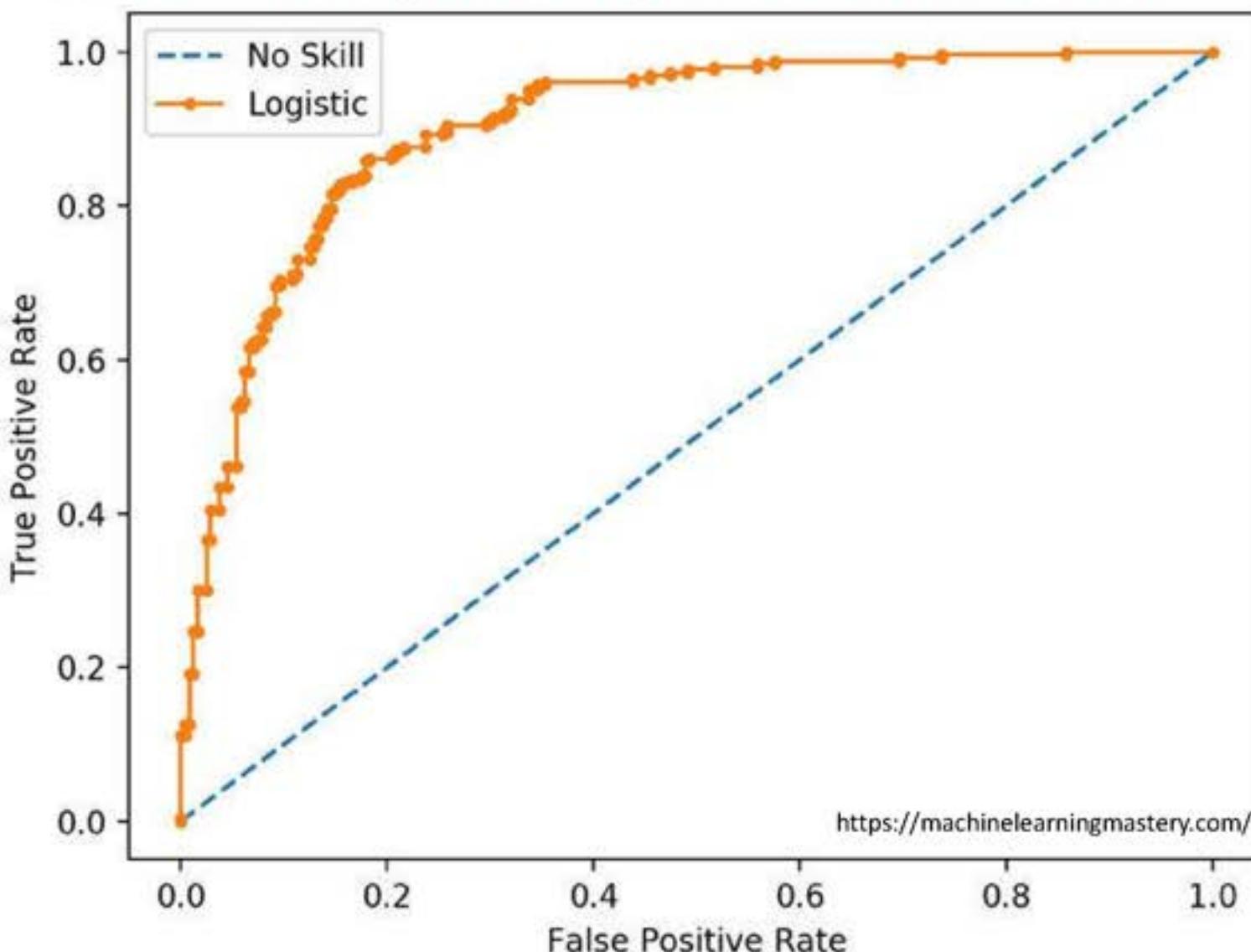
Right Fit – brightness, size, contrast with other similar objects

Over Fit – brightness, Contrast, elliptical shape, major/minor axis ratio, location, size etc.

Receiver Operating Characteristic (ROC) Curve



Receiver Operating Characteristic (ROC) Curve, AUC



Thank you !!!!!



Machine Learning (19CSE305)

Feature Selection



Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Feature extraction vs selection
- Search techniques
- Criterion functions

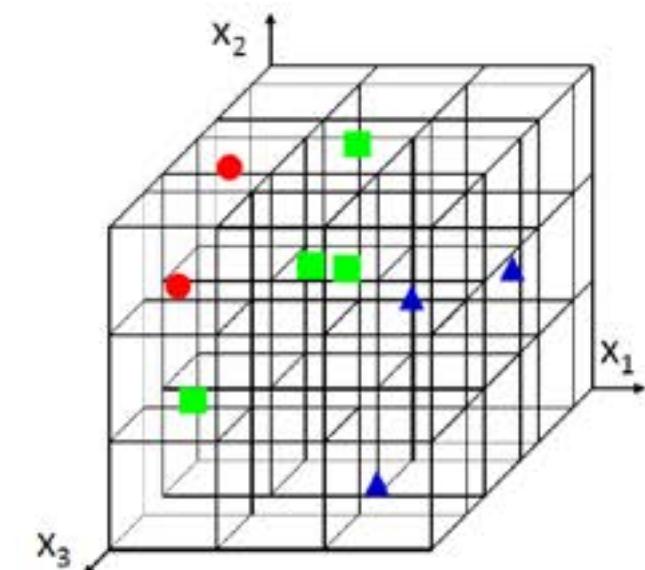
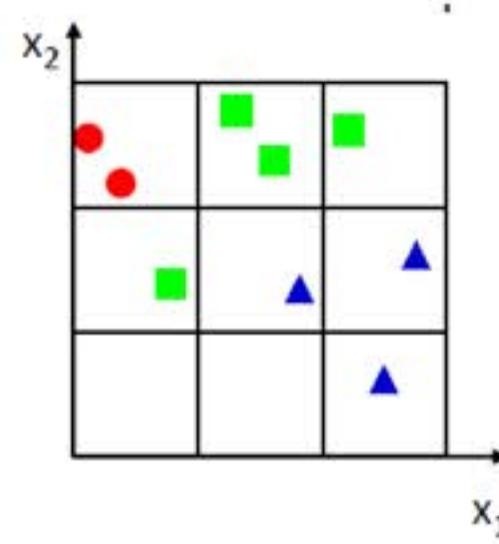
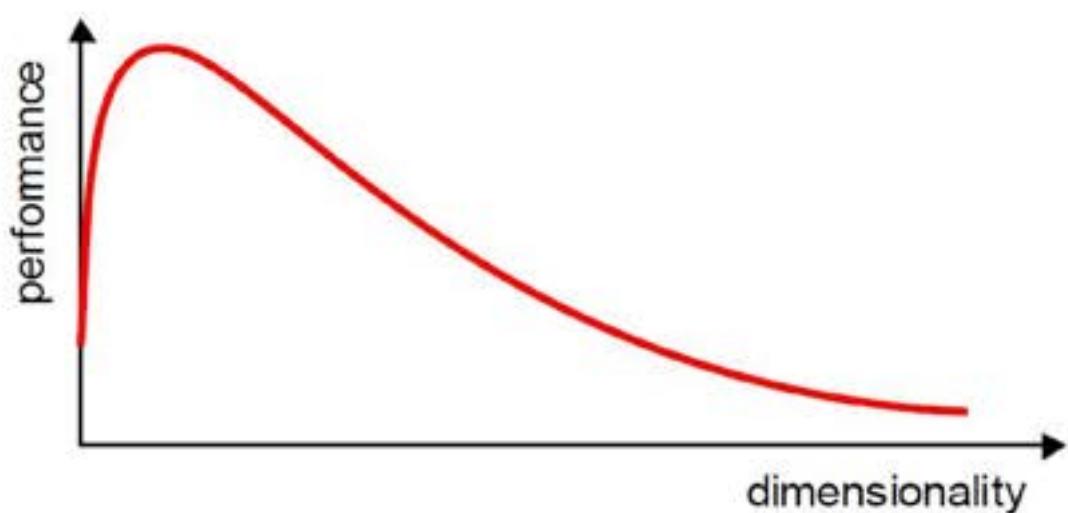
A very good reference book on this module:

Pattern Recognition 4th Edition by Theodoridis and Koutroumbas

Images used in these slides are taken from multiple sources. All authors are acknowledged.

Curse of Dimensionality

- Increasing features initially increases accuracy but deteriorates after some point
- The number of training examples required increases **exponentially** with dimensionality \mathbf{d} (i.e., k^d).
- Other curses – visualization & performance challenges



Ref: www.cse.unr.edu/~bebis/CS479/Lectures/DimensionalityReduction.ppt

Feature Extraction

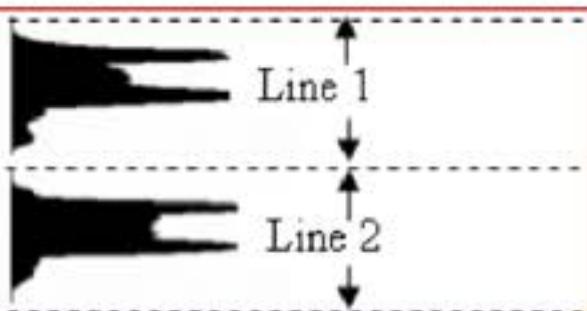
Extract a new set of features through a mapping function

$$\mathbf{y} = f(\mathbf{X})$$

Ex: projection profiles, DCT, MFCC etc.

Linear or non-linear combination

ದೀಪಾವಳಿ ಹಬ್ಬ ಎಲ್ಲ
ಕನಾಡಾಟಕ ರಾಜೀನ್ಯೋತ್ಸ

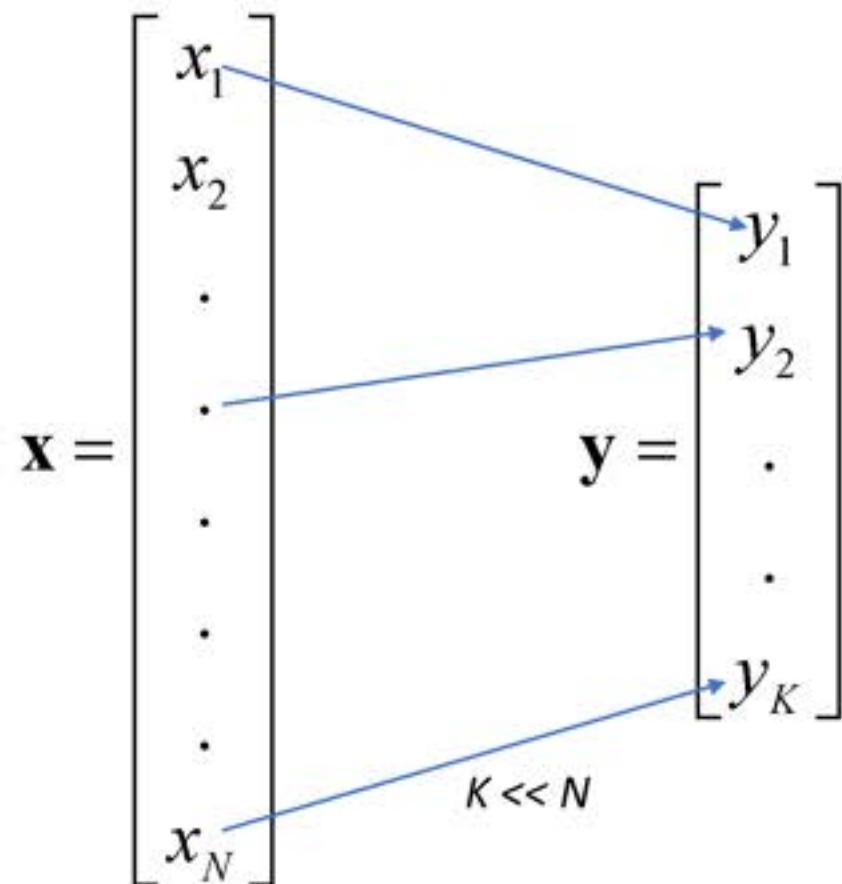


$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_K \end{bmatrix}$$

$K \ll N$

DOI: 10.1007/s12046-007-0039-1

Feature Selection



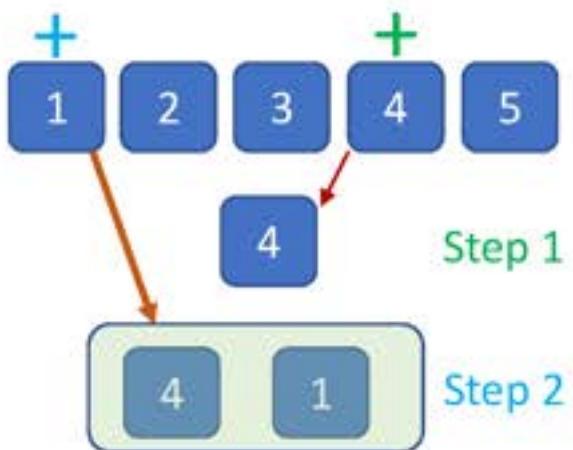
- Select an optimal subset of features from the original feature set
 - Search problem
 - Search Algorithms
 - Sequential forward search / generation
 - Sequential backward search / removal
 - Bidirectional search
 - Sequential forward-backward search
 - Selection criterion
 - Entropy
 - Information Gain
 - Accuracy of classification etc.

Sequential forward / backward search

Algorithm 1 Sequential forward feature set generation - SFG.

```
function SFG( $F$  - full set,  $U$  - measure)
    initialize:  $S = \{\}$ 
    repeat
         $f = \text{FINDNEXT}(F)$ 
         $S = S \cup \{f\}$ 
         $F = F - \{f\}$ 
    until  $S$  satisfies  $U$  or  $F = \{\}$ 
    return  $S$ 
end function
```

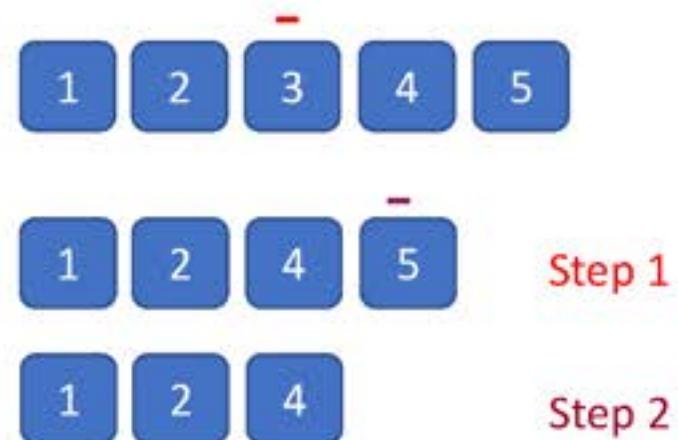
▷ S stores the selected features



Algorithm 2 Sequential backward feature set generation - SBG.

```
function SBG( $F$  - full set,  $U$  - measure)
    initialize:  $S = \{\}$ 
    repeat
         $f = \text{GETNEXT}(F)$ 
         $F = F - \{f\}$ 
         $S = S \cup \{f\}$ 
    until  $S$  does not satisfy  $U$  or  $F = \{\}$ 
    return  $F \cup \{f\}$ 
end function
```

▷ S holds the removed features



Bidirectional search

- Begins the search in both directions – SF & SB simultaneously
- Stop if:
 - one search finds optimal subset before it reaches the middle
 - both searches reach middle; select the better of 2 subsets from SF & SB search

Algorithm 3 Bidirectional feature set generation - BG.

```
function BG( $F_f$ ,  $F_b$  - full set,  $U$  - measure)
    initialize:  $S_f = \{\}$                                  $\triangleright S_f$  holds the selected features
    initialize:  $S_b = \{\}$                                  $\triangleright S_b$  holds the removed features
    repeat
         $f_f = \text{FINDNEXT}(F_f)$ 
         $f_b = \text{GETNEXT}(F_b)$ 
         $S_f = S_f \cup \{f_f\}$ 
         $F_b = F_b - \{f_b\}$ 
         $F_f = F_f - \{f_f\}$ 
         $S_b = S_b \cup \{f_b\}$ 
    until (a)  $S_f$  satisfies  $U$  or  $F_f = \{\}$  or (b)  $S_b$  does not satisfy  $U$  or  $F_b = \{\}$ 
    return  $S_f$  if (a) or  $F_b \cup \{f_b\}$  if (b)
end function
```

Sequential Forward-Backward Search

- Start with either SF or SB search; accuracy is criterion function
- If SF:
 - Start by searching for the feature that gives maximum accuracy
 - Add a feature
 - Check if removal of any (subset) feature(s) improves accuracy; remove
 - Continue
- If SB:
 - Start by searching for the feature that gives minimum accuracy
 - Remove a feature
 - Check if addition of any removed features improves accuracy; add
 - Continue
- May get into loop if proper care is not taken
 - Ex: for SF → same feature gets added and removed; will form a loop
- Modified version – continue adding or removing till optimal set is attained

Thank you !!!!!



Machine Learning (19CSE305)

PCA/SVD



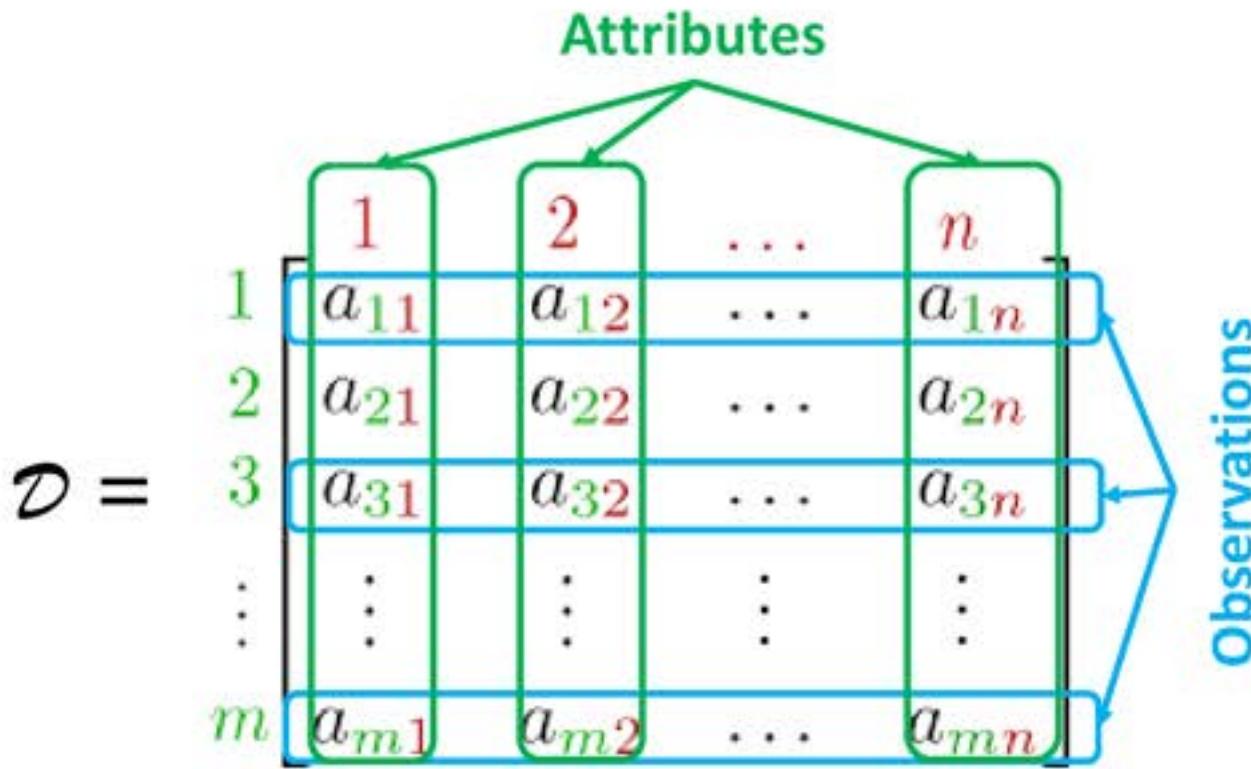
Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Covariance matrix
- Eigen values & vectors
- Principal components
- PCA & SVD
- Dimensionality reduction

Images used in these slides are taken from multiple sources. All authors are acknowledged.

Data



Mean, Variance & Covariance

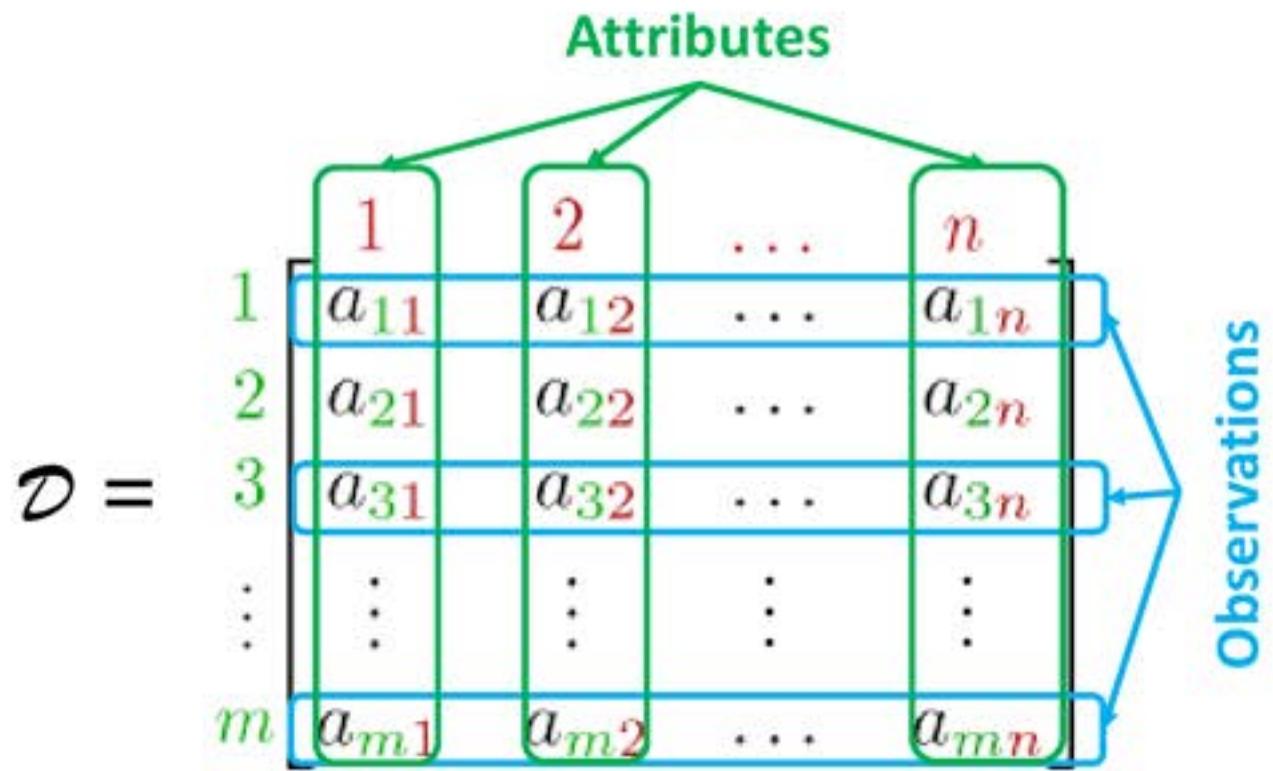
X & Y are random variables;

$$X = \{X_1, X_2, \dots, X_m\} \quad Y = \{Y_1, Y_2, \dots, Y_m\}$$

$$E[X] = \mu(X) = \frac{1}{N} \sum_{k=0}^n X_k \quad \sigma^2(X) = E[(X - \mu(X))^2] = \frac{1}{N} \sum_{k=0}^n [X_k - \mu(X)]^2$$

$$E[Y] = \mu(Y) = \frac{1}{N} \sum_{k=0}^n Y_k \quad COV(X, Y) = E[(X - \mu(X))(Y - \mu(Y))] \\ = \frac{1}{N} \sum_{k=0}^n [X_k - \mu(X)][Y_k - \mu(Y)]$$

Covariance Matrix



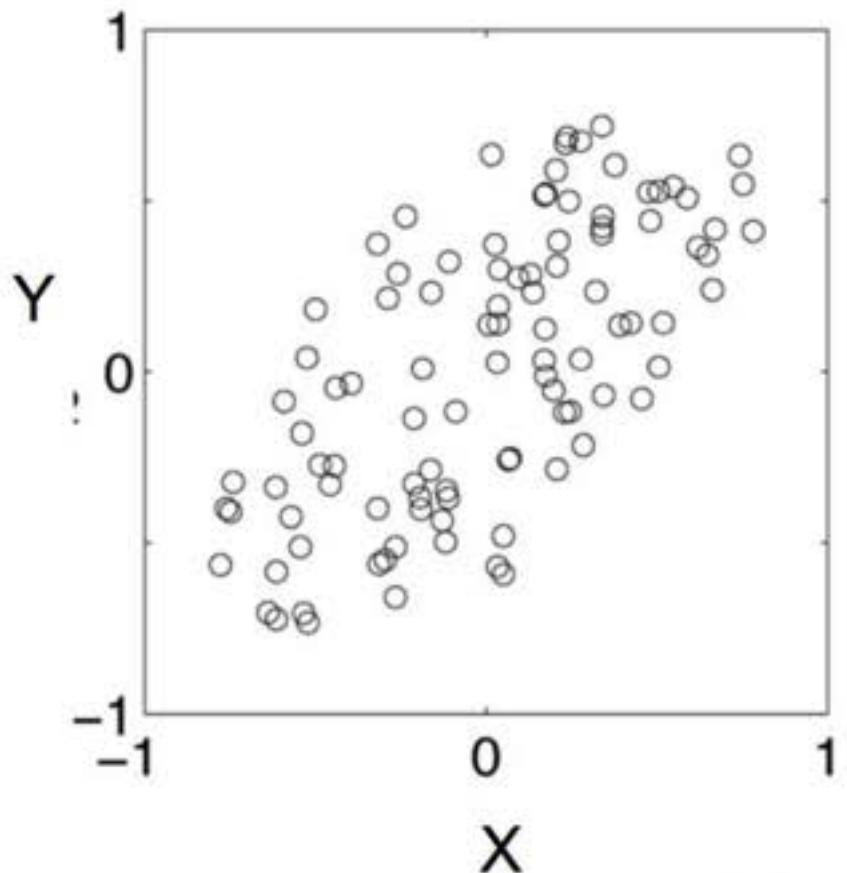
$$S = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1n} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_n^2 \end{bmatrix}$$

$$\sigma_{12} = \text{COV}(\text{Att}_1, \text{Att}_2)$$

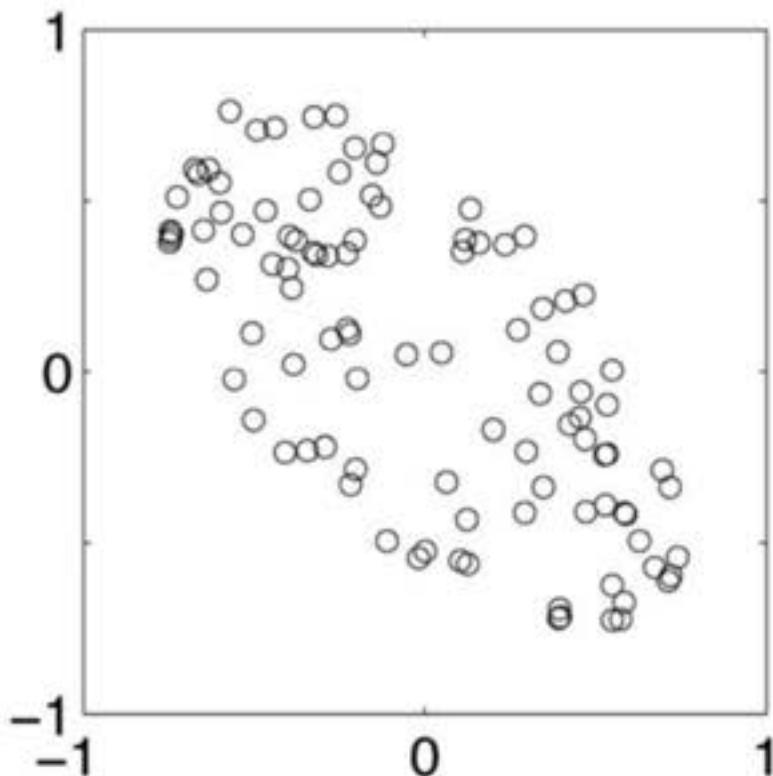
\mathcal{D} is the data matrix; S is the covariance matrix

Covariance Pattern

positive covariance



negative covariance



Fereshteh Sadeghi; CSEP 546, Univ. of Washington

Eigen Values & Vectors

$$\mathbf{S}\mathbf{v} = \lambda\mathbf{v}$$

(right) eigenvector eigenvalue

this is a n -th order equation in λ which can have **at most n distinct solutions** (roots of the characteristic polynomial) - can be complex even though \mathbf{S} is real.

$$\mathbf{S}\mathbf{v} = \lambda\mathbf{v} \iff (\mathbf{S} - \lambda\mathbf{I})\mathbf{v} = \mathbf{0}$$

Non-zero solution $\rightarrow |\mathbf{S} - \lambda\mathbf{I}| = 0$

$$\lambda \in \mathbb{R}$$

$$\{\mathbf{v} \neq \mathbf{0}\} \in \mathbb{R}^n$$

$$\mathbf{S} \in \mathbb{R}^{n \times n}$$

For symmetric matrices, eigenvectors for distinct eigenvalues are **orthogonal**

$$S\mathbf{v}_{\{1,2\}} = \lambda_{\{1,2\}}\mathbf{v}_{\{1,2\}}, \text{ and } \lambda_1 \neq \lambda_2 \implies \mathbf{v}_1 \bullet \mathbf{v}_2 = 0$$

All eigenvalues of a real symmetric matrix are **real**.

Eigen Values & Vectors - Example

$$S = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\lambda_1 = 3, \lambda_2 = 2 \text{ & } \lambda_3 = 0$$

$$v_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \quad v_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad v_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Any vector (say $x = \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix}$) can be viewed as a combination of the eigenvectors:

$$x = 2v_1 + 4v_2 + 6v_3$$

$$S = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

Real, symmetric.

$$S - \lambda I = \begin{bmatrix} 2 - \lambda & 1 \\ 1 & 2 - \lambda \end{bmatrix}$$

$$\Rightarrow (2 - \lambda)^2 - 1 = 0$$

$$\Rightarrow \lambda_1 = 1 \text{ & } \lambda_2 = 3$$

Real & non-negative

$$\begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Real, Orthogonal

Eigen Decomposition

Let $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$, $\lambda_i \geq \lambda_{i+1}$

$$\longrightarrow \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}$$

Let U be the matrix with eigenvectors as columns

Earlier: $Sv = \lambda v$

$$U = \begin{bmatrix} v_1 & \dots & v_n \end{bmatrix}$$

Then, SU can be written

$$SU = S \begin{bmatrix} v_1 & \dots & v_n \end{bmatrix} = \begin{bmatrix} \lambda_1 v_1 & \dots & \lambda_n v_n \end{bmatrix} = \begin{bmatrix} v_1 & \dots & v_n \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} \quad S \in \mathbb{R}^{n \times n}$$

$$\text{Thus } SU = U\Lambda, \quad \Rightarrow (SU) U^{-1} = (U\Lambda) U^{-1} \quad \Rightarrow S = U\Lambda U^{-1}$$

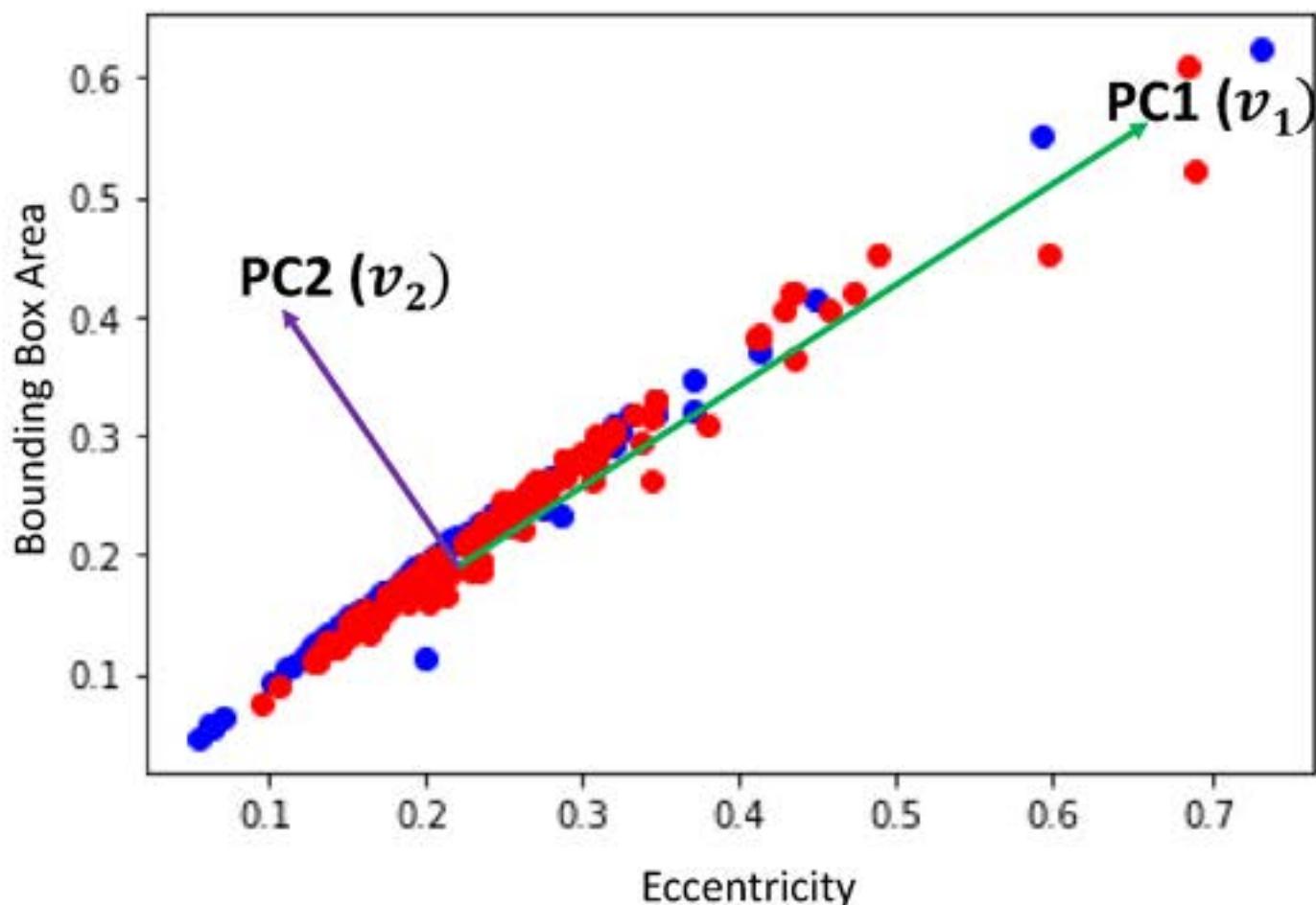
Goals of PCA

A goal of PCA is to find a transformation of the data that satisfies the following properties:

1. Each pair of new attributes has 0 covariance (for distinct attributes).
2. The attributes are ordered with respect to how much of the variance of the data each attribute captures.
3. The first attribute captures as much of the variance of the data as possible.
4. Subject to the orthogonality requirement, each successive attribute captures as much of the remaining variance as possible.

PCA

- PC1 – first principal component is the direction of maximum variability
- PC2 – 2nd principal component is the direction for 2nd most variability; is orthogonal to PC1



PCA Outcome

- We started with Data Matrix D (mean removed attributes)
- $S = D^T D = U \Lambda U^{-1}$
- Transformed Data Matrix: $D' = DU$
- Each new attribute is a linear combination of original attributes
 - Weights for combination for i^{th} attribute come from the i^{th} eigen vector (also the i^{th} PC)

SVD

$$A = \sum_{k=0}^{\text{rank}(A)} \sigma_k \mathbf{u}_k \mathbf{v}_k^T = U \Sigma V^T$$

Refer Theorem A.2; Introduction to Data Mining, Tan, Steinbach, Kumar

$$\mathbf{A} \in \mathbb{R}^{m \times n}$$

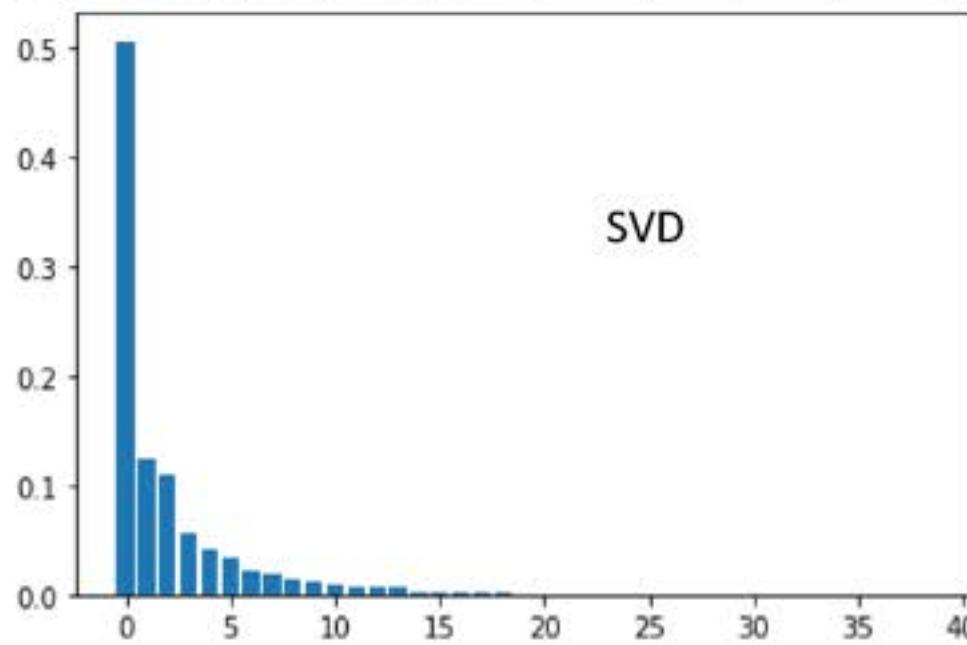
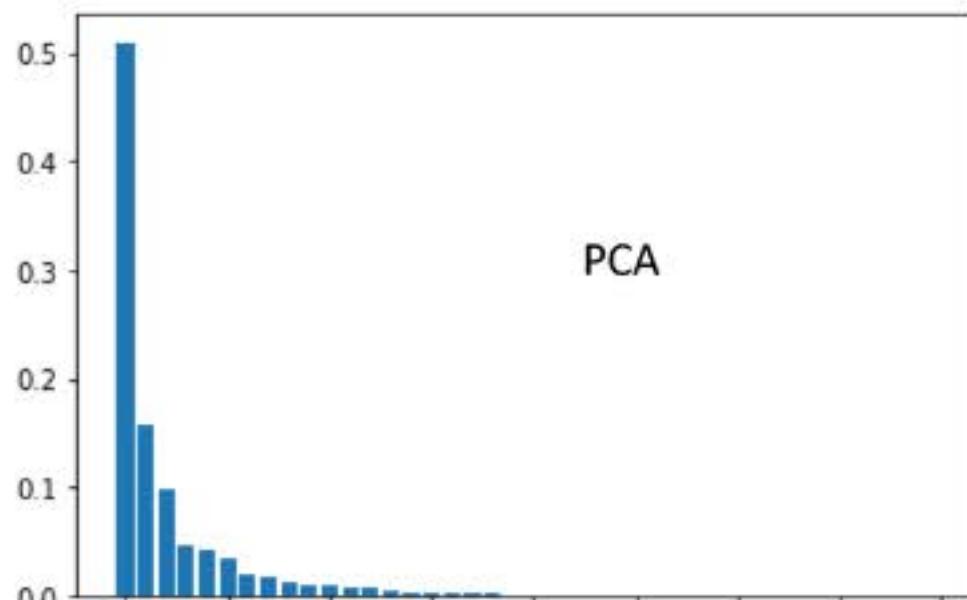
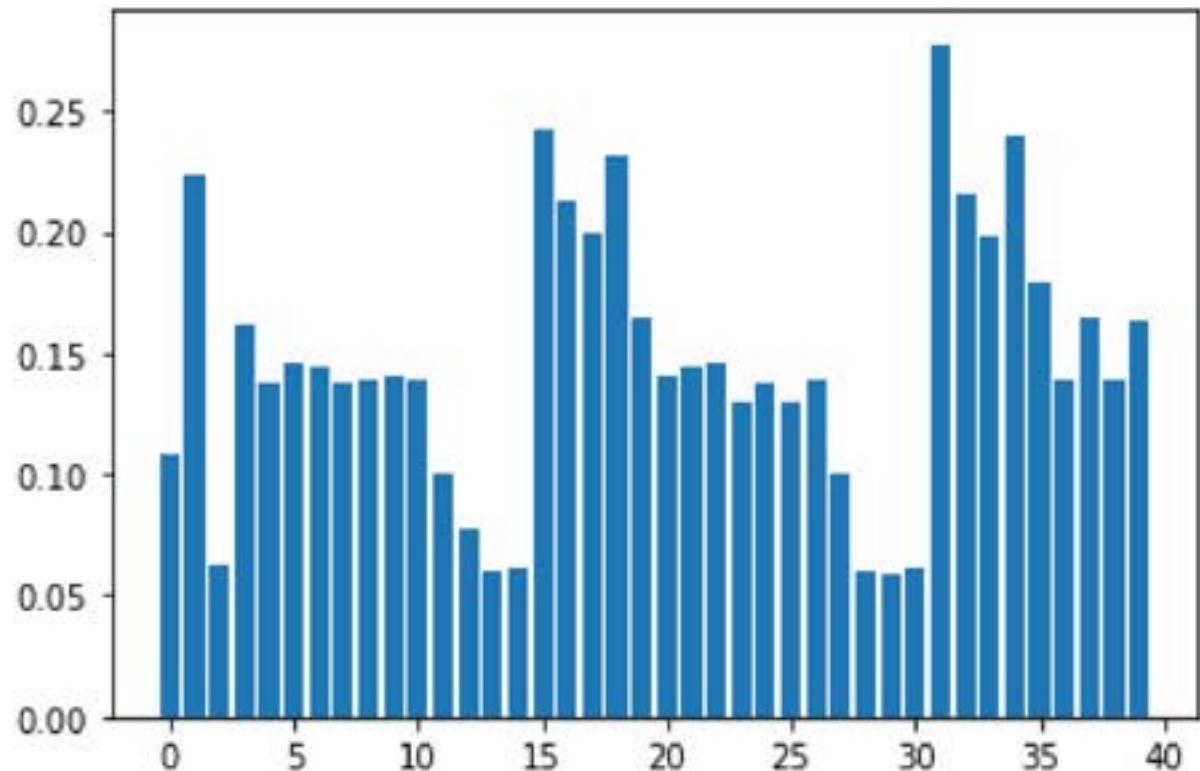
σ_k is the k^{th} singular value of \mathbf{A}
 \mathbf{u}_k & \mathbf{v}_k \rightarrow left / right singular vectors

$$\mathbf{U} \in \mathbb{R}^{m \times m} \quad \mathbf{V} \in \mathbb{R}^{n \times n} \quad \Sigma \in \mathbb{R}^{m \times n}$$

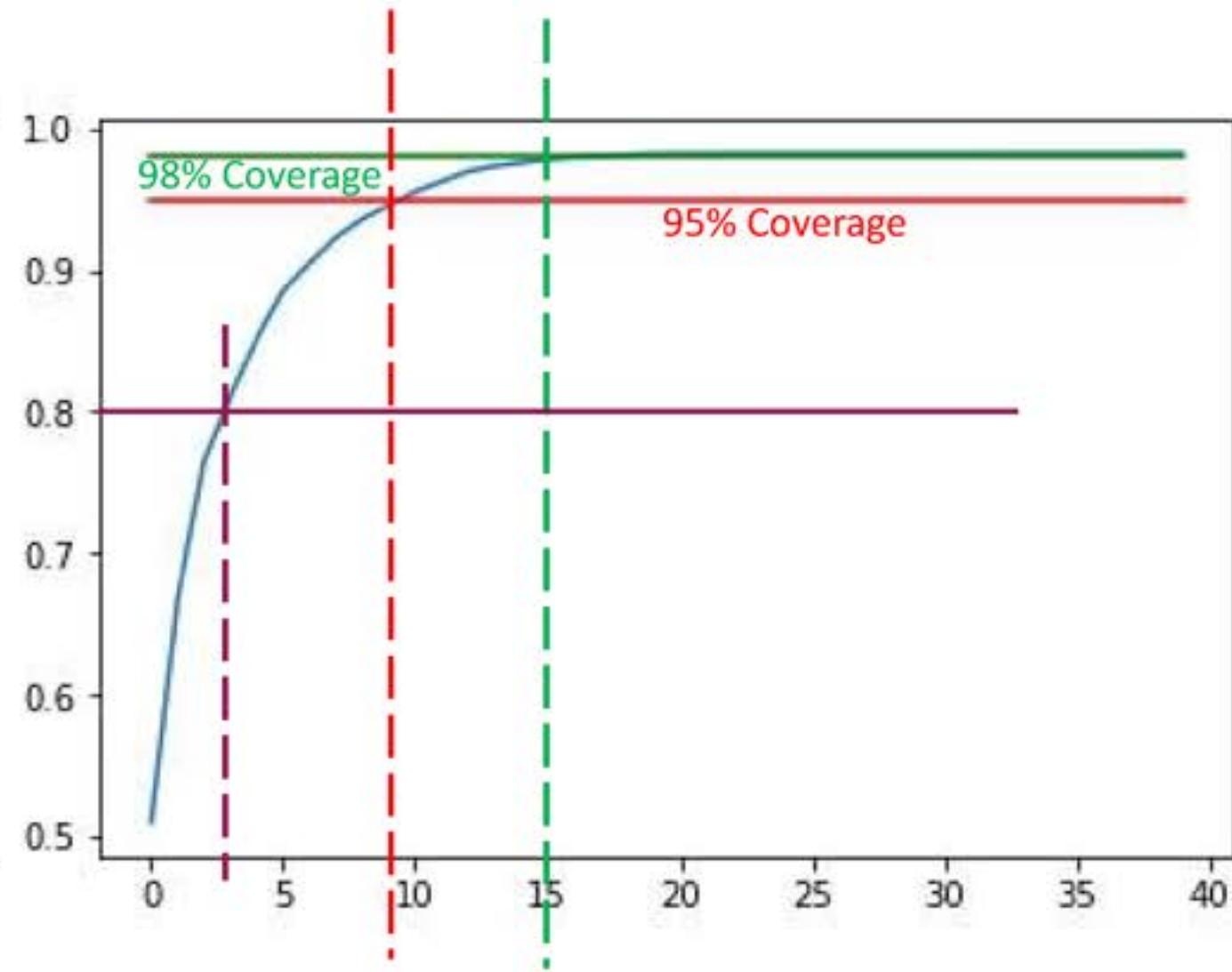
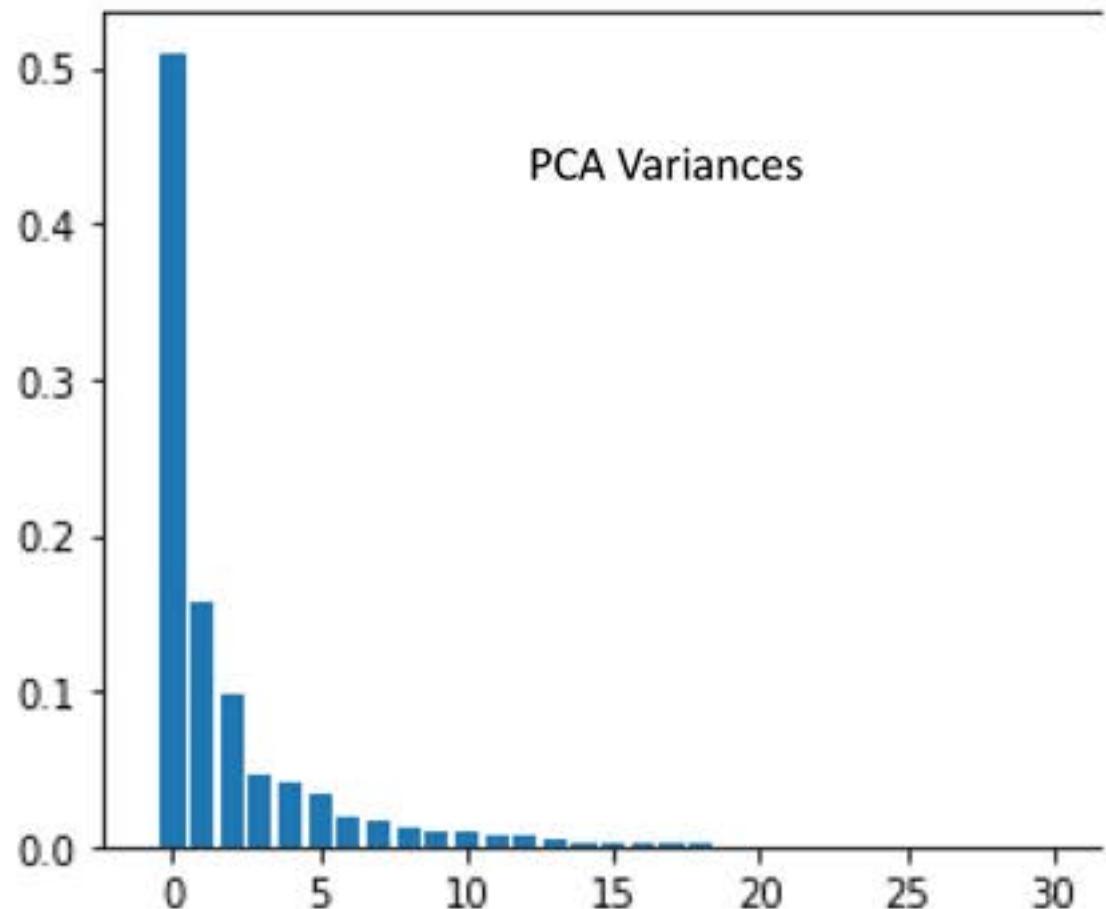
- SVD is performed on matrix without mean removal
- columns of \mathbf{V} \rightarrow Patterns among the attributes
- columns of \mathbf{U} \rightarrow Patterns among the objects

$$\mathbf{D}' = \mathbf{D} * [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k]$$

Variance Capture: PCA / SVD



PCA: Dimensionality Reduction



References

- web.stanford.edu/class/cs276a/handouts/lecture15.pdf
- courses.cs.washington.edu/courses/csep546/16sp/slides/PCA_csep546.pdf
- www.cs.princeton.edu/picasso/mats/Lecture1_jps.ppt

Thank you !!!!!



Machine Learning (19CSE305)

Session 32-33 Clustering



Dr. Peeta Basa Pati
Ms. Priyanka V
Department of Computer Science & Engineering,
Amrita School of Engineering, Bengaluru

Topics

- Clustering.
- k-means clustering

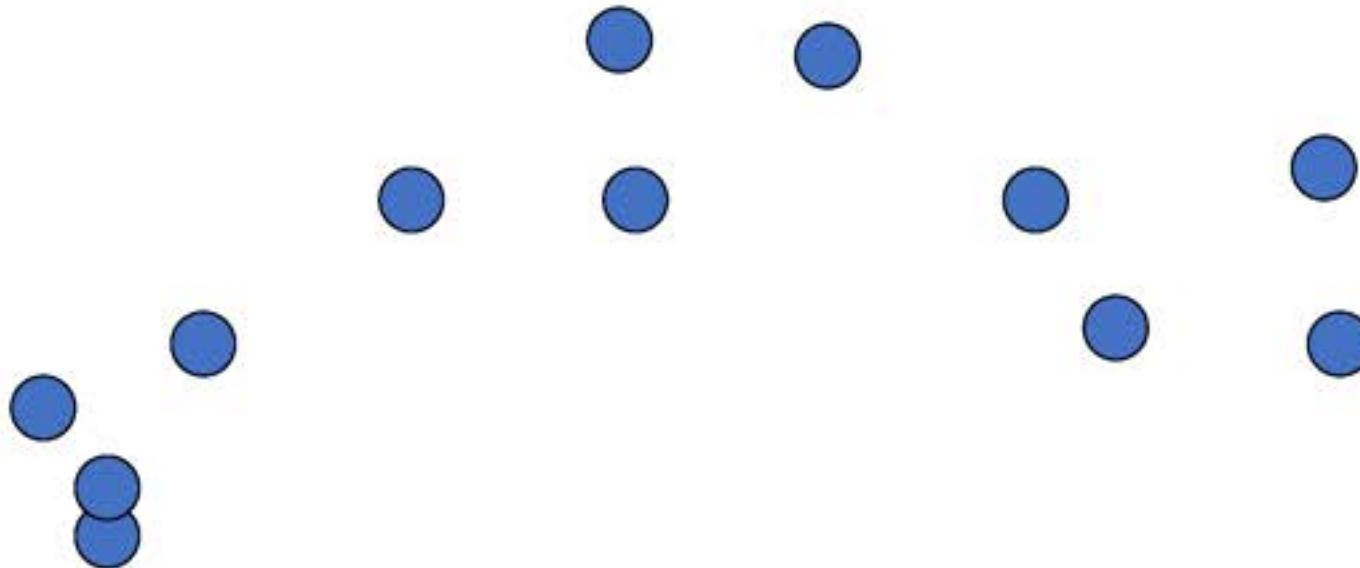
Clustering

- Cluster: A collection of data objects
 - similar (or related) to one another within the same group
 - dissimilar (or unrelated) to the objects in other groups
- Cluster analysis
 - Finding similarities between data according to the characteristics found in the data and grouping similar data objects into clusters
- Unsupervised learning: no predefined classes
- Typical applications
 - As a stand-alone tool to get insight into data distribution
 - As a preprocessing step for other algorithms

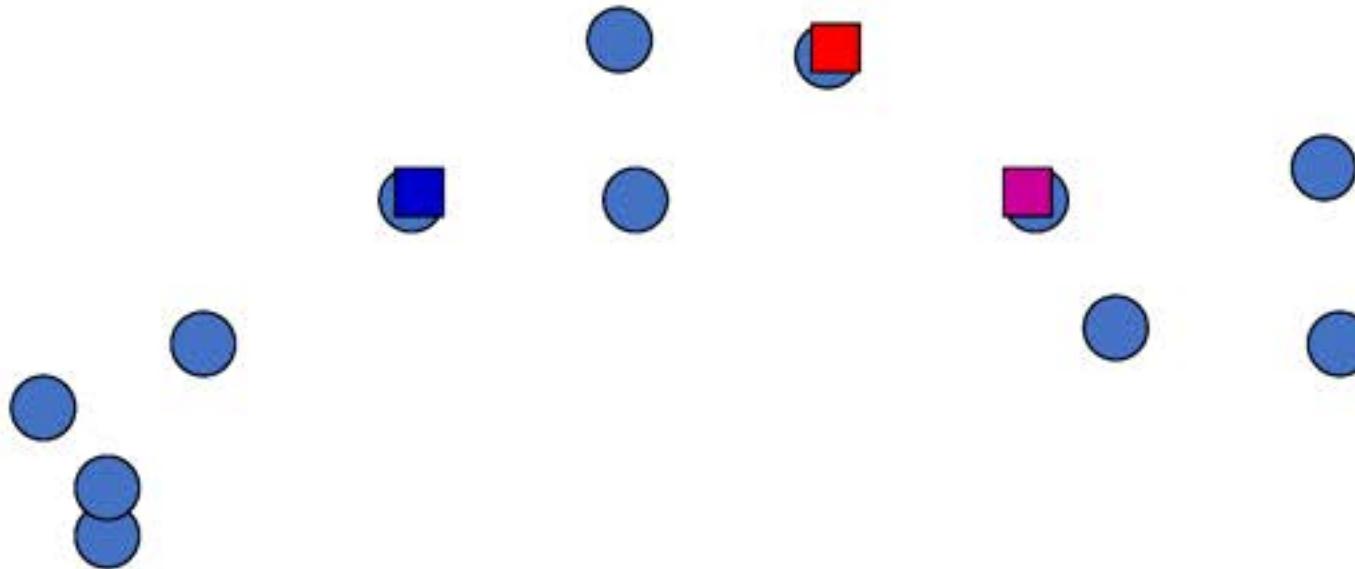
The *K-Means* Clustering Method

- Given k , the *k-means* algorithm is implemented in four steps:
 - Partition objects into k nonempty subsets
 - Compute seed points as the centroids of the clusters of the current partitioning (the centroid is the center, i.e., *mean point*, of the cluster)
 - Assign each object to the cluster with the nearest seed point
 - Go back to Step 2, stop when the assignment does not change

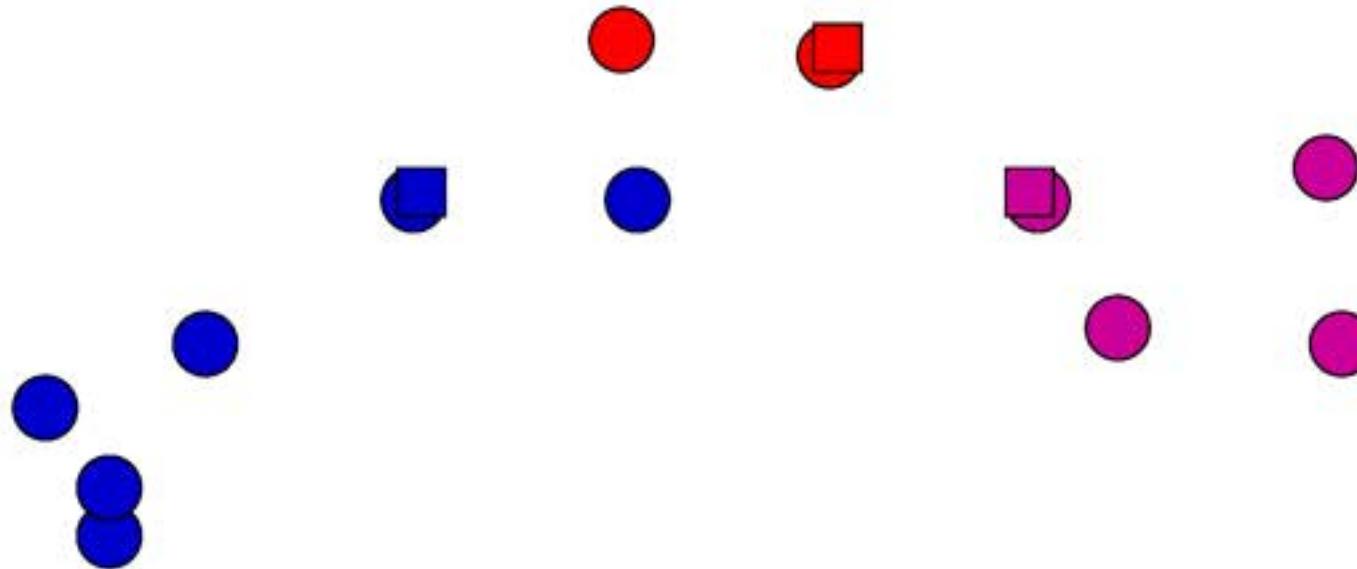
K-means: an example



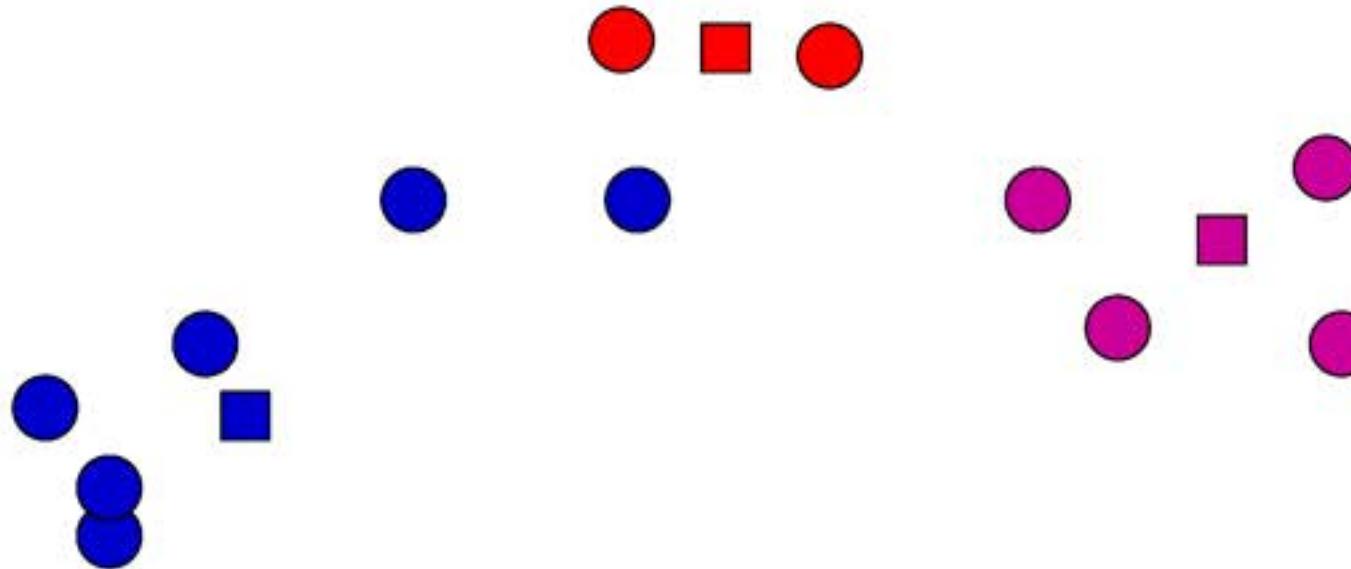
K-means: Initialize centers randomly



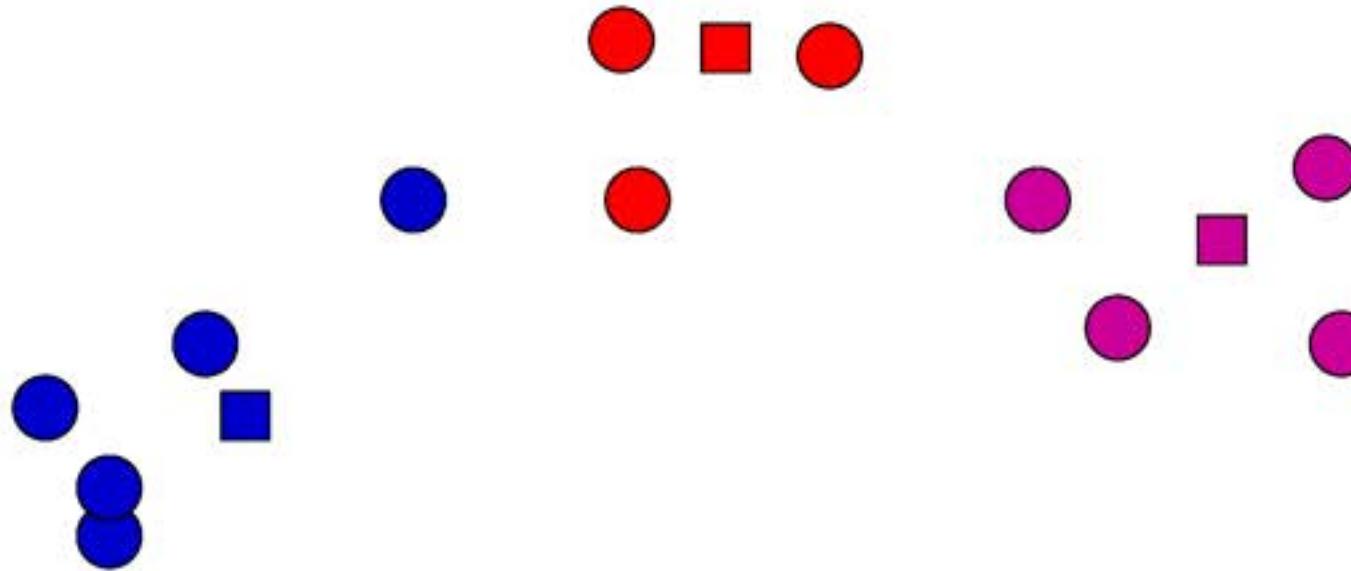
K-means: assign points to nearest center



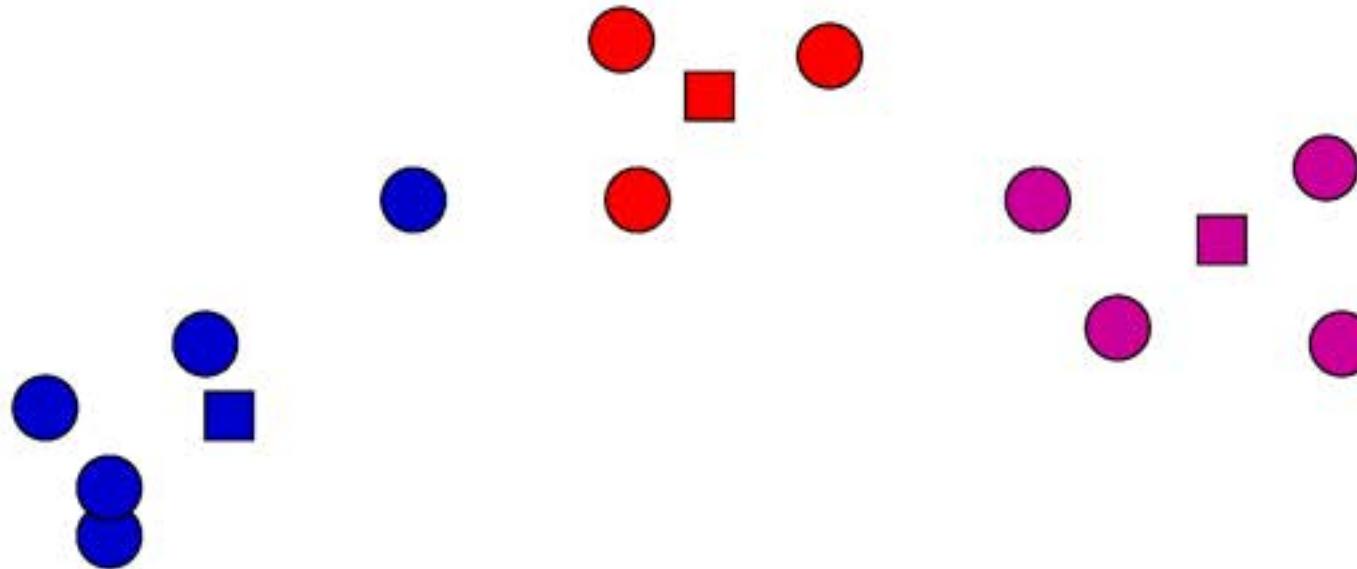
K-means: readjust centers



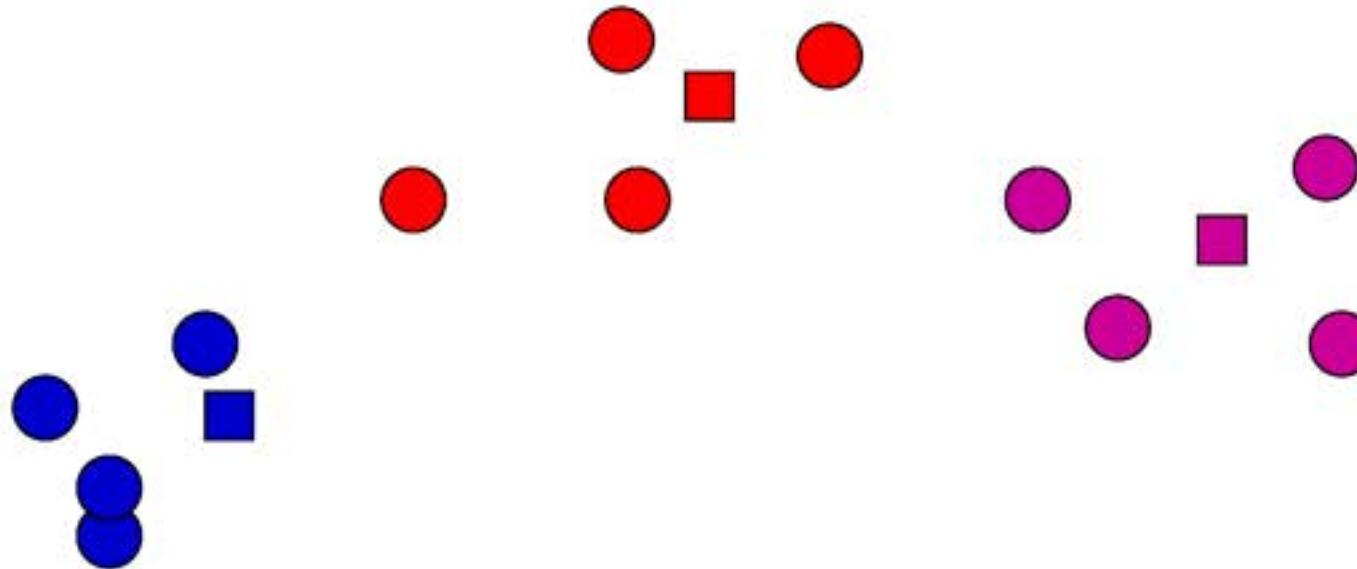
K-means: assign points to nearest center



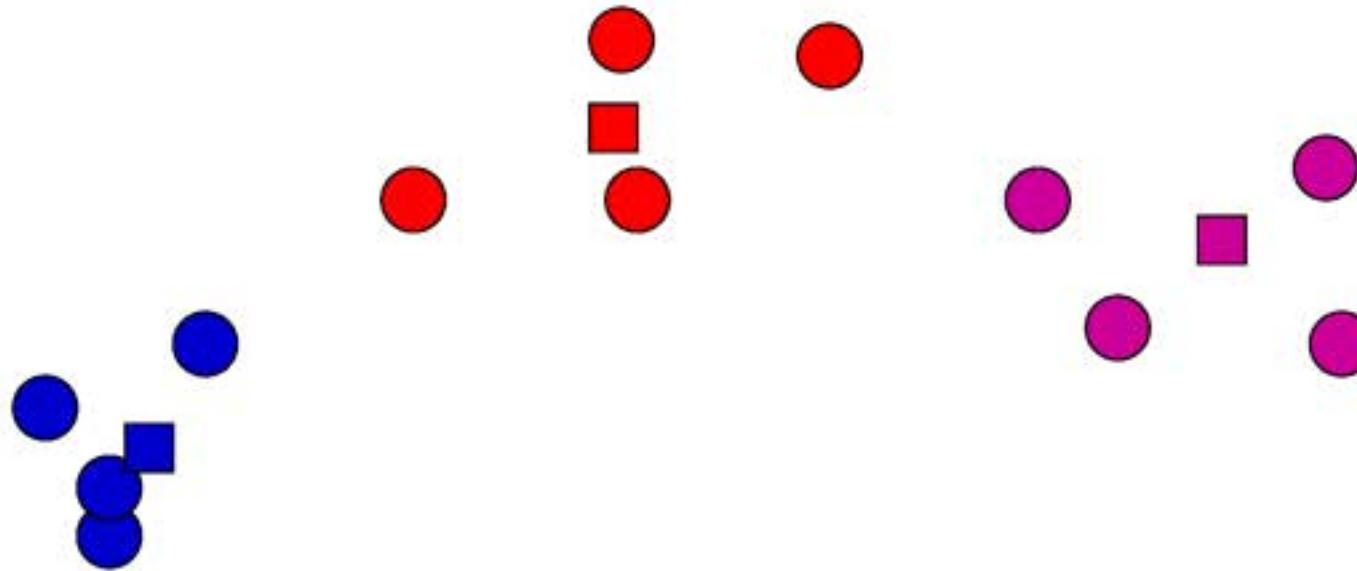
K-means: readjust centers



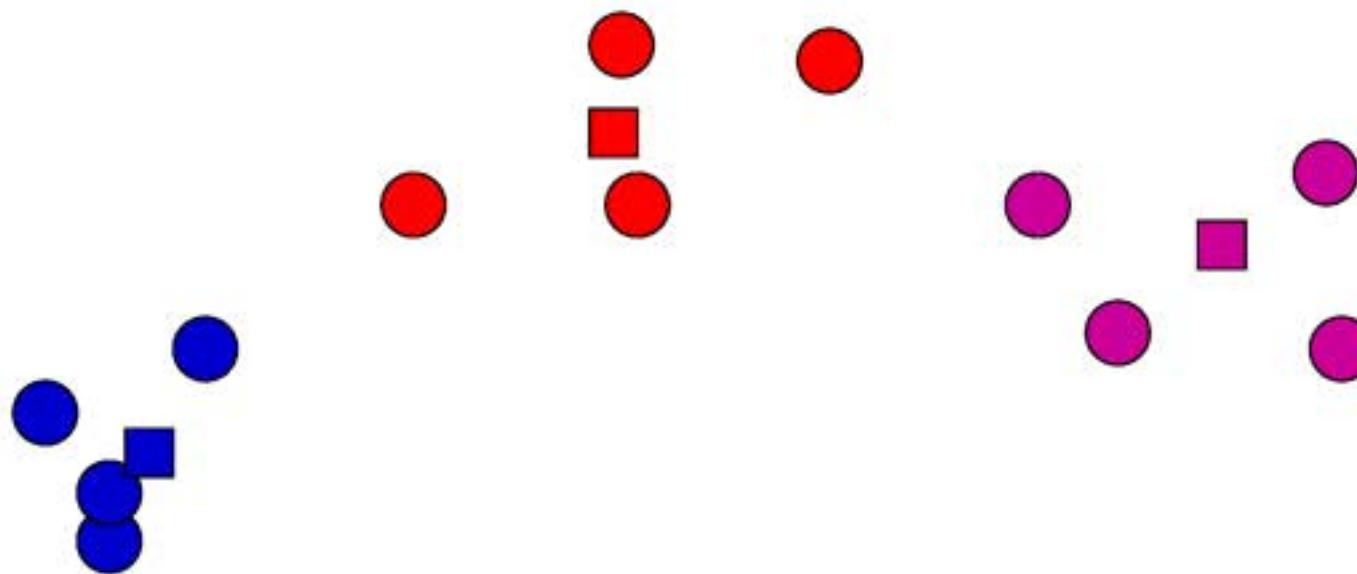
K-means: assign points to nearest center



K-means: readjust centers



K-means: assign points to nearest center

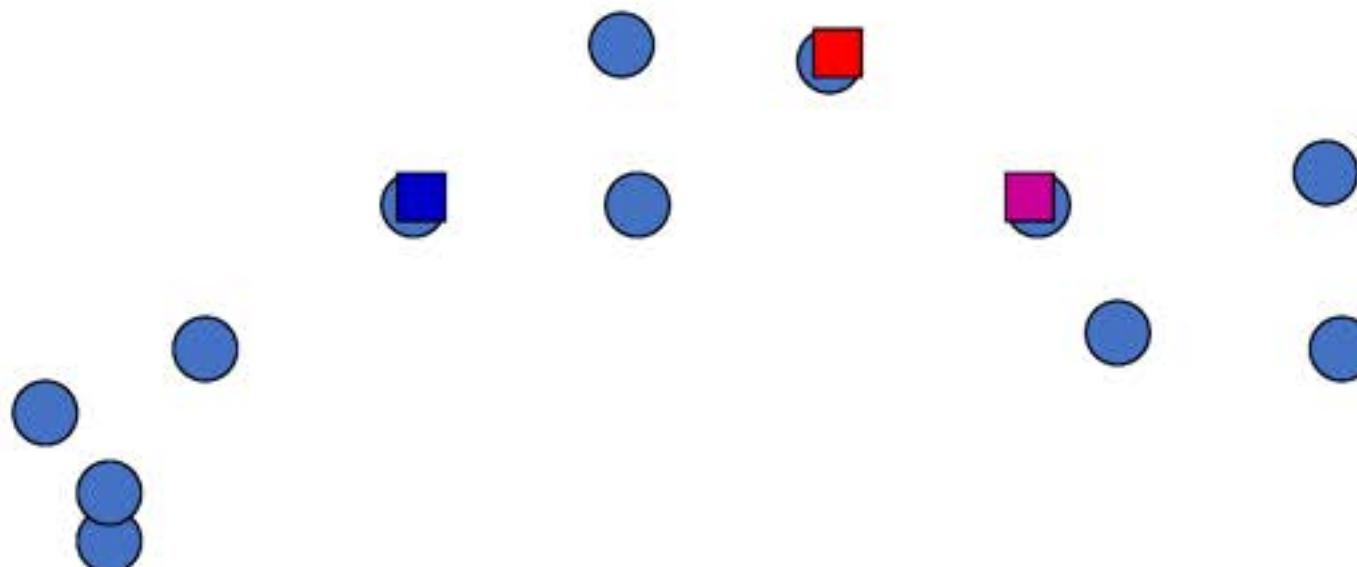


No changes: Done

K-means

Iterate:

- **Assign/cluster each example to closest center**
- Recalculate centers as the mean of the points in a cluster

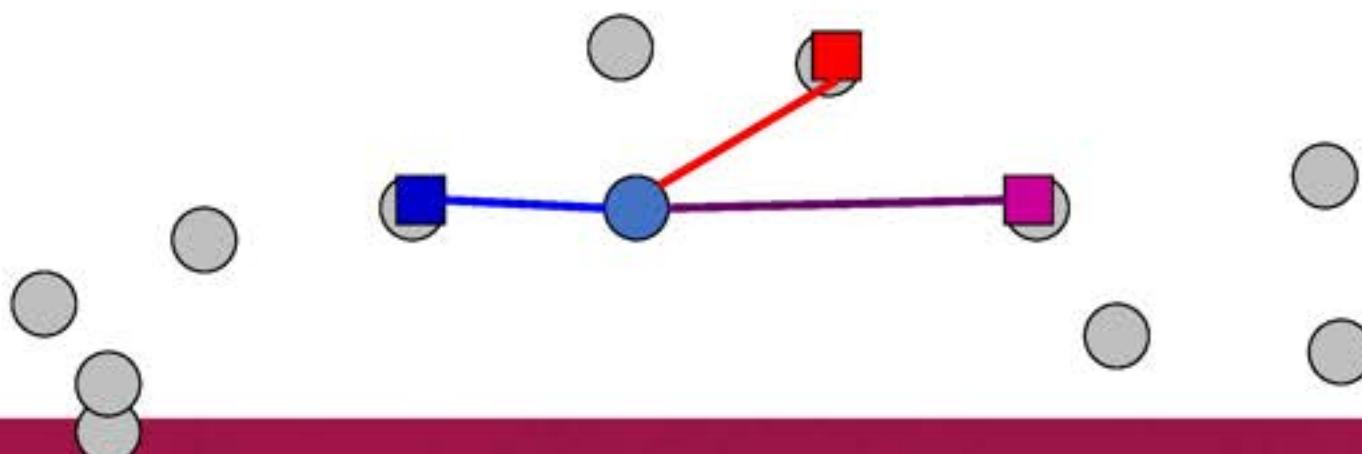


How do we do this?

K-means

Iterate:

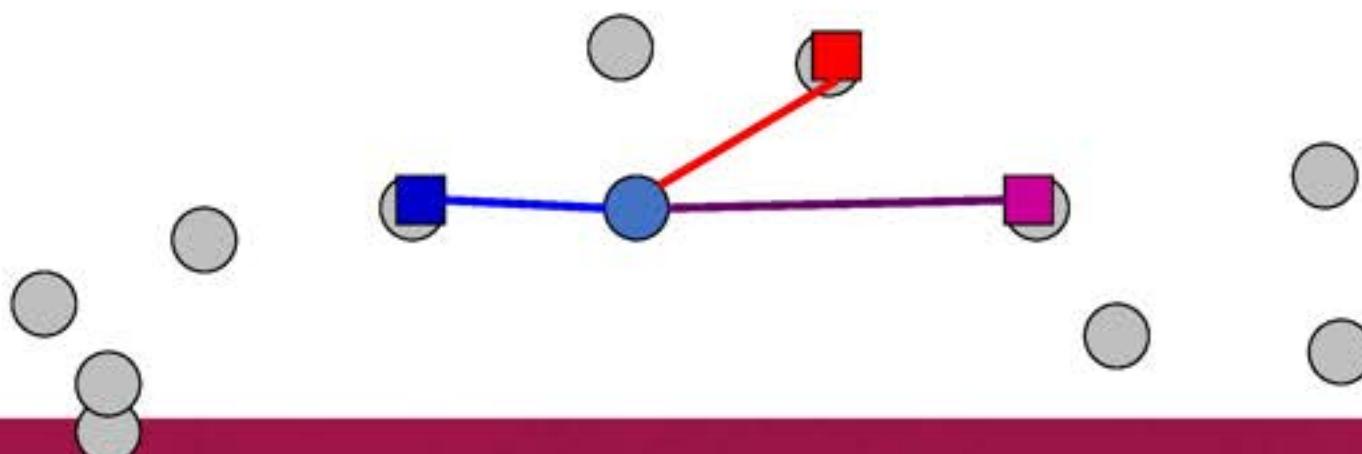
- **Assign/cluster each example to closest center**
iterate over each point:
 - get distance to each cluster center
 - assign to closest center (hard cluster)
- Recalculate centers as the mean of the points in a cluster



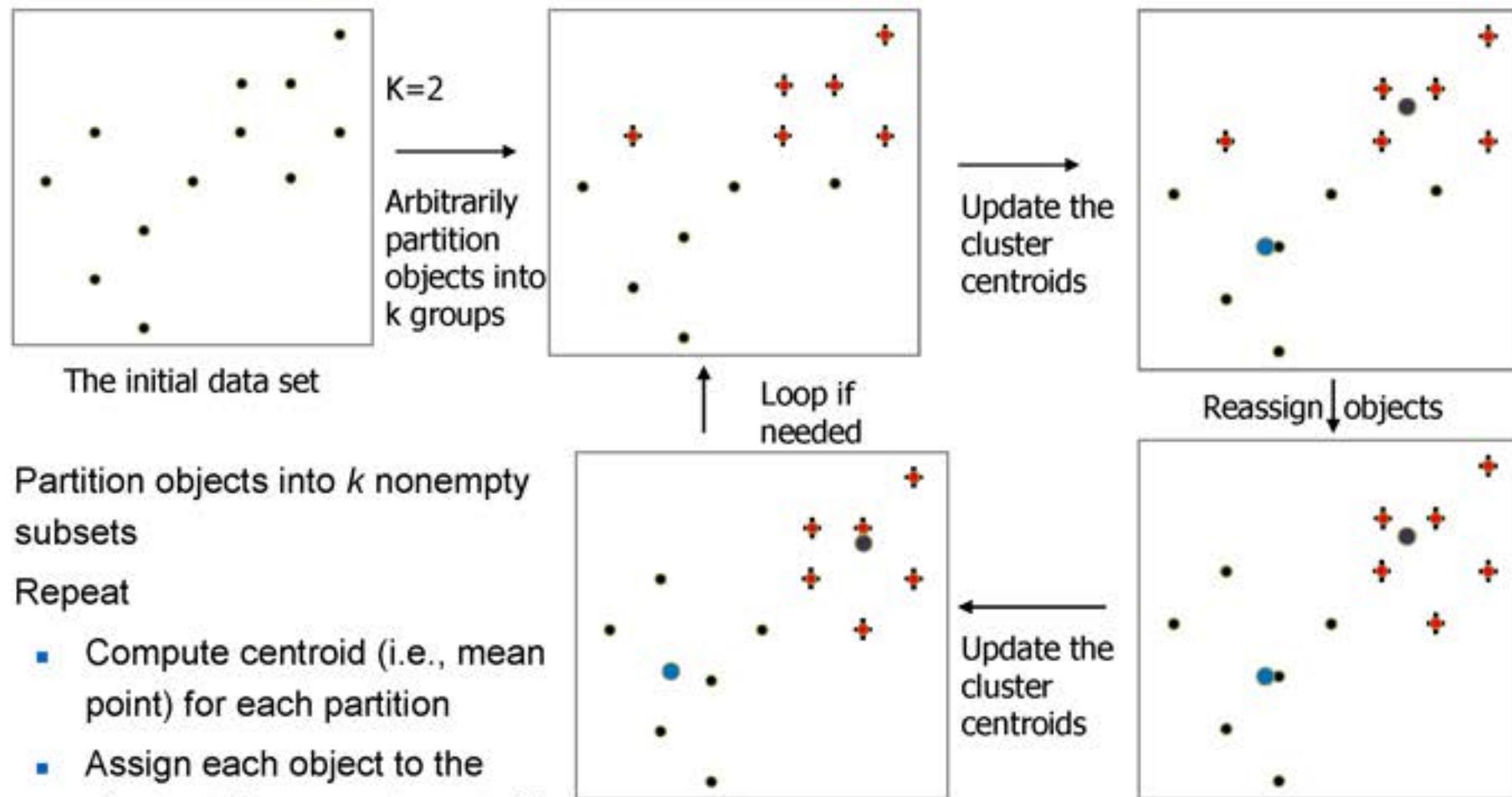
K-means

Iterate:

- **Assign/cluster each example to closest center**
iterate over each point:
 - get **distance** to each cluster center
 - assign to closest center (hard cluster)
- Recalculate centers as the mean of the points in a cluster



An Example of *K*-Means Clustering



A Simple example showing the implementation of k-means algorithm (using K=2)



Individual	Variable 1	Variable 2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5



Step 1:

Initialization: Randomly we choose following two centroids ($k=2$) for two clusters.

In this case the 2 centroid are: $m_1=(1.0, 1.0)$ and $m_2=(5.0, 7.0)$.

Individual	Variable 1	Variable 2
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

Individual	Mean Vector
Group 1	(1.0, 1.0)
Group 2	(5.0, 7.0)

Step 2:

- Thus, we obtain two clusters containing:
{1,2,3} and {4,5,6,7}.
- Their new centroids are:

$$m_1 = \left(\frac{1}{3}(1.0 + 1.5 + 3.0), \frac{1}{3}(1.0 + 2.0 + 4.0) \right) = (1.83, 2.33)$$

$$\begin{aligned} m_2 &= \left(\frac{1}{4}(5.0 + 3.5 + 4.5 + 3.5), \frac{1}{4}(7.0 + 5.0 + 5.0 + 4.5) \right) \\ &= (4.12, 5.38) \end{aligned}$$

Individual	Centroid 1	Centroid 2
1	0	7.21
2 (1.5, 2.0)	1.12	6.10
3	3.61	3.61
4	7.21	0
5	4.72	2.5
6	5.31	2.06
7	4.30	2.92

$$d(m_1, 2) = \sqrt{|1.0 - 1.5|^2 + |1.0 - 2.0|^2} = 1.12$$

$$d(m_2, 2) = \sqrt{|5.0 - 1.5|^2 + |7.0 - 2.0|^2} = 6.10$$



Step 3:

- Now using these centroids we compute the Euclidean distance of each object, as shown in table.
- Therefore, the new clusters are:
 $\{1,2\}$ and $\{3,4,5,6,7\}$
- Next centroids are:
 $m_1 = (1.25, 1.5)$ and $m_2 = (3.9, 5.1)$

Individual	Centroid 1	Centroid 2
1	1.57	5.38
2	0.47	4.28
3	2.04	1.78
4	5.84	1.84
5	3.15	0.73
6	3.78	0.54
7	2.74	1.08



- Step 4 :

The clusters obtained are:
 $\{1,2\}$ and $\{3,4,5,6,7\}$

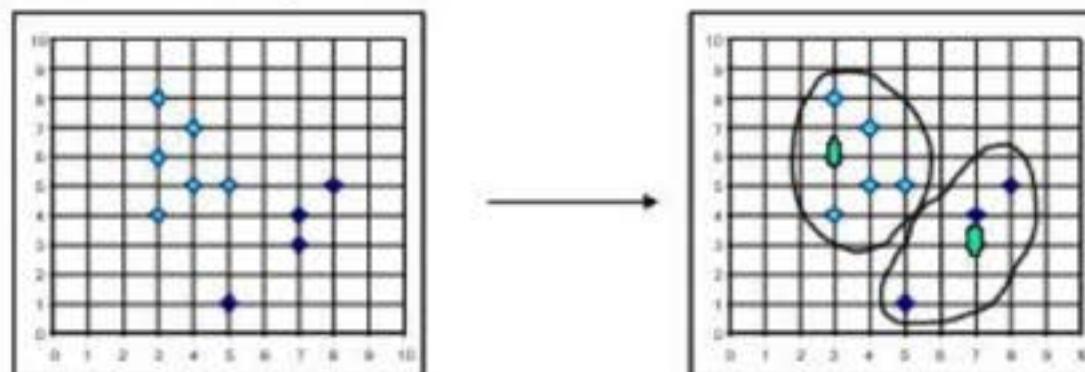
- Therefore, there is no change in the cluster.
- Thus, the algorithm comes to a halt here and final result consist of 2 clusters $\{1,2\}$ and $\{3,4,5,6,7\}$.

Individual	Centroid 1	Centroid 2
1	0.58	5.02
2	0.58	3.92
3	3.05	1.42
4	6.08	2.20
5	4.18	0.41
6	4.78	0.61
7	3.75	0.72

What Is the Problem of the K-Means Method?

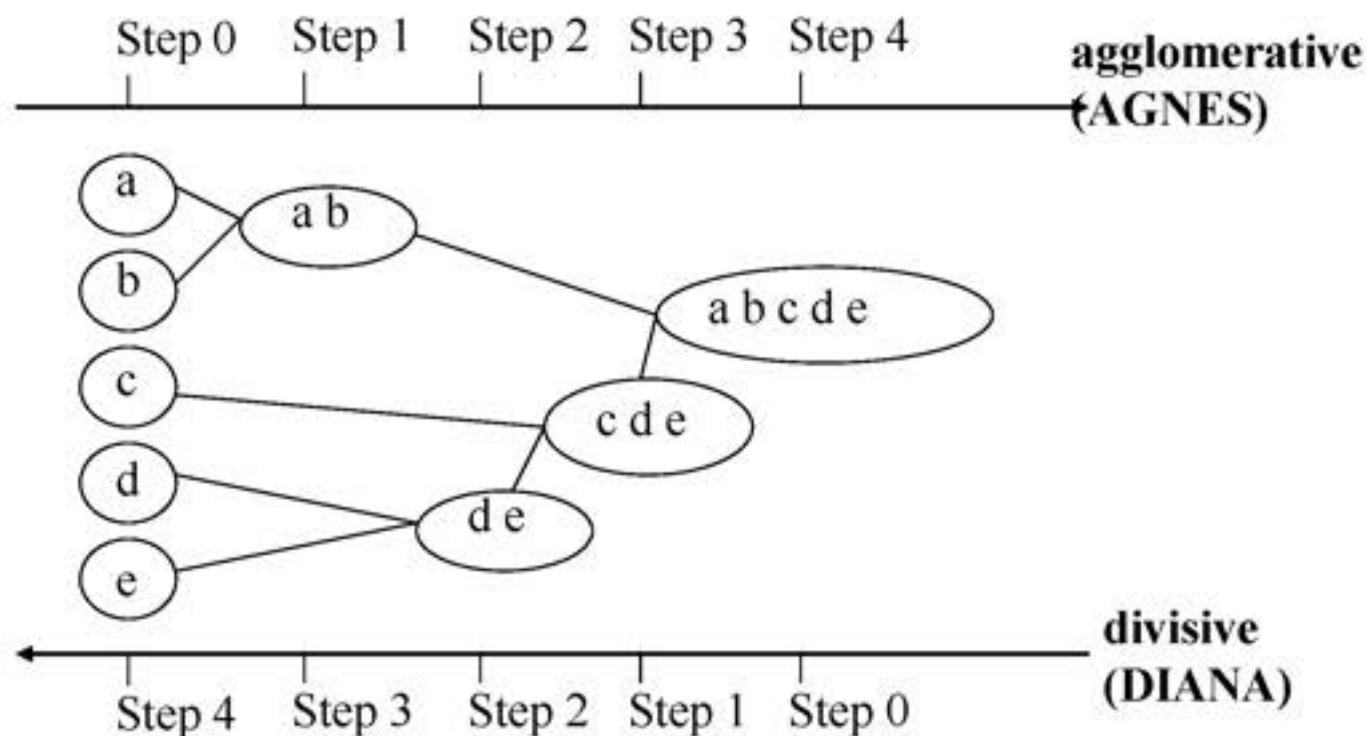
- Since an object with an extremely large value may substantially distort the distribution of the data – sensitive to outliers
- The k-means method is not guaranteed to converge to the global optimum and often terminates at a local optimum.
- The k-means method can be applied only when the mean of a set of objects is defined. This may not be the case in some applications such as when data with nominal attributes are involved.
- The necessity for users to specify k , the number of clusters, in advance

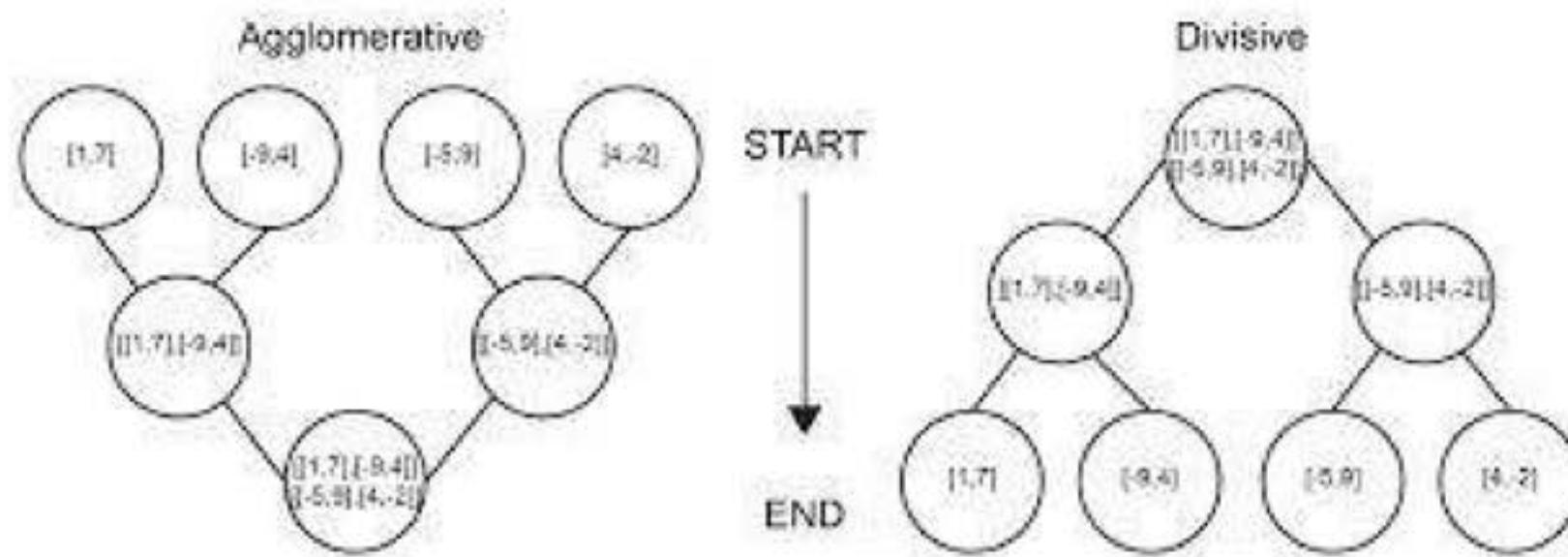
- The k-means algorithm is sensitive to outliers !
 - Since an object with an extremely large value may substantially distort the distribution of the data.
- K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster.



Hierarchical Clustering

- Use distance matrix as clustering criteria. This method does not require the number of clusters k as an input.

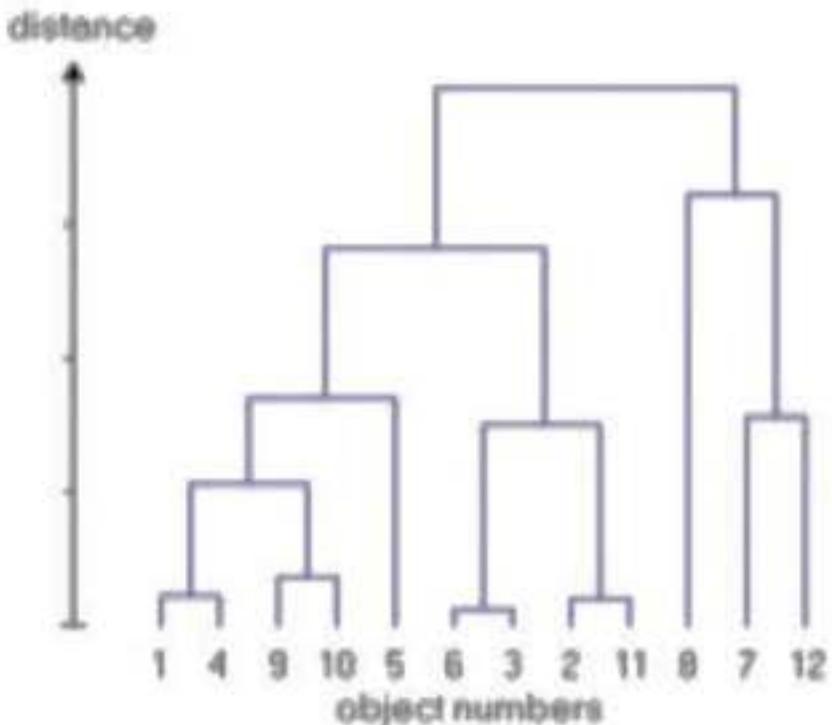




Types of hierarchical clustering

- **Agglomerative (bottom up) clustering:** It builds the dendrogram (tree) from the bottom level, and
 - merges the most similar (or nearest) pair of clusters
 - stops when all the data points are merged into a single cluster (i.e., the root cluster).
- **Divisive (top down) clustering:** It starts with all data points in one cluster, the root.
 - Splits the root into a set of child clusters. Each child cluster is recursively divided further
 - stops when only singleton clusters of individual data points remain, i.e., each cluster with only a single point

Dendrogram: Hierarchical Clustering



Dendrogram

- Each level of the tree represents a partition of the input data into several (nested) clusters or groups.
- May be cut at any level: Each connected component forms a cluster.

Single linkage

	x1	x2
A	1	1
B	1.5	1.5
C	5	5
D	3	4
E	4	4
F	3	3.5

Distance Matrix

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

Closest D, F so merge

Single Linkage

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	?	4.24
B	0.71	0.00	4.95	?	3.54
C	5.66	4.95	0.00	?	1.41
D, F	?	?	?	0.00	?
E	4.24	3.54	1.41	?	0.00

$$d_{(D,F) \rightarrow A} = \min(d_{DA}, d_{FA}) = \min(3.61, 3.20) = 3.20$$

$$d_{(D,F) \rightarrow B} = \min(d_{DB}, d_{FB}) = \min(2.92, 2.50) = 2.50$$

$$d_{(D,F) \rightarrow C} = \min(d_{DC}, d_{FC}) = \min(2.24, 2.50) = 2.24$$

$$d_{E \rightarrow (D,F)} = \min(d_{ED}, d_{EF}) = \min(1.00, 1.12) = 1.00$$

	A	B	C	(D,F)	E
A	0	0.71	5.66	3.2	4.24
B	0.71	0	4.95	2.5	3.54
C	5.66	4.95	0	2.24	1.41
(D,F)	3.2	2.5	2.24	0	1
E	4.24	3.54	1.41	1	0

Min is 0.71 . Merge A & B

Dist	A,B	C	(D, F)	E
A,B	0	?	?	?
C	?	0	2.24	1.41
(D, F)	?	2.24	0	1.00
E	?	1.41	1.00	0

$$d_{C \rightarrow (A,B)} = \min(d_{CA}, d_{CB}) = \min(5.66, 4.95) = 4.95$$

$$d_{(D,F) \rightarrow (A,B)} = \min(d_{DFA}, d_{DFB}) = \min(3.2, 2.5) = 2.5$$

$$d_{E \rightarrow (A,B)} = \min(d_{EA}, d_{EB}) = \min(4.24, 3.54) = 3.54$$

Min Distance (Single Linkage)

Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	4.95	0	2.24	1.41
(D, F)	2.50	2.24	0	1.00
E	3.54	1.41	1.00	0

Min Distance (Single Linkage)

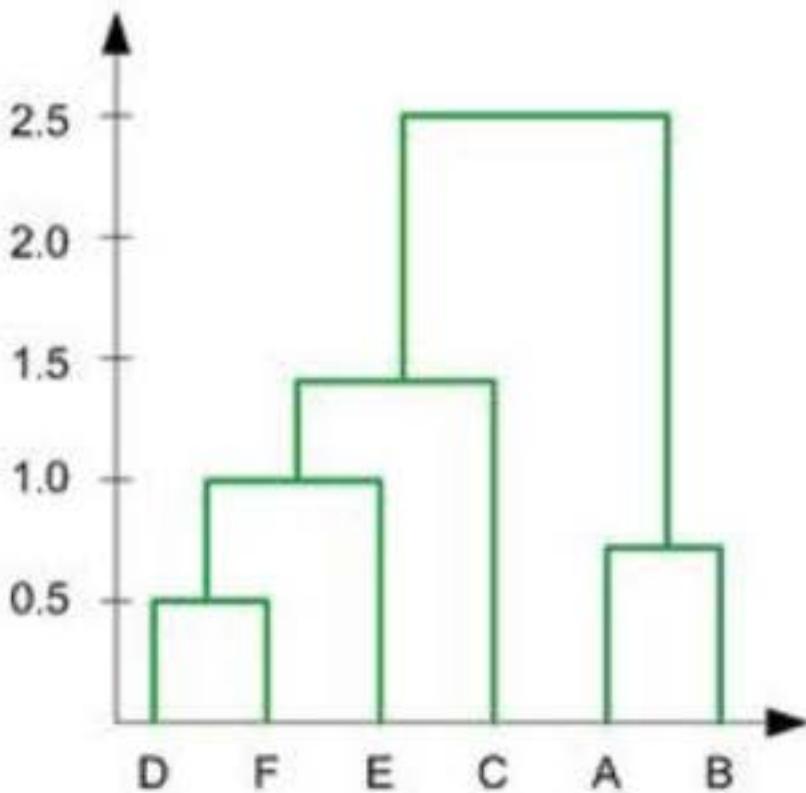
Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	2.50	1.41	0.00

Min Distance (Single Linkage)

Dist	(A,B)	(D, F), E), C
(A,B)	0.00	2.50
((D, F), E), C	2.50	0.00

1. In the beginning we have 6 clusters: A, B, C, D, E and F
2. We merge cluster D and F into cluster (D, F) at distance 0.50
3. We merge cluster A and cluster B into (A, B) at distance 0.71
4. We merge cluster E and (D, F) into ((D, F), E) at distance 1.00
5. We merge cluster ((D, F), E) and C into (((D, F), E), C) at distance 1.41
6. We merge cluster (((D, F), E), C) and (A, B) into ((((D, F), E), C), (A, B)) at distance 2.50
7. The last cluster contain all the objects, thus conclude the computation

Dendrogram



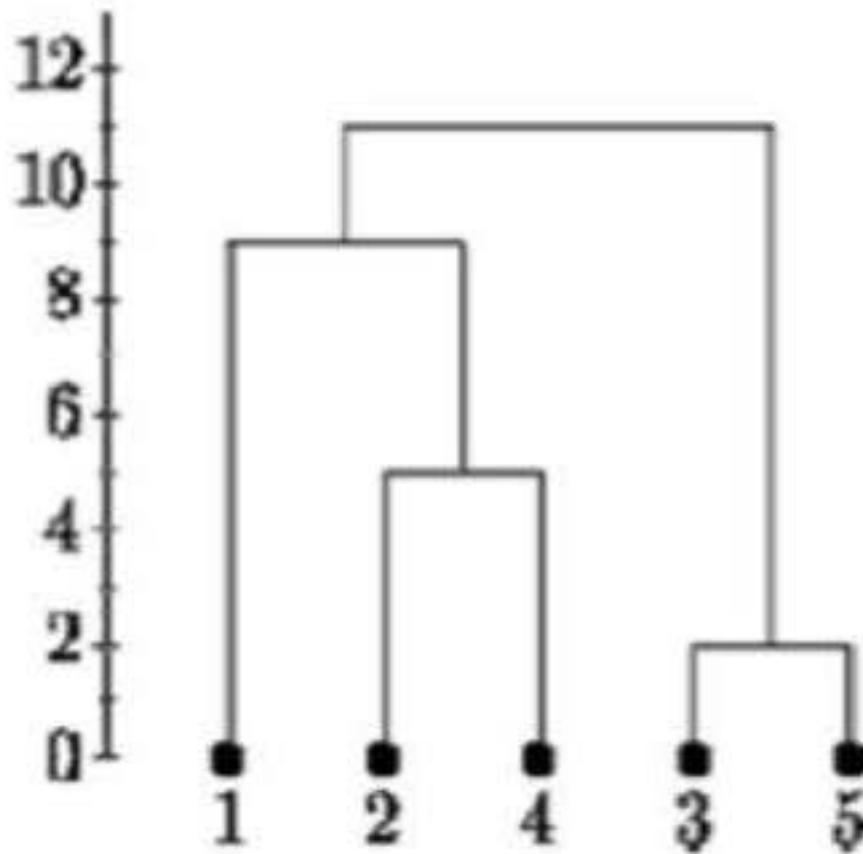
Complete Linkage

	1	2	3	4	5
1	0				
2	9	0			
3	3	7	0		
4	6	5	9	0	
5	11	10	2	8	0

Complete Linkage

- Since we are using complete linkage clustering, the distance between "35" and every other item is the maximum of the distance between this item and 3 and this item and 5. For example, $d(1,3)=3$ and $d(1,5)=11$. So, $D(1,"35")=11$.

	3	5	1	2	4	
3	5	0				
1		1	1	0		
2		1	0	9	0	
4		9	6	5	0	



Complete Linkage

Linkage criteria



- Single link: smallest distance between an element in one cluster and an element in the other, i.e., $\text{dist}(K_i, K_j) = \min(t_{ip}, t_{jq})$
- Complete link: largest distance between an element in one cluster and an element in the other, i.e., $\text{dist}(K_i, K_j) = \max(t_{ip}, t_{jq})$
- Average: avg distance between an element in one cluster and an element in the other, i.e., $\text{dist}(K_i, K_j) = \text{avg}(t_{ip}, t_{jq})$

Determine the Number of Clusters

- Empirical method
 - # of clusters $\approx \sqrt{n}/2$ for a dataset of n points
- Elbow method
 - Use the turning point in the curve of sum of within cluster variance w.r.t the # of clusters
- Cross validation method
 - Divide a given data set into m parts
 - Use $m - 1$ parts to obtain a clustering model
 - Use the remaining part to test the quality of the clustering
 - E.g., For each point in the test set, find the closest centroid, and use the sum of squared distance between all points in the test set and the closest centroids to measure how well the model fits the test set
 - For any $k > 0$, repeat it m times, compare the overall quality measure w.r.t. different k 's, and find # of clusters that fits the data the best

Measuring Clustering Quality

- External: supervised, employ criteria not inherent to the dataset
 - Compare a clustering against prior or expert-specified knowledge using certain clustering quality measure
- Internal: unsupervised, criteria derived from data itself
 - Evaluate the goodness of a clustering by considering how well the clusters are separated, and how compact the clusters are, e.g., Silhouette coefficient
- Relative: directly compare different clusterings, usually those obtained via different parameter settings for the same algorithm

- K-means and K-medoids algorithms are popular partitioning-based clustering algorithms
- Birch and Chameleon are interesting hierarchical clustering algorithms, and there are also probabilistic hierarchical clustering algorithms
- DBSCAN, OPTICS, and DENCLU are interesting density-based algorithms
- STING and CLIQUE are grid-based methods, where CLIQUE is also a subspace clustering algorithm

Thank you !!!!!

