

# **Bachelor of Computer Science (Hons) Year-2 Sep 2023**

# Welcome to Intelligent Systems

CAI3204N

# Learning Objectives

- ❑ At the end of the course, students will be able to:
  - ❑ CO1: Identify the types of problem that are amenable to "intelligent" solutions.
  - ❑ CO2: Compare and contrast the various intelligent system techniques to solve such problems.
  - ❑ CO3: Select and apply appropriate intelligent techniques to a given problem.
  - ❑ CO4: Critically discuss intelligent system research issues and their applications.

# Artificial Neural Networks

Perceptron

# Learning Objectives

- ❑ At the end of the course, students will be able to:
  - ❑ CO1: Identify the types of problem that are amenable to "intelligent" solutions.
  - ❑ CO2: Compare and contrast the various intelligent system techniques to solve such problems.
  - ❑ CO3: Select and apply appropriate intelligent techniques to a given problem.
  - ❑ CO4: Critically discuss intelligent system research issues and their applications.

# Revisit

Perceptron

# Today's Overview

- The Perceptron
- Learning
- Limits of Learning

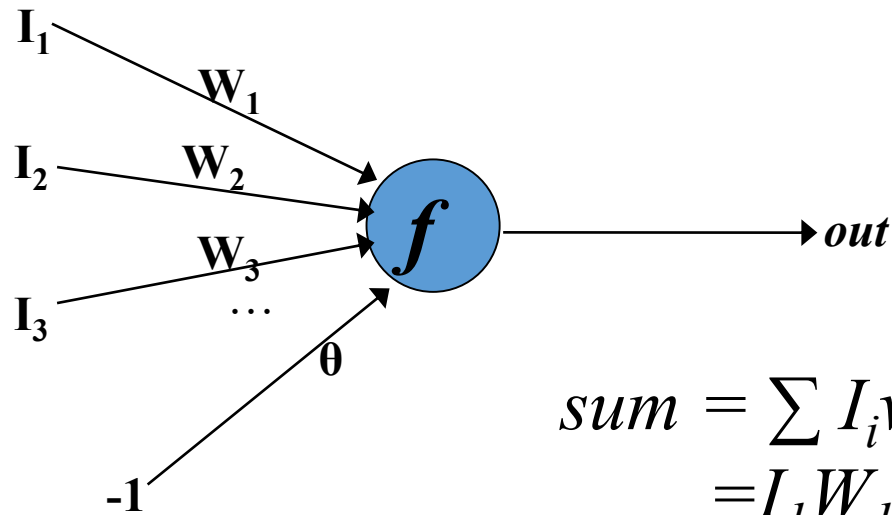
# The Perceptron

- The simplest form of neural network
- Developed by Rosenblatt, 1957
- Used as a *decision system*
  - *ie* a **two-class classifier**
- Thresholded step activation function
- Bipolar or binary output
- <http://en.wikipedia.org/wiki/Perceptron>



# Perceptron Thresholds

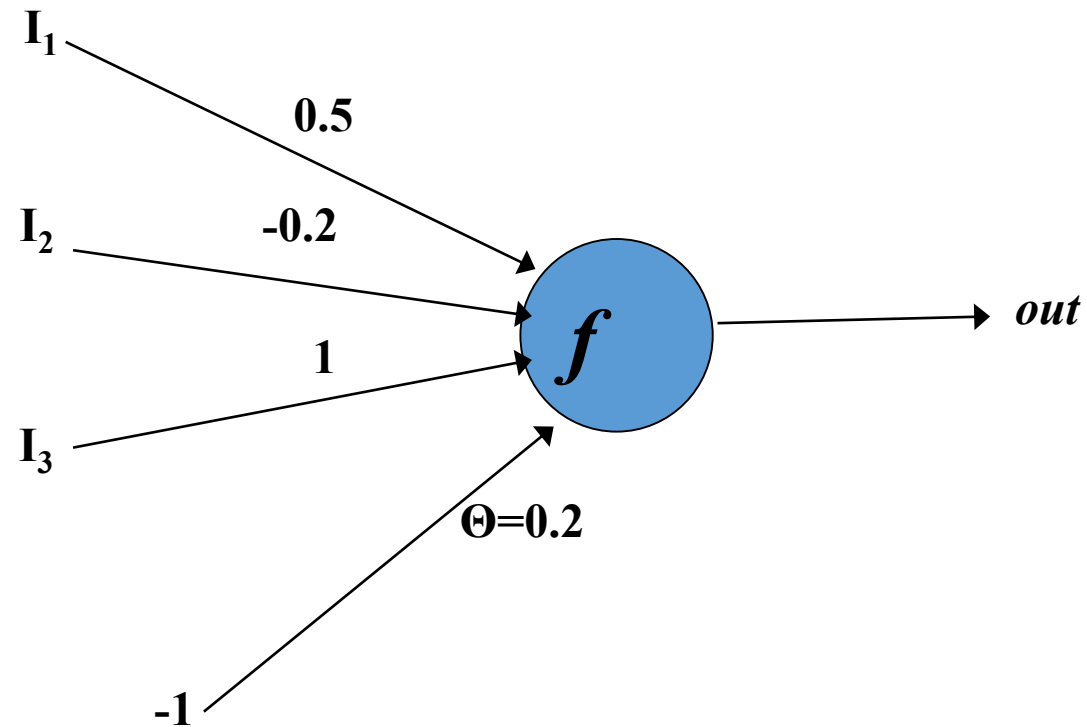
- Threshold can be represented as a weighted input



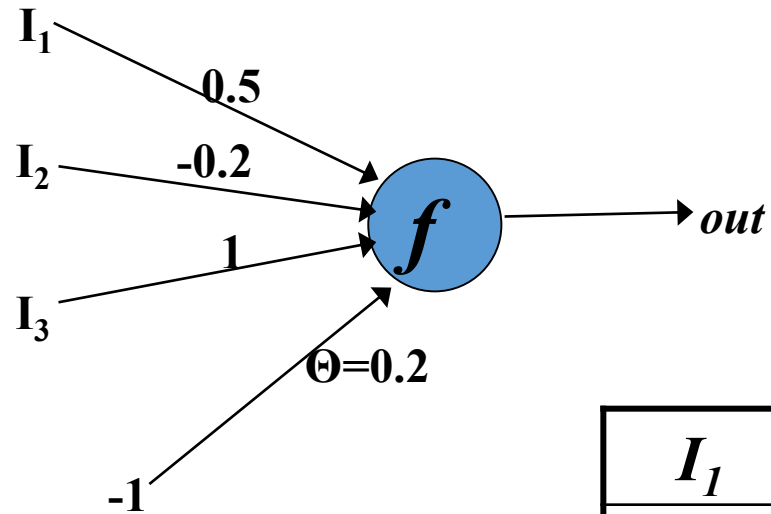
$$\begin{aligned} sum &= \sum I_i w_i \\ &= I_1 W_1 + I_2 W_2 + I_3 W_3 + \dots \end{aligned}$$

$out = 1$  if  $sum - \theta > 0$ ,  $0$  (or  $-1$ ) otherwise

# Perceptron Example



# Perceptron Example



$I_1$	$I_2$	$I_3$	$sum$	$out$
0.0	1.0	0.5	0.1	+1
-0.2	0.0	-1.0	-1.3	-1
0.2	-0.5	0.2	0.2	+1
-0.8	0.2	-0.6	-1.44	-1

# Perceptron Learning

- How do we get a perceptron to learn to classify examples correctly?
- Present each of the examples in the training set
- See what output you get
- Adjust the weights to get the output 'more right'
- Until it does what you want

# Training a Perceptron

1. Start weights **at random**
2. Present **inputs and calculate** outputs
3. Find error compared with desired output
4. Adjust weights
5. Repeat 2-4 until:
  - *Either* got the outputs you want
  - *Or* results not getting any better
6. Then use network to make predictions

# Learning Rule

- How to adjust the weights?
  - If  $\text{input} > 0$ :
    - If the answer is too big, reduce the weight
    - If too small, increase it
  - *but if input < 0*:
    - If the answer is too big, *increase* the weight
    - If too small, *reduce* it
  - Only **increase/decrease by a little bit at a time**

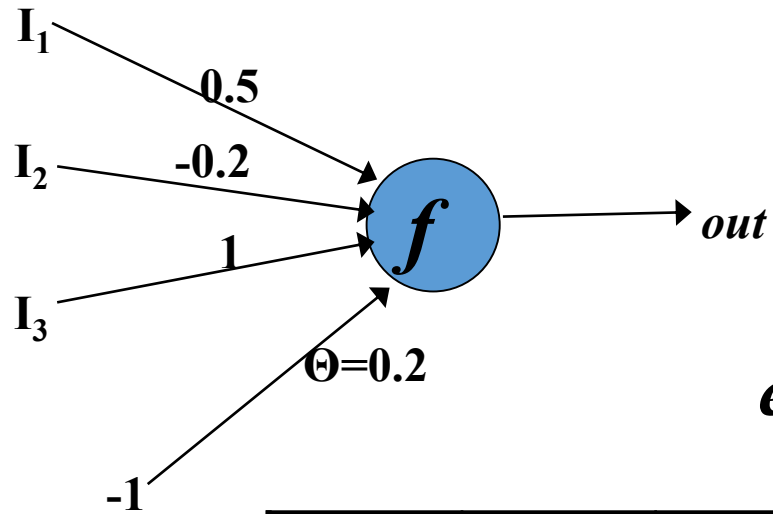
# Learning Rule

*error = output – target*

*rate = 0.2* (for example)

$$W_{new} = W_{old} - (error \times input \times rate)$$

# Learning Example - Error

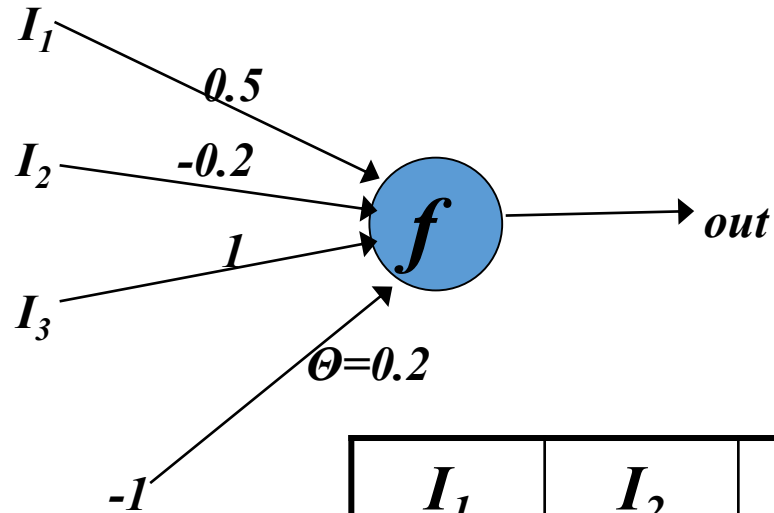


$$error = output - target$$

$I_1$	$I_2$	$I_3$	$sum$	$out$	<i>target</i>	<i>error</i>
0.0	1.0	0.5	0.1	+1	+1	0
-0.2	0.0	-1.0	-1.3	-1	+1	-2
0.2	-0.5	0.2	0.2	+1	-1	+2
-0.8	0.2	-0.6	-1.44	-1	-1	0



# Learning Example



$I_1$	$I_2$	$I_3$	<i>sum</i>	<i>out</i>	<i>target</i>	<i>error</i>
0.2	-0.5	0.2	0.2	+1	-1	+2

$W_1=0.5$ ,  $I_1=0.2$ , so decrease  $W_1$

$W_2=-0.2$ ,  $I_2=-0.5$ , so decrease  $W_2$

$W_3=1.0$ ,  $I_3=0.2$ , so decrease  $W_3$

$\vartheta=0.2$ ,  $I_\theta=-1$ , so increase  $\theta$

# Training Regimes

- Suppose there are three examples in training set
  - {A,B,C}
  - Present each example repeatedly until successful on each
  - *Or* Present complete set in turn until successful on all?
- Which training regime would be best?
- The latter, since by the time we have trained on B and C, may have 'forgotten' how to do A

A	x	A	x
A	x	B	x
A	x	C	✓
A	✓	A	✓
B	x	B	x
B	x	C	✓
B	✓	A	x
C	x	B	x
C	x	C	✓
C	x	...	
C	x	A	✓
C	✓	B	✓
		C	✓

OR



## Second Half

- Human Learning
- Machine Learning
  - Supervised Learning
  - Examples
- Data Mining
  - Examples

# Why Machine Learning?



- How do we get our machine to produce intelligent outputs in response to inputs?
- Programming it ourselves is very hard
- How do Humans get to produce them?
  - *They Learn*

# So how do humans learn?

# Learning 'By Rote'

- Sometimes told the right outputs for each possible input
  - *E.g.* learning times table / actors learning lines
  - Requires memory, not intelligence
  - = putting data in a database
  - Or a 'look-up' table
  - *Eg* The Chinese Room
- But no capacity to *generalise*
  - can only answer a question if we have been given the exact question and answer before
  - Cannot answer new questions, no matter how similar they are to previously seen ones

# Rule-Following

- Sometimes we are told 'the rules'
  - *e.g.* learning how to do long division
  - Just remembering the rules and how to apply them
  - Is it learning? Is it intelligence?
  - = writing a computer program to solve a problem explicitly. The computer can now do something it couldn't before; but is this learning?
- Problem: not always possible to work out what the rules should be
  - May not even be any clear definite rules at all



# Supervised Learning

- Sometimes shown a few examples and then required to apply the lessons to new cases
  - *eg* learning verb endings:
    - Jump, jumps, *jumping*, jumped
  - Need to *generalise* from past experience
  - AKA **Supervised Learning**
- Requires some regularity in the examples
  - *ie* similar inputs produce similar outputs
  - Cannot deal with completely exceptional cases
  - The problem is to spot the regularity
- The most common form of Machine Learning

# Reinforcement Learning

- Sometimes learn from **painful** (or happy) experience
  - *eg* Learning motor skills
  - Trial and Error
  - **AKA Reinforcement Learning**
- Because we don't get 'rich' feedback (ie never told explicitly what the answer should be, then can **take a lot of trials and a lot of error**)

# Unsupervised

- Sometimes not sure what we are supposed to learn
  - Eg baby learns to see objects in unorganised visual sense data
  - Identifying regularities in input
  - May then use that skill in other tasks
  - **AKA Unsupervised Learning**

# Supervised Machine Learning

- The most common form of machine learning
- System is given a set of example inputs (questions) with correct outputs (answers)
- Then makes predictions about other instances

Input	Output
<i>Input 1</i>	<i>Output 1</i>
<i>Input 2</i>	<i>Output 2</i>
<i>Input 3</i>	<i>Output 3</i>
<i>Input 4</i>	<i>Output 4</i>
<i>Input 5</i>	<i>Output 5</i>
<i>Input 6</i>	<i>Output 6</i>
<i>Input 7</i>	<b>?</b>

# Supervised Machine Learning

- May know more than one thing about each of the examples
- Which may, or may not, be helpful

**Instances or  
Samples**

Input			Out
$In_{11}$	$In_{12}$	$In_{13}$	$Out_1$
$In_{21}$	$In_{22}$	$In_{23}$	$Out_2$
$In_{31}$	$In_{32}$	$In_{33}$	$Out_3$
$In_{41}$	$In_{42}$	$In_{43}$	$Out_4$
$In_{51}$	$In_{52}$	$In_{53}$	$Out_5$
$In_{61}$	$In_{62}$	$In_{63}$	$Out_6$
$In_{71}$	$In_{72}$	$In_{73}$	?

**Attributes or  
Features or Variables**

# Supervised Machine Learning

Input			Out	
$In_{11}$	$In_{12}$	$In_{13}$	$Out_{11}$	$Out_{12}$
$In_{21}$	$In_{22}$	$In_{23}$	$Out_{21}$	$Out_{22}$
$In_{31}$	$In_{32}$	$In_{33}$	$Out_{31}$	$Out_{32}$
$In_{41}$	$In_{42}$	$In_{43}$	$Out_{41}$	$Out_{42}$
$In_{51}$	$In_{52}$	$In_{53}$	$Out_{51}$	$Out_{52}$
$In_{61}$	$In_{62}$	$In_{63}$	$Out_{61}$	$Out_{62}$
$In_{71}$	$In_{72}$	$In_{73}$	?	?

=

Input			Out
$In_{11}$	$In_{12}$	$In_{13}$	$Out_{11}$
$In_{21}$	$In_{22}$	$In_{23}$	$Out_{21}$
$In_{31}$	$In_{32}$	$In_{33}$	$Out_{31}$
$In_{41}$	$In_{42}$	$In_{43}$	$Out_{41}$
$In_{51}$	$In_{52}$	$In_{53}$	$Out_{51}$
$In_{61}$	$In_{62}$	$In_{63}$	$Out_{61}$
$In_{71}$	$In_{72}$	$In_{73}$	?

+

Input			Out
$In_{11}$	$In_{12}$	$In_{13}$	$Out_{12}$
$In_{21}$	$In_{22}$	$In_{23}$	$Out_{22}$
$In_{31}$	$In_{32}$	$In_{33}$	$Out_{32}$
$In_{41}$	$In_{42}$	$In_{43}$	$Out_{42}$
$In_{51}$	$In_{52}$	$In_{53}$	$Out_{52}$
$In_{61}$	$In_{62}$	$In_{63}$	$Out_{62}$
$In_{71}$	$In_{72}$	$In_{73}$	?

- May want to predict more than one thing about each of the examples
- Split this into *n* separate problems

# Testing and Training

- **Big Problem:**

- *How do we know that the system has learnt correctly?*
- *How can we trust future 'guesses'?*
- *AKA Generalisation performance?*

- **Answer:**

- *Test the learner on known examples*

**Training Set**

**Testing Set**

Input	Output
Input 1	Output 1
Input 2	Output 2
Input 3	Output 3
Input 4	Output 4
Input 5	?
Input 6	?

# Supervised Machine Learning Example

- Learning **verb endings**
  - As with a human infant, it is shown a number of examples and has to try to generalise to new cases

**Training Set**

**Testing Set**

Input	Output
<i>Jumps</i>	<i>Jumped</i>
<i>Runs</i>	<i>Ran</i>
<i>Talks</i>	<i>Talked</i>
<i>Snows</i>	<i>Snowed</i>
<i>Sleeps</i>	<i>Slept</i>
<i>Walks</i>	<i>Walked</i>
<i>Eats</i>	<b>?</b>
<i>Looks</i>	<b>?</b>



# Types of Supervised Learning Problem

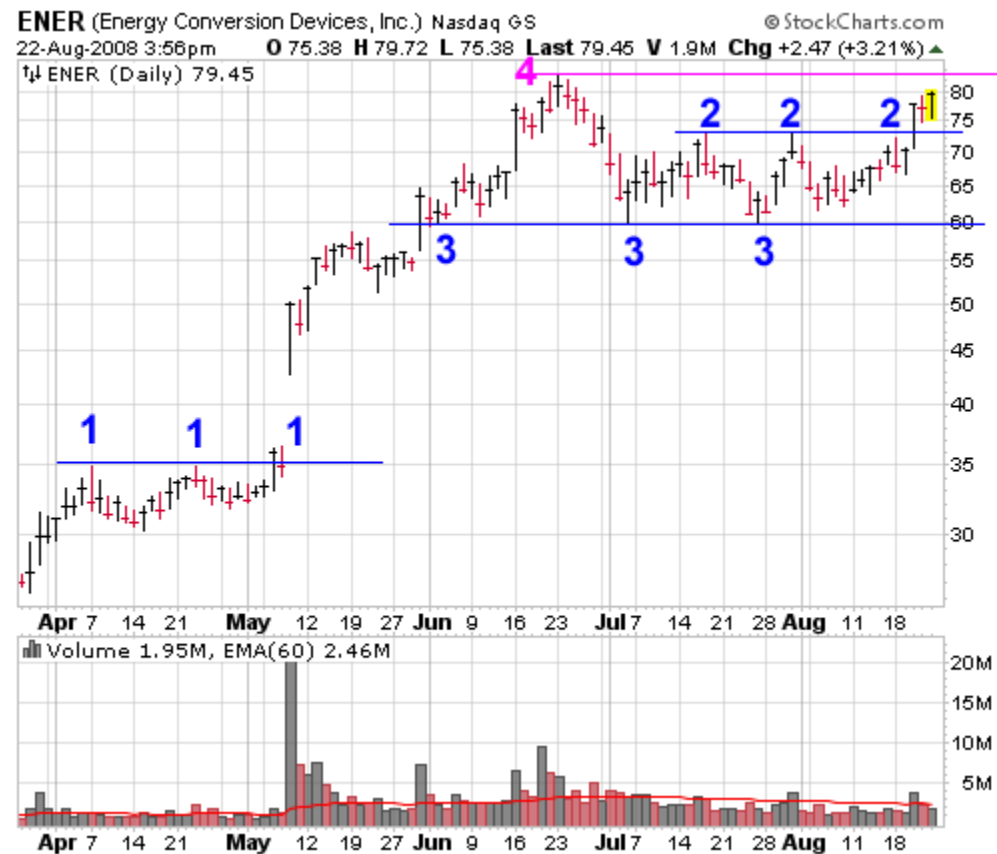
- Supervised learning problems can be characterised by the kinds of outputs they are looking for:
  - Decision Problems: yes/no answers (binary)
  - Prediction Problems: Real number outputs
  - Classification Problems: A,B,or C answers

# Example Supervised Learning Applications

- For each application **need to ask:**
  - *What are the inputs?*
  - *What are the outputs?*
  - *Decision or Prediction or Classification?*
  - *How will we test it?*

# Stock Market Prediction

- Inputs: Characteristics of previous movements
- Outputs: **buy/sell** recommendations
- **Decision** problem



# Stock Market Prediction

Stock	Volatility	1day movement	5day movement	Volume	%age short	Future Movement
IBM	8.9	+2.3	+2.6	42.3	1.2	-2.4
DELL	9.5	-2.4	+12.3	12.4	21.6	+12.4
MS	7.8	+12.4	-2.8	33.3	31.3	-22.3
SHELL	7.7	-22.3	+4.5	22.3	11.2	21.6
HP	9.7	-3.1	+12.3	13.1	5.4	buy/sell?

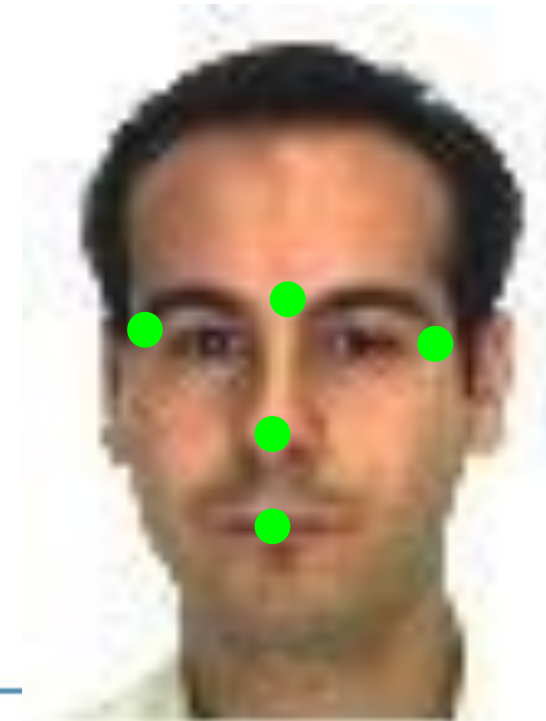
# Cancer Type Classification

- **Inputs** = expression levels of particular genes in sample from tumour
- **Outputs** = type of cancer
  - BL = Burkitt Lymphoma
  - EWS = Ewing Sarcoma
  - NB = neuroblastoma
  - RMS = rhabdomyosarcoma
- **Classification Problem**

IN				OUT
0.2779	0.5165	0.9175	4.5005	BL
0.0994	0.5097	0.8425	3.2358	BL
0.1012	0.9389	0.8719	3.2145	BL
0.2587	1.1716	0.9443	1.3051	BL
0.248	2.3301	1.7641	2.5211	BL
0.2568	1.1629	1.286	2.7443	BL
0.1924	1.2601	1.5374	1.7685	BL
0.0872	0.7068	0.6526	0.6641	EWS
0.4286	0.5497	0.4186	0.5765	EWS
0.0567	0.267	0.2012	0.479	EWS
0.056	0.3257	0.2299	0.2323	EWS
0.1164	0.6136	0.4159	1.0973	EWS
0.1991	0.4287	0.9301	0.3987	NB
0.2137	0.2514	0.4563	1.047	NB
0.4574	0.1699	1.0981	1.2523	NB
0.165	0.5502	0.2658	0.693	NB
0.1483	0.2747	0.76	0.667	NB
2.4431	0.3299	0.3888	0.9141	RMS
1.4417	0.3728	0.8756	0.8837	RMS
1.4669	0.1572	0.926	2.3591	RMS
0.959	0.2671	0.1464	1.4814	RMS
0.4919	0.1782	0.6528	0.5859	?
1.294	0.3598	0.3183	1.0059	?

# Defining Machine Learning Problems

- Many different problems can be posed as ML problems
- Often the key is to transform the original data into a form suitable for machine learning
- E.g. **Face Recognition**:
  - Take the original image and extract features, which can then be fed into a machine learner



# Face Recognition as a Machine Learning Problem

Eye Width	Face Width	Nose Length	Mouth Width	Eye Distance	Nose Width	Person
1.2	8.9	2.3	2.6	2.3	1.2	Jon
1.6	9.5	2.4	2.3	2.4	1.6	Jon
1.3	7.8	2.4	2.8	3.3	1.3	Jack
1.3	7.7	2.3	2.4	2.3	1.2	Jack
1.4	9.7	3.1	2.3	3.1	1.4	?

# From ML to DM

- **Machine Learning**
  - Old established academic AI discipline
- **Data Mining**
  - More recent business-oriented practice
  - First *International workshop on Knowledge Discovery and Data Mining* was in 1995
- *But really they're the same thing*



# What is Data Mining?

- Organisations have large repositories of data about customers, processes, employees, transactions, products,
- Many **are starting to realise** there may be **'hidden gold'**
  - Useful information hidden in the patterns and trends in the data
- Data Mining is the attempt to 'explicitise' that

# Data Mining Examples

- Targeted Marketing
  - **Business problem:** Who to target in a list of prospects for direct mailing campaign
  - **Solution:** Use Data Mining to identify most promising respondents by combining demographic and geographic data with data on past purchase behaviour
  - **Benefit:** Better response rate, savings in campaign cost

# Data Mining Examples

- Churn Analysis

- **Business Problem:** Telcos *etc* lose money on customers switching to other providers. Prevent loss of customers, avoid adding churn-prone customers
- **Solution:** Identify typical patterns of telephone usage of likely-to-defect and likely-to-churn customers
- **Benefit:** Retention of customers, more effective promotions

# Data Mining Examples

- Insurance Fraud Detection
  - **Business Problem:** car insurance costs inflated by fraudulent claims
  - **Solution:** use data mining to identify potential fraud cases based on data about previous known cases, and investigate these more carefully
  - **Benefits:** lower costs and increased profits

# Data Mining Examples

- Recommendation Systems
  - **Business Opportunity:** Online customers indicate preferences by their history of purchases
  - **Solution:** Use 'collaborative filtering' to cross-sell other items that may be of interest to a particular customer
  - **Benefit:** Increased sales
- Example: Netflix Prize

# Data Mining Examples

- Customer Conversion

- **Business Problem:** Only a small percentage of web site browsers actually buy
- **Solution:** Use 'clickstreams' to identify behavioural patterns of buyers. Redesign website to offer information that encourages purchases
- **Result:** increased sales

## *Computer Models Find Patterns In Asymmetric Threats*

- University of Alabama researchers to develop a database capable of [1]anticipating targets for future guerrilla attacks. Quoting Space War: "Adversaries the US currently faces in Iraq rely on surprise and apparent randomness to compensate for their lack of organization, technology, and firepower. 'One way to combat these attacks is to identify trends in the attackers' methods, then use those trends to predict their future actions,' said UA-Huntsville researcher Wes Colley. 'Some trends from these attacks show important day-to-day correlations. If we can draw inferences from those correlations, then we may be able to save lives by heightening awareness of possible events or changing the allocation of our security assets to provide more protection.' Researchers reviewed the behavior signatures of terrorists on 12,000 attacks between 2003 and mid-2007 to calculate relative probabilities of future attacks on various target types."

## *Toddlers May Learn Language By Data Mining*

- “Toddlers' brains can effortlessly do what the most powerful computers with the most sophisticated software cannot: learn language simply by hearing it used. A ground-breaking new theory postulates that young children are able to learn large groups of words rapidly by data-mining. ... After viewing various combinations of words and images the children were surprisingly successful at figuring out which word went with which picture. Yu and Smith say it's possible that the more words tots hear, and the more information available for any individual word, the better their brains can begin simultaneously ruling out and putting together word-object pairings, thus learning what's what. Yu says if they can identify key factors involved in this form of learning and how it can be manipulated, they might be able to make learning languages easier for children and adults. Understanding children's learning mechanisms could also further machine learning.”



# Suggested Reading


- *White Paper on Machine Learning*, Tom Mitchell, Carnegie Mellon University

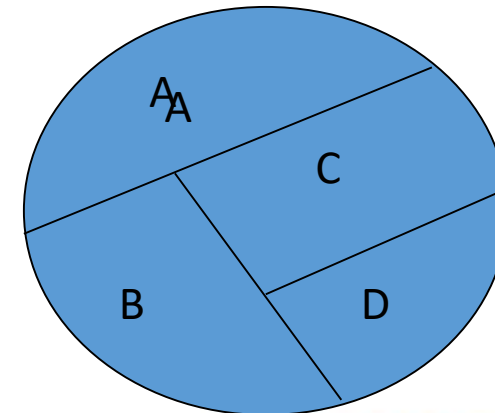
# Classification Systems

# Overview

- Classification Problems
- The KNN Classifier
- Euclidean Distance
- Normalisation
- Example

# Classifications

- A Classification is a  *partition* of a set of data
  - Every member of the set belongs to *exactly one* class
  - *ie* none left out
  - and none in multiple classes
  - and no subsets



# Classification Problem

- Classification is a supervised learning problem in which we try to predict class membership on the basis of the membership of previous examples

Input	Class
<i>Input 1</i>	<i>A</i>
<i>Input 2</i>	<i>A</i>
<i>Input 3</i>	<i>B</i>
<i>Input 4</i>	<i>B</i>
<i>Input 5</i>	<i>C</i>
<i>Input 6</i>	<i>C</i>
<i>Input 7</i>	<i>?</i>

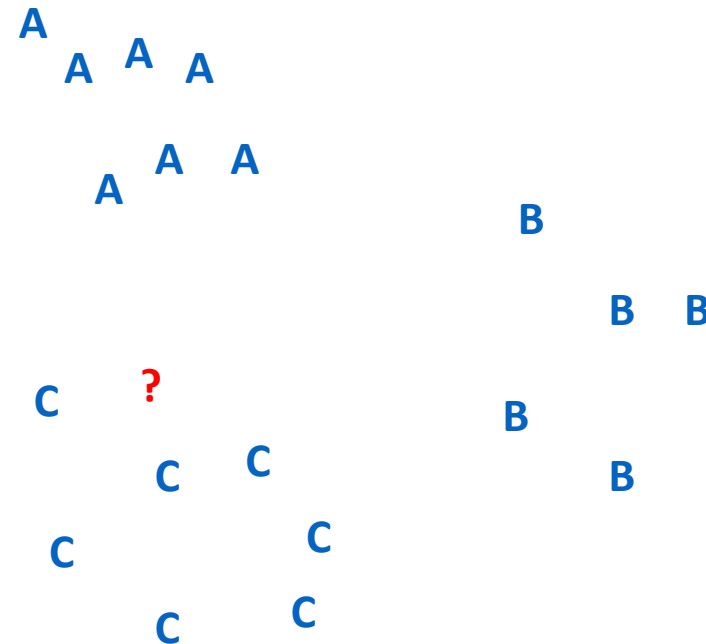
# Cancer Type Classification

- **Inputs** = expression levels of particular genes in sample from tumour
- **Outputs** = type of cancer
  - BL = Burkitt Lymphoma
  - EWS = Ewing Sarcoma
  - NB = neuroblastoma
  - RMS = rhabdomyosarcoma
- Used to determine **best treatment** for unknown tumour (**supervised**)
- Or to **discover new subtypes** (**unsupervised**)

	IN			OUT
0.2779	0.5165	0.9175	4.5005	BL
0.0994	0.5097	0.8425	3.2358	BL
0.1012	0.9389	0.8719	3.2145	BL
0.2587	1.1716	0.9443	1.3051	BL
0.248	2.3301	1.7641	2.5211	BL
0.2568	1.1629	1.286	2.7443	BL
0.1924	1.2601	1.5374	1.7685	BL
0.0872	0.7068	0.6526	0.6641	EWS
0.4286	0.5497	0.4186	0.5765	EWS
0.0567	0.267	0.2012	0.479	EWS
0.056	0.3257	0.2299	0.2323	EWS
0.1164	0.6136	0.4159	1.0973	EWS
0.1991	0.4287	0.9301	0.3987	NB
0.2137	0.2514	0.4563	1.047	NB
0.4574	0.1699	1.0981	1.2523	NB
0.165	0.5502	0.2658	0.693	NB
0.1483	0.2747	0.76	0.667	NB
2.4431	0.3299	0.3888	0.9141	RMS
1.4417	0.3728	0.8756	0.8837	RMS
1.4669	0.1572	0.926	2.3591	RMS
0.959	0.2671	0.1464	1.4814	RMS
0.4919	0.1782	0.6528	0.5859	RMS
1.294	0.3598	0.3183	1.0059	RMS

# The K-Nearest Neighbour Classifier

- A simple but effective general purpose classifier
- What class is ?
  - Probably **C**
- Why?
  - *Because the other similar examples are all C*
  - *ie all its Nearest Neighbours are C*
- KNN predicts classification based on the classification of the  $K$  nearest neighbours
  - ( $K=1$ , or 2 or 3 etc)



# KNN Example:

## *Predicting Voting Behaviour*

- Suppose you wanted to predict how someone was likely to vote. What information would you want to know?
  - Age
  - Income
  - Location
  - Job type
  - Gender
  - *etc*



# KNN Distance Measures

- In order to use KNN we need to know the distance (*ie* similarity) between instances
- Need to combine differences in many different variables
  - Possibly of different types
- Want to predict Jon's voting behaviour, but which two are more similar: Jack and Jon *or* Jill and Jon?

	Age	Gender	Income	Weight	Vote
Jack	25	Male	£40,000	65Kg	Labour
Jill	26	Female	£35,000	56Kg	Tory
Jon	34	Male	£28,000	78Kg	?

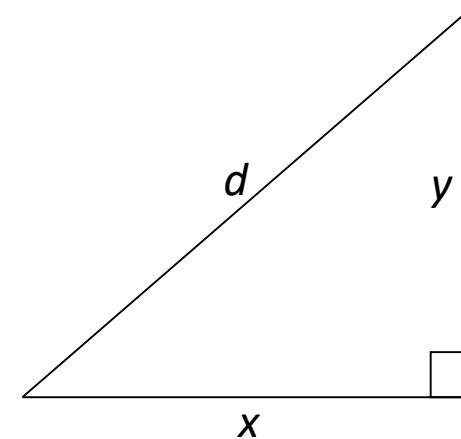
# Euclidean Distance

- A general purpose way of finding distances between real-valued data
- How to find  $d$  given  $x$  and  $y$ ?
- Pythagoras' Theorem:

$$x^2 + y^2 = d^2$$

*therefore*

$$d = \sqrt{x^2 + y^2}$$

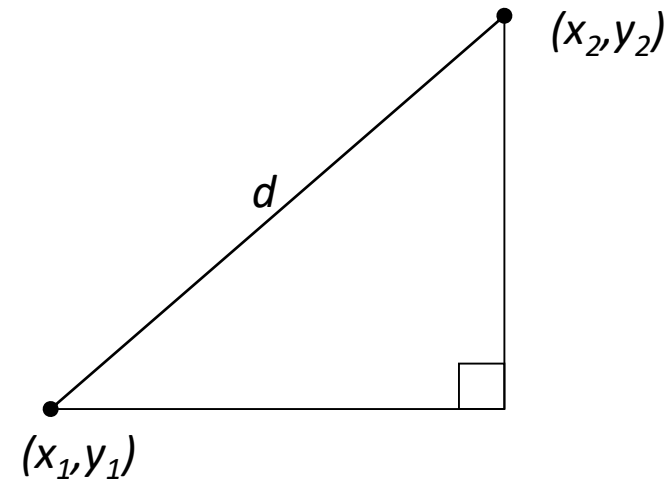


# 2D Euclidean Distance

- Can use Pythagoras' Theorem to define *Euclidean Distance* between points in 2D space

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

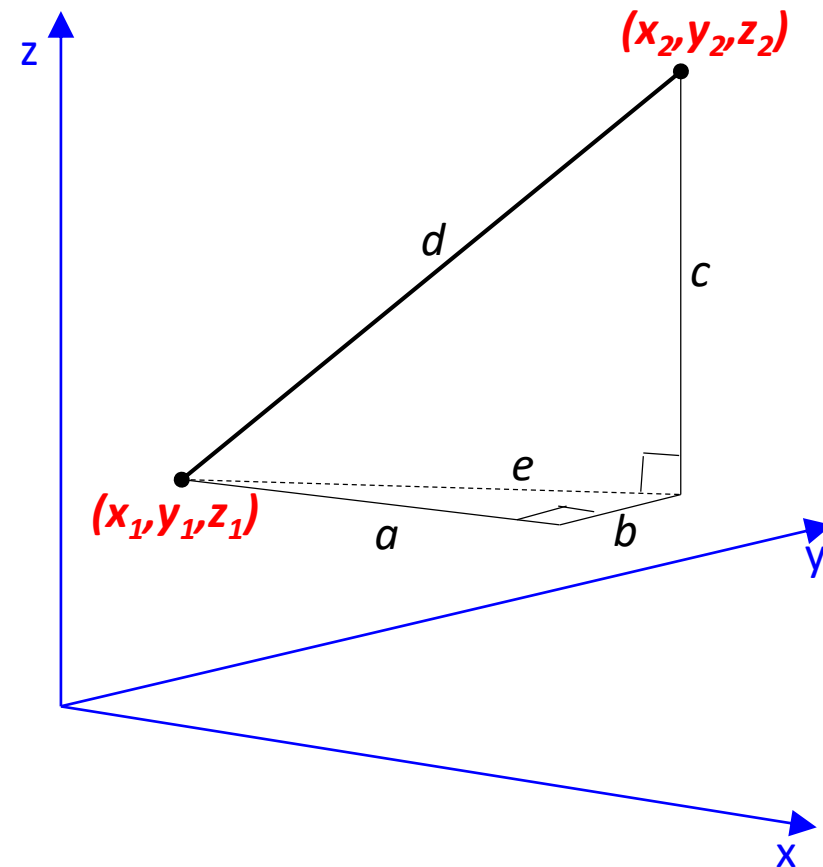
- But what if  $x_1 > x_2$ , or  $y_1 > y_2$ ?



# Generalising Euclidean Distance

- eg 3 dimensions:
  - 2 right-triangles:
$$d^2 = e^2 + c^2$$
$$e^2 = a^2 + b^2$$
  - therefore
$$d^2 = a^2 + b^2 + c^2$$

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



# Generalised Euclidean Distance

- In general, given two points in **N-space**

$$a = (a_1, a_2, \dots, a_n)$$

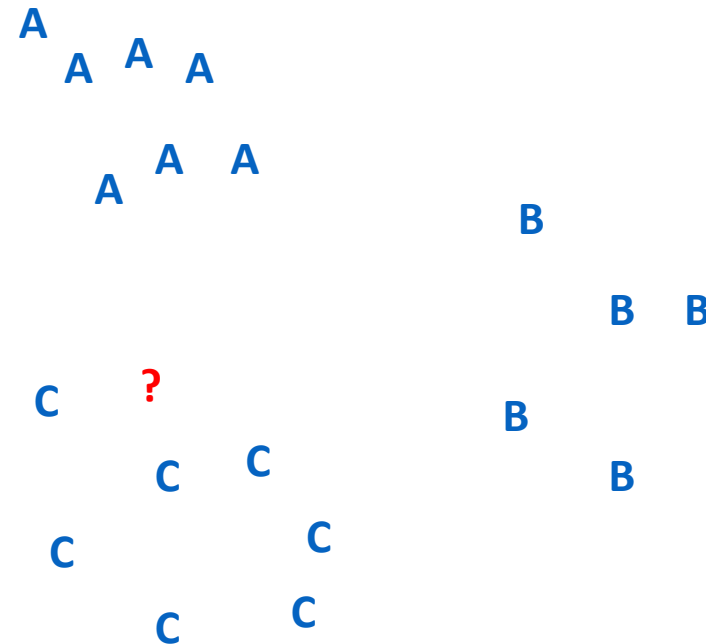
$$b = (b_1, b_2, \dots, b_n)$$

Then the *Euclidean Distance* between them

$$\begin{aligned} D(a,b) &= \sqrt{(a_1-b_1)^2 + (a_2-b_2)^2 + (a_3-b_3)^2 + \dots + (a_n-b_n)^2} \\ &= \sqrt{\sum (a_i-b_i)^2} \end{aligned}$$

# The K-Nearest Neighbour Classifier

- *To find the class of a new instance:*
  - Find the (Euclidean) distance from the new instance to each known instance
  - Find the  $k$  nearest neighbours
  - The class of the new instance is then the most popular class amongst the  $K$  neighbours



# Non-Real Data Types

- Euclidean Distance only works with real-valued data
  - ie where we can calculate the numerical difference between values
  - $\text{difference}(35,28) = (35-28) = 7$  etc
- But what is the difference between nominal or boolean values?
  - $\text{difference}(\text{male-female}) = ?$
  - $\text{difference}(\text{true-false}) = ?$

# Non-Real Data

- One solution:
  - $\text{difference}(a,b) = 0$  if  $a=b$
  - $\text{difference}(a,b) = 1$  if  $a \neq b$
- For example
  - $\text{difference}(\text{male}, \text{female}) = 1$
  - $\text{difference}(\text{female}, \text{female}) = 0$
  - $\text{difference}(\text{tory}, \text{labour}) = 1$
  - $\text{difference}(\text{tory}, \text{tory}) = 0$



# Normalisation

- One problem with Euclidean Distance is that it combines different types of attribute
  - age, income, weight, etc
  - *The larger attributes can 'out-weigh' the smaller ones*

# Normalisation Example

	Age	Income
Jack	25	£20,000
Jill	44	£21,000
Jon	25	£21,000

Which two are more similar:

- Jon & Jack
- Jon & Jill?

$$D(\text{Jack}, \text{Jon}) = \sqrt{(25-25)^2 + (21000-20000)^2} = \sqrt{0 + 1000000} = 1000$$

$$D(\text{Jill}, \text{Jon}) = \sqrt{(44-25)^2 + (21000-21000)^2} = \sqrt{324 + 0} \approx 18$$

*Jon is more similar to Jack than to Jill, but the **small differences** in **income** between Jon and Jack '**hides**' the difference in their **ages***

# Normalisation

- So **'normalise'** the data
- *ie* put it all in the same range
  - Typically 0 to 1 or -1 to 1
- For example
  - age = 0 to 100 (say), so divide actual age by 100
  - income = 0 to 100,000 (say), so divide actual age by 100,000

	Age	Income
Jack	25	£20,000
Jill	44	£21,000
Jon	25	£21,000

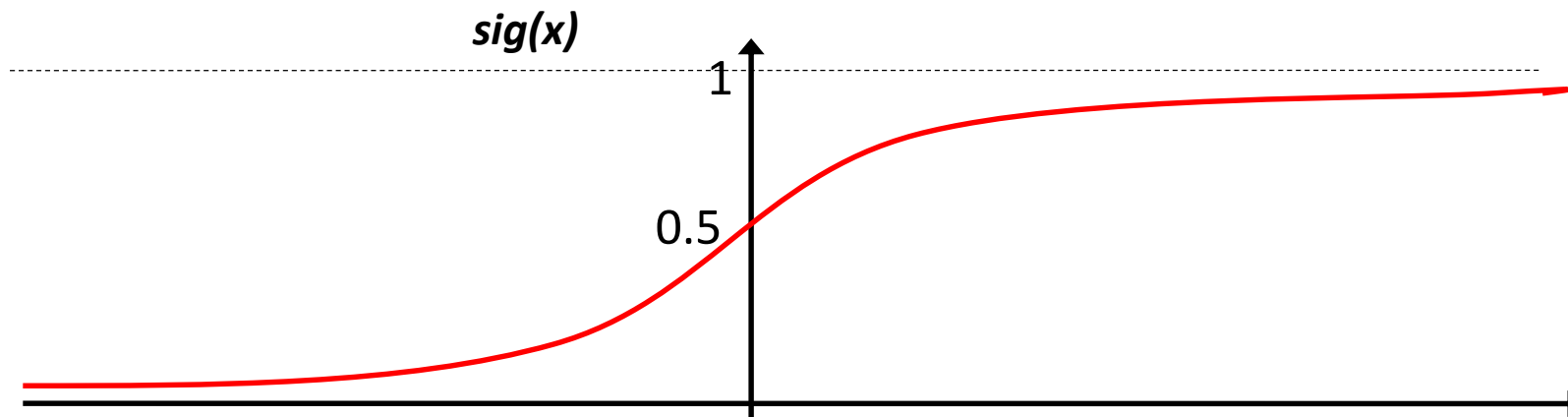


	Age	Income
Jack	0.25	0.20
Jill	0.44	0.21
Jon	0.25	0.21

# Unbounded Normalisation

- But what if an attribute is unbounded?
- Can use a sigmoidal (squashing) function to **normalise** into a range **[0,1]**
- Two *asymptotes*:
  - $\text{sig}(x) \rightarrow 1$  as  $x \rightarrow \infty$
  - $\text{sig}(x) \rightarrow 0$  as  $x \rightarrow -\infty$

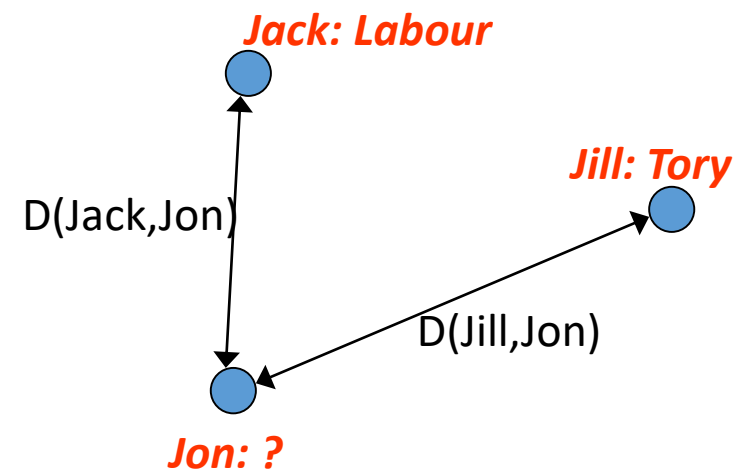
$$\text{sig}(x) = \frac{1}{1 + e^{-x}}$$



# KNN Example

	Age	Gender	Income	Weight	<i>Vote</i>
<b>Jack</b>	25	Male	£40,000	65Kg	<i>Labour</i>
<b>Jill</b>	26	Female	£35,000	56Kg	<i>Tory</i>
<b>Jon</b>	34	Male	£28,000	78Kg	<i>?</i>

- Normalise data
- Calculate distances
- Find nearest neighbour(s)



# KNN Example

	Age	Gender	Income	Weight	Vote
Jack	0.25	Male	0.40	0.65	Labour
Jill	0.26	Female	0.35	0.56	Tory
Jon	0.34	Male	0.28	0.78	?

$$\begin{aligned} D(\text{Jon}, \text{Jack}) &= \sqrt{(0.34-0.25)^2 + 0^2 + (0.28-0.40)^2 + (0.78-0.65)^2} \\ &= \sqrt{0.081 + 0 + 0.0144 + 0.0169} = \sqrt{0.1123} = 0.335 \end{aligned}$$

$$\begin{aligned} D(\text{Jon}, \text{Jill}) &= \sqrt{(0.34-0.26)^2 + 1^2 + (0.28-0.35)^2 + (0.78-0.56)^2} \\ &= \sqrt{0.064 + 1 + 0.049 + 0.0484} = \sqrt{1.164} = 1.077 \end{aligned}$$

- So **Jon** is most likely to **vote Labour**
- Would have to test on larger data sets to determine how reliable this is
- May need to alter weighting/normalisation to get successful predictions

# Other KNN Issues

- What is  $K$ ?
  - Best value depends on the data
  - Typically 1, 3, or 5
- What if there is a tie?
  - Use some form of tie-breaking
  - *eg* weight 'votes' by distance
- Other distance measures are available:
  - Maximum:  $\max |a_i - b_i|$
  - City Block (Manhattan):  $\sum |a_i - b_i|$
  - Lot of research about best measures for particular applications

# Reading

- K. K. Fukunaga and P.M. Narendra - *A branch and bound algorithm for computing k-nearest neighbours*. IEEE Transactions on Computers, 24:750-753, 1975.



# Stretch Break

- Seminar

Thank you