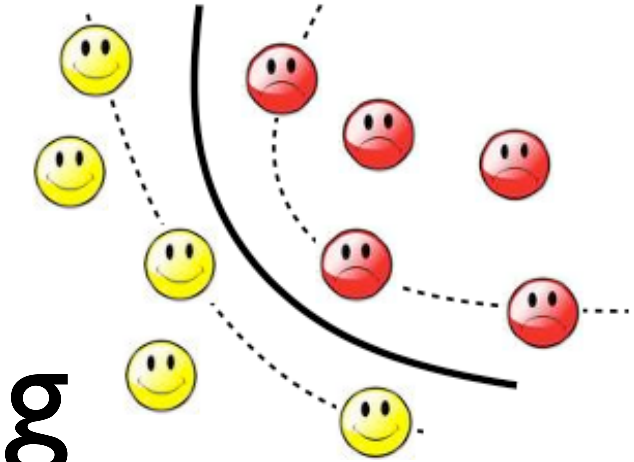
THE UNIVERSITY OF SYDNEY



Machine Learning and Data Mining

(COMP 5318)

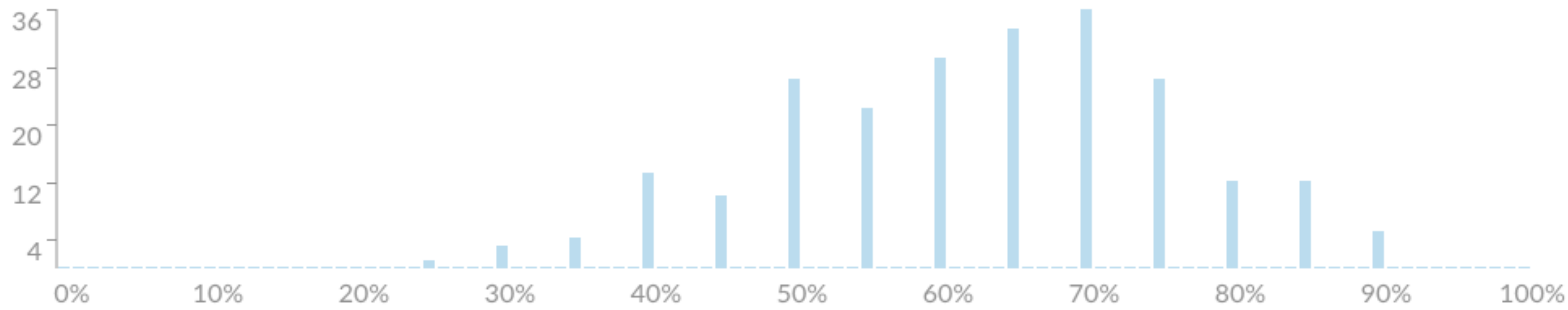
Basics of Classification and ROC curves

Fabio Ramos
Roman Marchant

Quiz results

Quiz Summary

Average Score	High Score	Low Score	Standard Deviation	Average Time
63%	90%	25%	13.54	53:48



Announcements

- Feedback from Quiz will be available after all students have taken the Quiz. We will announce later.
- Assignment 1 will be available later this week
  - Assignment 1 due on 07/05, 5pm
- This lecture is based on:
  - Murphy's book 1.4, 3.5

Assignment 1

THE UNIVERSITY OF SYDNEY

Summary

The goal of this assignment is to build a classifier to classify apps from the Apps Market into a set of categories based on their descriptions. The dataset is quite large, so you need to be smart on which method you gonna use and perhaps perform a pre-processing step to reduce the amount of computation. Part of your marks will be a function of the performance of your classifier on the test set.

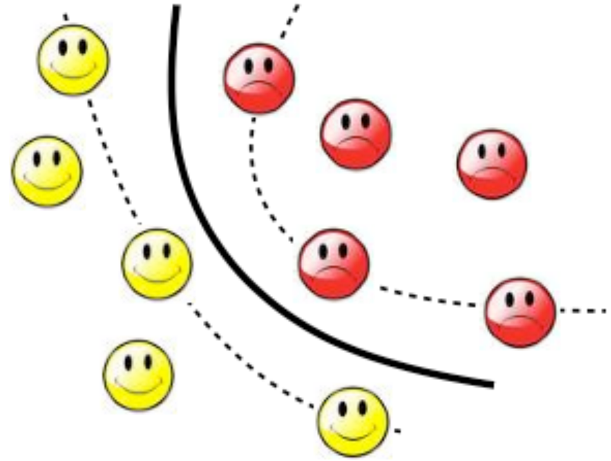
Assignment 1

1 Dataset description

The dataset is collected from the Apps Market. There are four main files:

1. training_data.csv:

   - There are 20,104 rows; each row corresponds to an app.
   - For each row, each column is separated by comma (,). The first column is the app's name, with the remaining columns containing the tf-idf values. The tf-idf values are extracted from words in the description of each app. We have done some pre-processing steps which resulted in 13,626 unique words. If a word is found in the description of an app, it has a tf-idf value (the tf-idf value is not zero). On the other hand, its tf-idf value is equal to zero if the word is not found in the description of the app. More information about tf-idf could be found in http://en.wikipedia.org/wiki/Tf%E2%80%93idf
   - In summary, data_train.txt is a matrix with dimension: $20{,}104 \times 13{,}627$ (remember the first column is the app's name).



Classification

Supervised learning

- Learn a mapping function *f* from *x* to *t*

$$t = f(\mathbf{x})$$

- If $t \in \{1, 2, 3, \ldots, C\}$ the problem is called classification
- If $t \in \mathbb{R}$ the problem is called regression

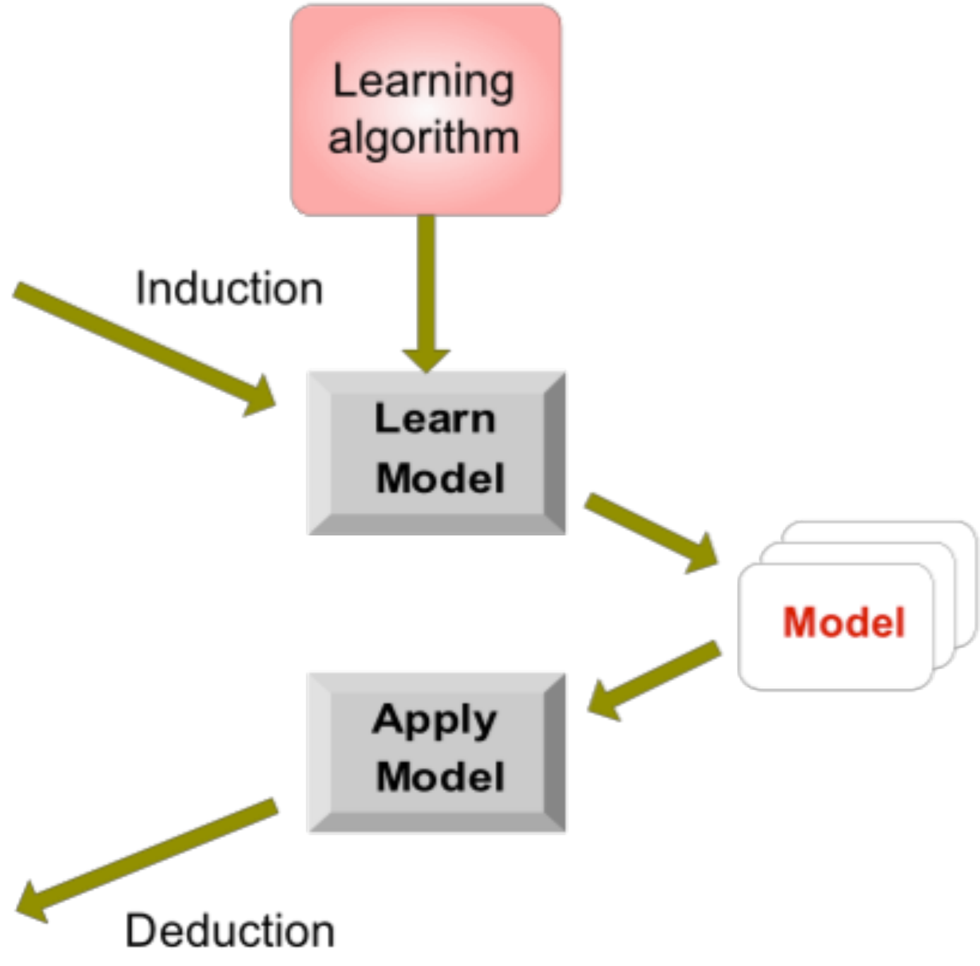
Classification: Definition

- Given a collection of records (*training set*)
  - Each record contains a set of *attributes*, one of the attributes is the *class*.
- Find a *model* for class attribute as a function of the values of other attributes.
- Goal: previously unseen records should be assigned a class as accurately as possible.
  - A *test set* is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

Illustrating Classification Task

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

Training Set



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

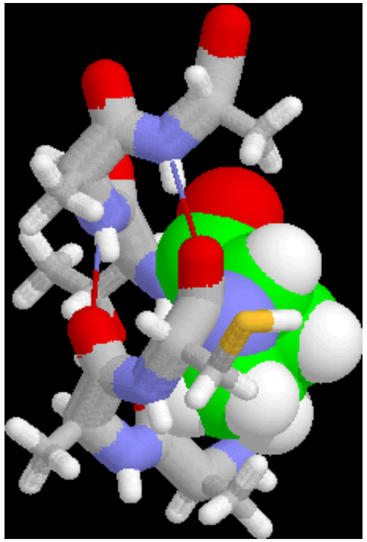
Test Set

Examples of Classification Tasks

- Predicting tumour cells as benign or malignant

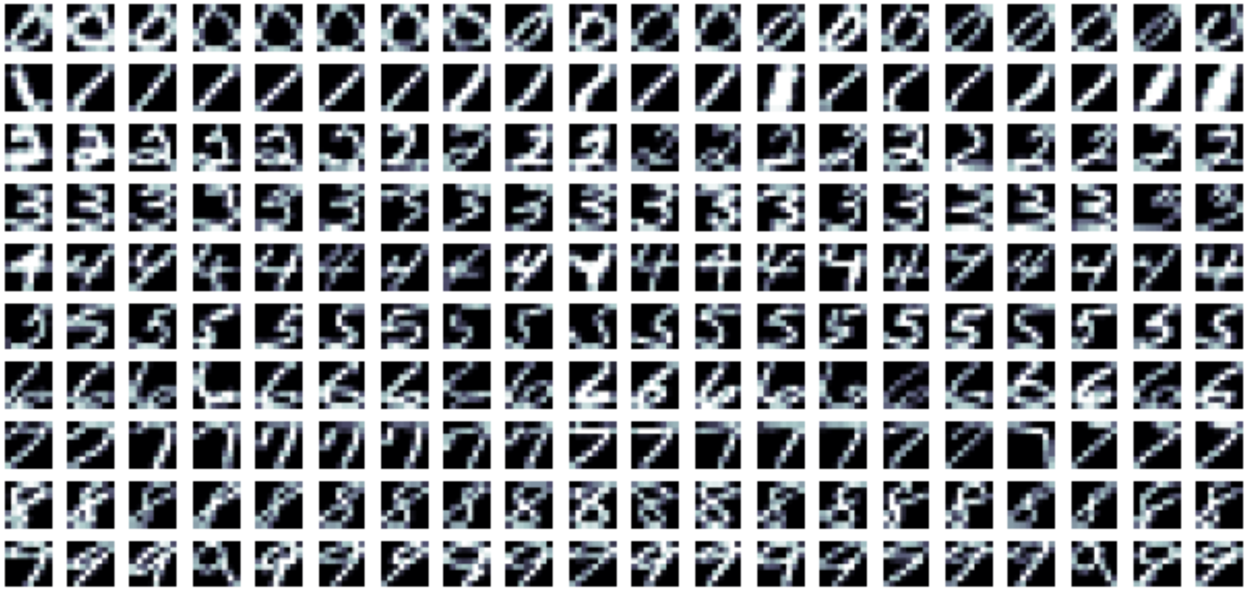
- Classifying credit card transactions as legitimate or fraudulent

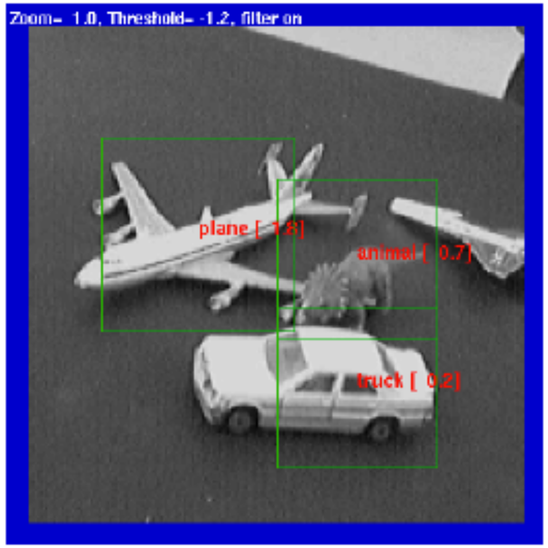
- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil



- Categorising news stories as finance, weather, entertainment, sports, etc

Object and handwriting recognition

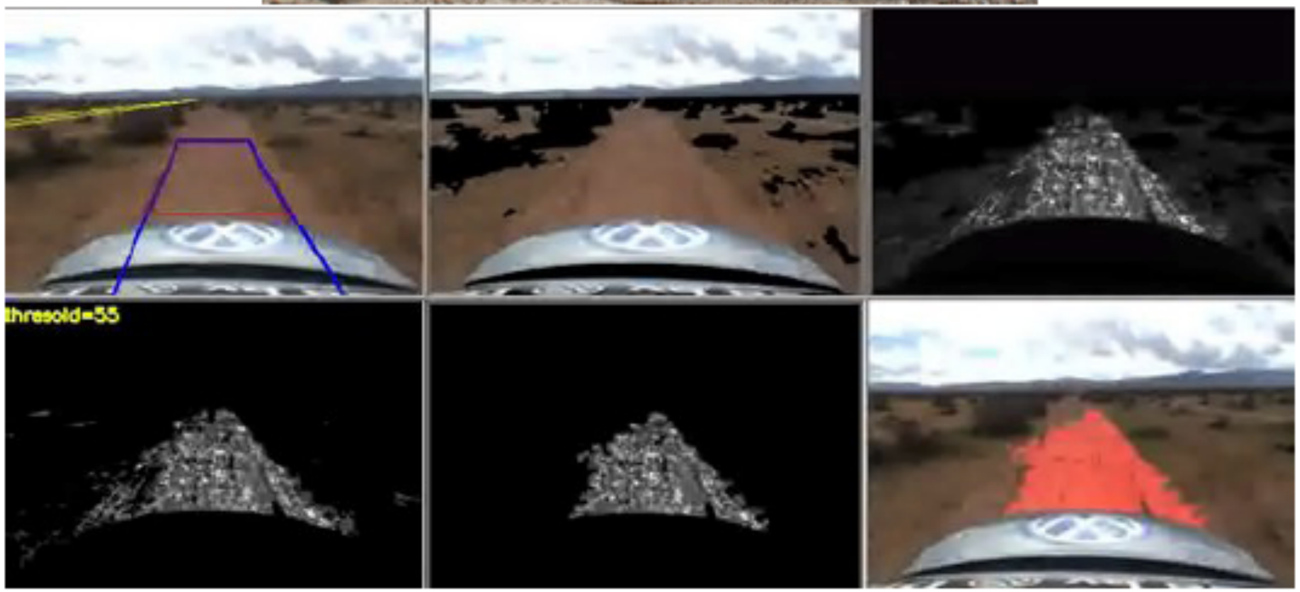




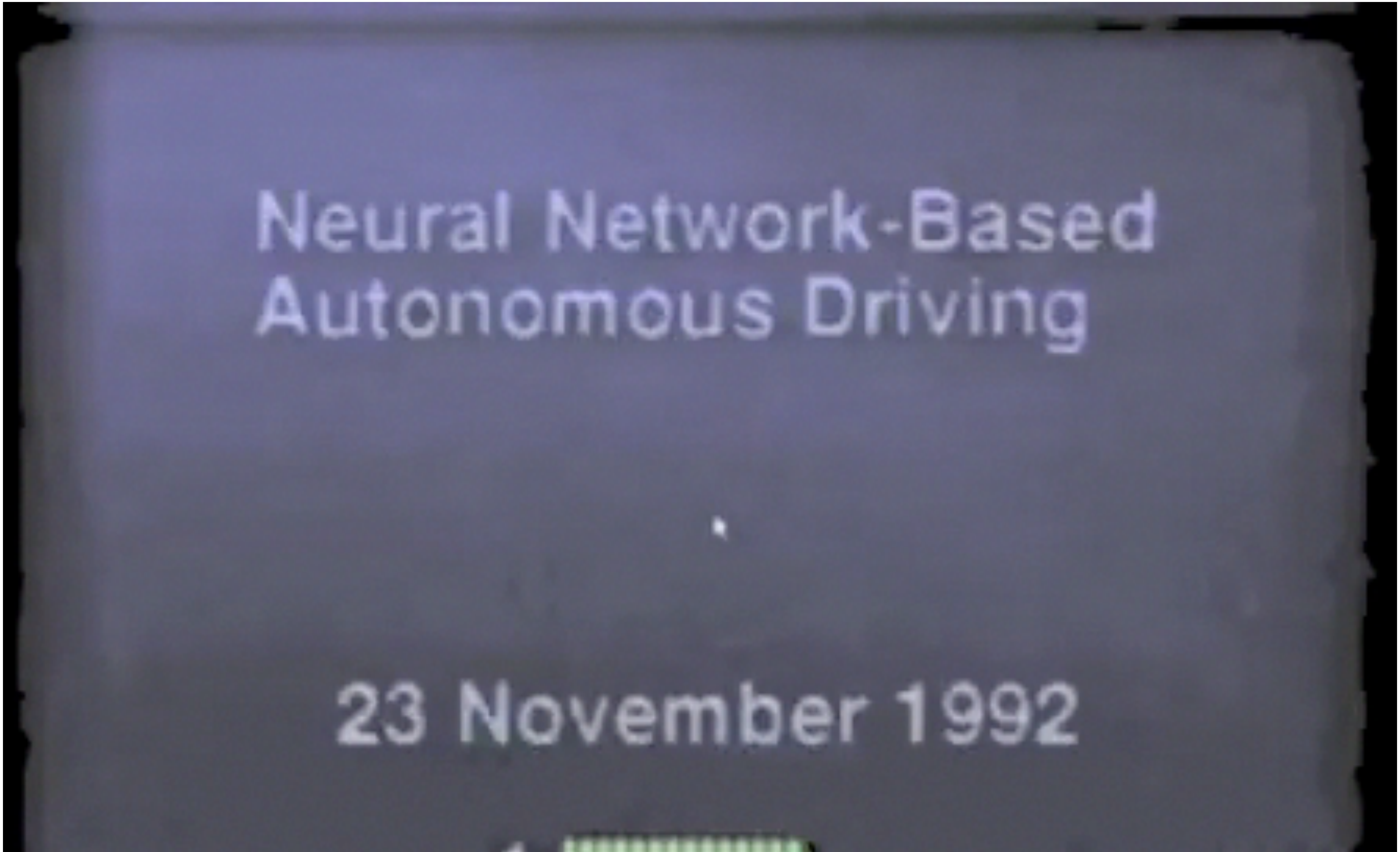
(NORB image from Yann LeCun)

Robotics

THE UNIVERSITY OF SYDNEY

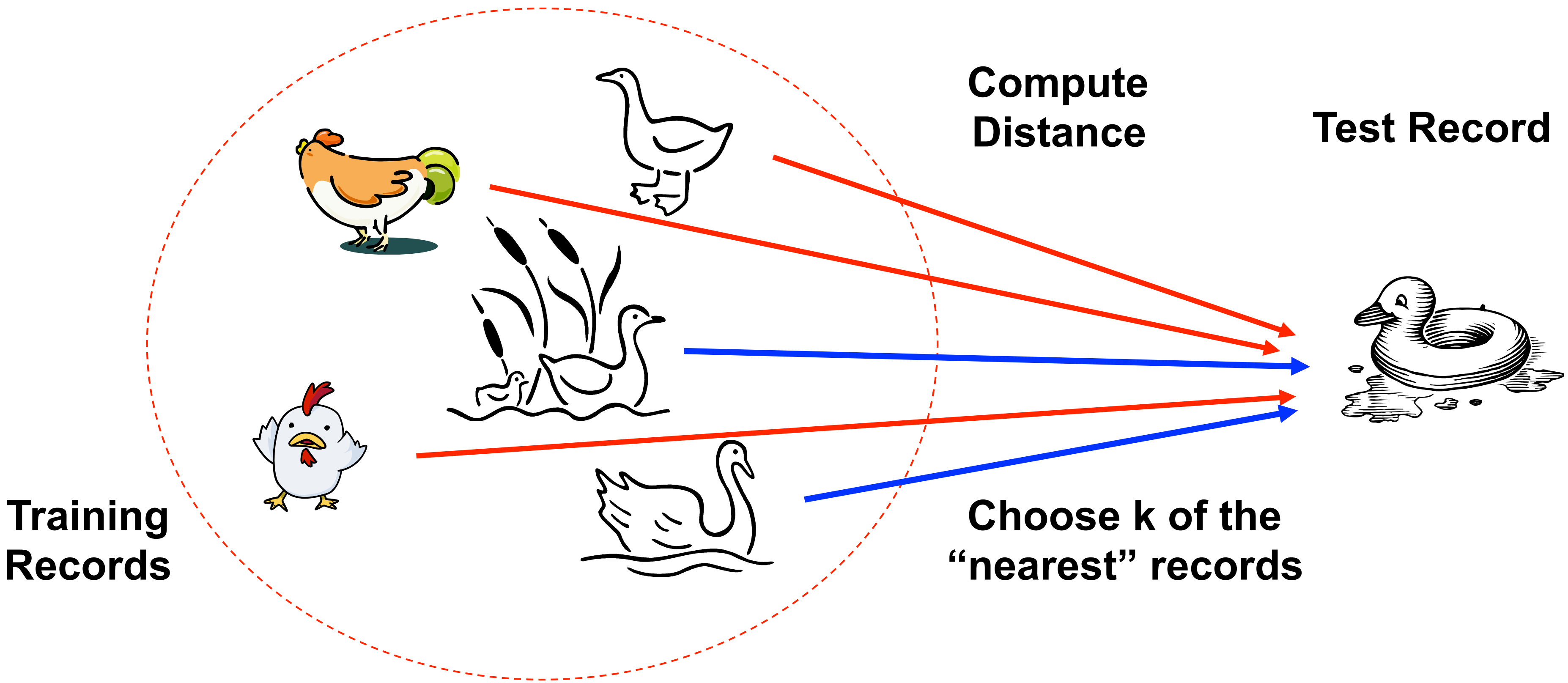


Classification for Autonomous Driving

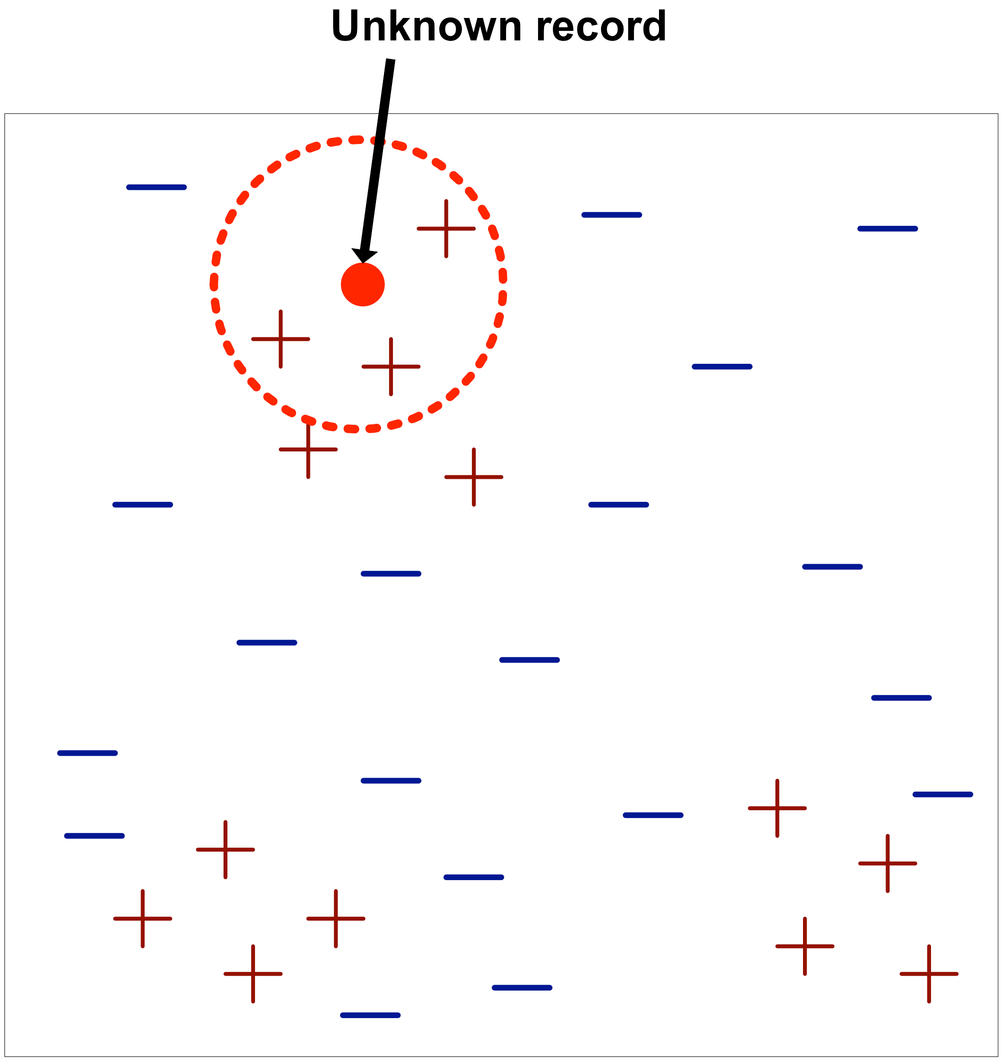


Nearest Neighbour Classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



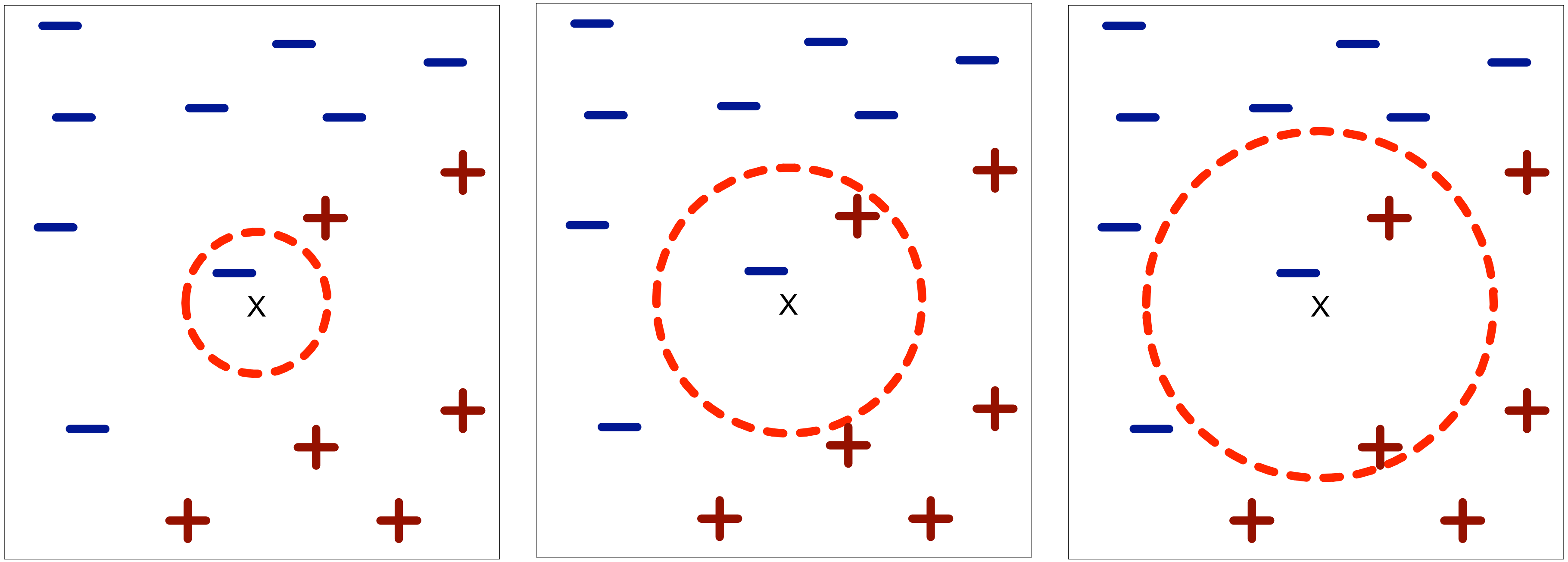
Nearest Neighbour Classifiers



- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbours to retrieve

- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbours
  - Use class labels of nearest neighbours to determine the class label of unknown record (e.g., by taking majority vote)

Definition of Nearest Neighbour

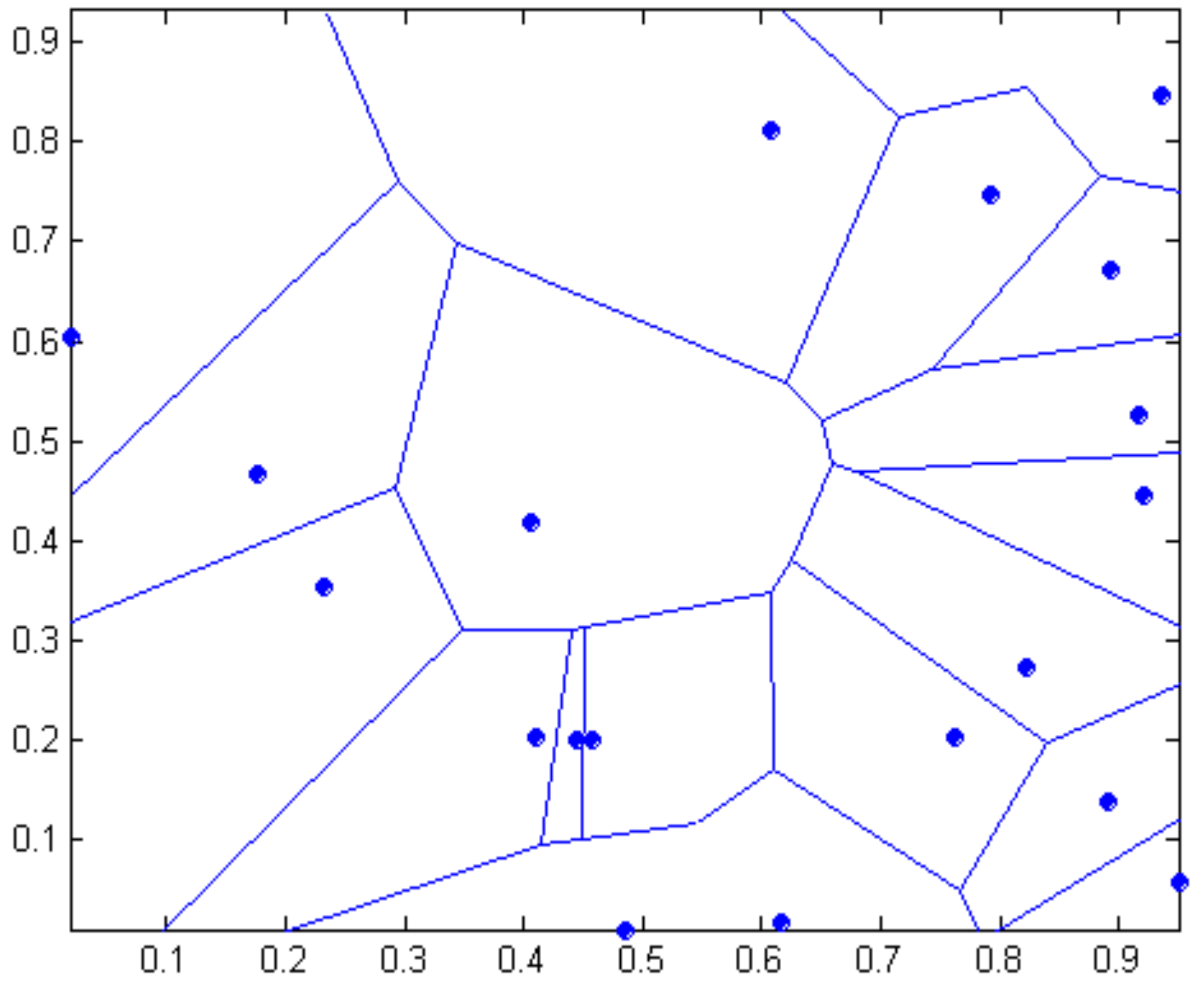


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

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

1 nearest-neighbour

Voronoi Diagram



Nearest Neighbour Classification

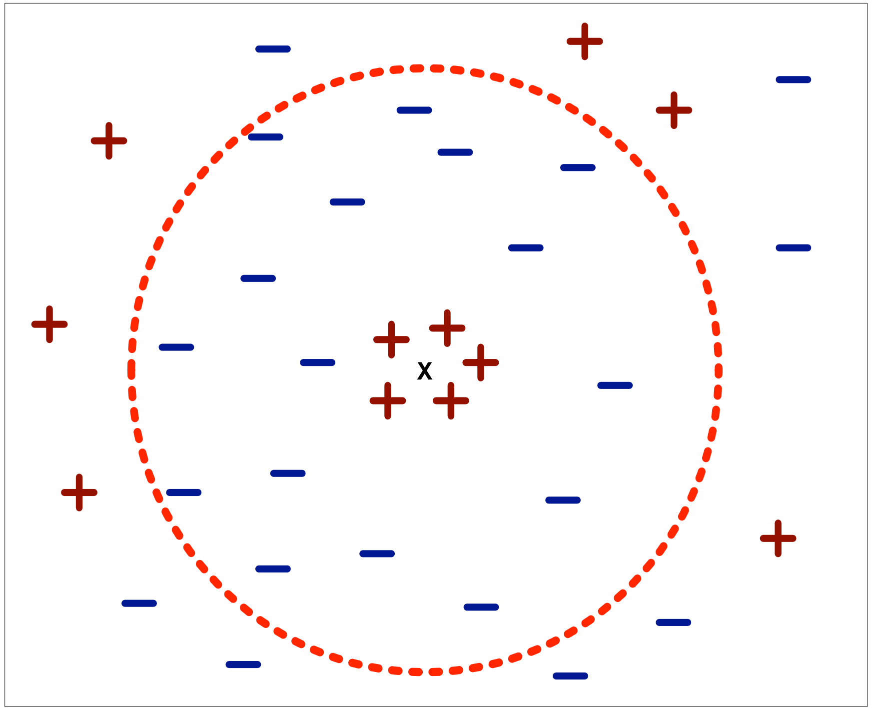
- Compute distance between two points:
  - Euclidean distance

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Determine the class from nearest neighbour list
  - Take the majority vote of class labels among the k-nearest neighbours
  - Weight the vote according to distance
    - weight factor, $w = 1/d^2$

Nearest Neighbour Classification

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighbourhood may include points from other classes



Nearest Neighbour 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 40kg to 150kg
    - income of a person may vary from $10K to $1M

Nearest Neighbour Classification

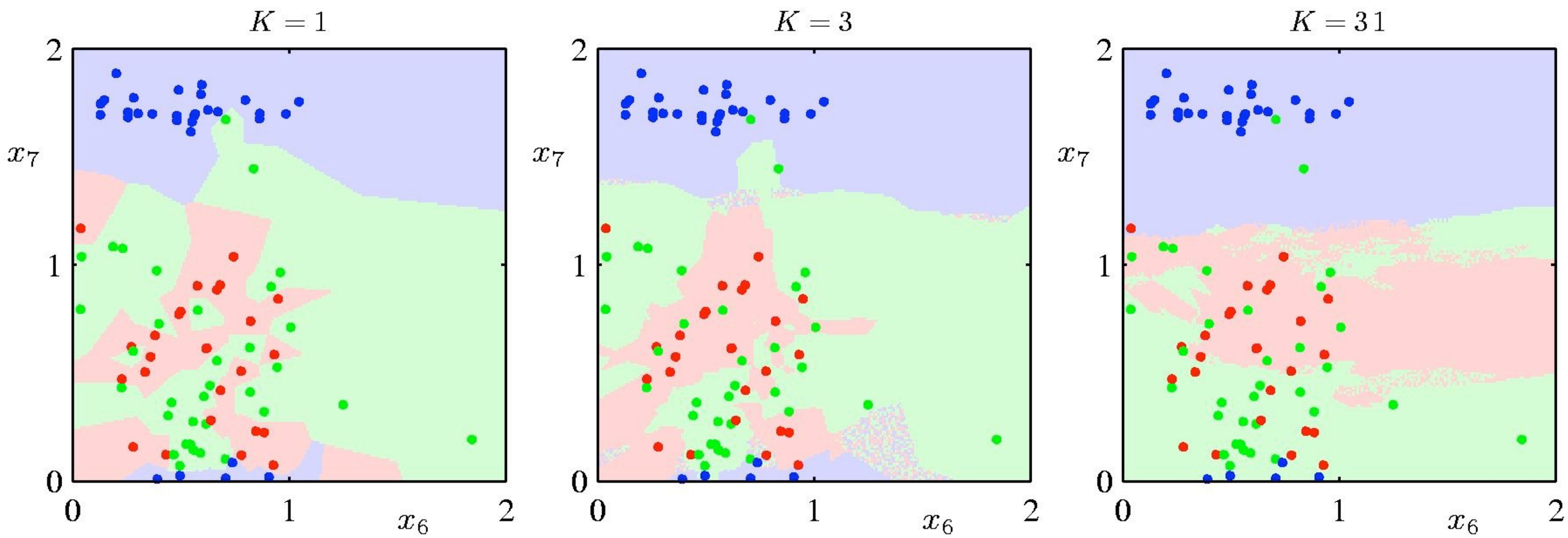
- Problem with Euclidean measure:
  - High dimensional data
    - curse of dimensionality
  - Can produce counter-intuitive results

1 1 1 1 1 1 1 1 1 1 1 0		1 0 0 0 0 0 0 0 0 0 0 0	
	vs		
0 1 1 1 1 1 1 1 1 1 1 1		0 0 0 0 0 0 0 0 0 0 0 1	

d = 1.4142 d = 1.4142

◆ Solution: Normalise the vectors to unit length

Nearest Neighbour Classification



- K acts as a smother
- For $N \to \infty$, the error rate of the 1-nearest-neighbour classifier is never more than twice the optimal error (obtained from the true conditional class distributions).

Online Self-Supervised Multi-Instance Segmentation of Dynamic Objects

Alex Bewley, Vitor Guizilini,
Fabio Ramos and Ben Upcroft

QUT CyPhy Lab

THE UNIVERSITY OF SYDNEY

k-NN Classifier Summary

- k-NN classifiers are lazy learners
  - It does not build models explicitly
  - Unlike eager learners such as decision tree induction and rule-based systems
  - Classifying unknown records are relatively expensive

Bayesian Classifier

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

$$P(C \mid A) = \frac{P(A, C)}{P(A)}$$

$$P(A \mid C) = \frac{P(A, C)}{P(C)}$$

- Bayes theorem:

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

Bayesian Classifiers

- Consider each attribute and class label as random variables
- Given a record with attributes $(A_1, A_2, \ldots, A_n)$
  - Goal is to predict class C
  - Specifically, we want to find the value of C that maximises $P(C \mid A_1, A_2, \ldots, A_n)$
- Can we estimate $P(C \mid A_1, A_2, \ldots, A_n)$ directly from data?

Bayesian Classifiers

- Approach:
  - compute the posterior probability $P(C \mid A_1, A_2, \ldots, A_n)$ for all values of C using the Bayes' theorem

$$\boxed{P(C \mid A_1 A_2 \ldots A_n) = \frac{P(A_1 A_2 \ldots A_n \mid C) P(C)}{P(A_1 A_2 \ldots A_n)}}$$

  - Choose value of C that maximises
    $P(C \mid A_1, A_2, \ldots, A_n)$
  - Equivalent to choosing value of C that maximises
    $P(A_1, A_2, \ldots, A_n \mid C)\, P(C)$
- How to estimate $P(A_1, A_2, \ldots, A_n \mid C)$?

Naïve Bayes Classifier

- Assume independence among attributes A_i when class is given:
  - $P(A_1, A_2, \ldots, A_n | C) = P(A_1| C_j)\, P(A_2| C_j) \ldots P(A_n| C_j)$
  - Can estimate $P(A_i| C_j)$ for all A_i and C_j.
  - New point is classified to C_j if $P(C_j) \prod P(A_i| C_j)$ is maximal.

How to Estimate Probabilities from Data?

THE UNIVERSITY OF SYDNEY

#	Refund	Status	Salary	Class
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

- Class: $P(C) = N_c/N$
  - e.g., $P(No)$ = 7/10, $P(Yes)$ = 3/10
- For discrete attributes:

$$P(A_i \mid C_k) = |A_{ik}| / N_{c^k}$$

  - where $|A_{ik}|$ is number of instances having attribute A_i and belongs to class C_k
  - Examples:
    $P(Status{=}Married|No)$ = 4/7
    $P(Refund{=}Yes|Yes)$ = 0

Example of Naïve Bayes Classifier

THE UNIVERSITY OF SYDNEY

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

A: attributes

M: mammals

N: non-mammals

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

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

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

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

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

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

=> Mammals

Bayesian classifier continuous

Generative model:

$$p(\mathbf{x}, \mathcal{C}_k) = p(\mathcal{C}_k) p(\mathbf{x}_n | \mathcal{C}_k) = \pi \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \Sigma_k)$$

class conditional density

class prior

class posterior

$$p(\mathcal{C}_k | \mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k) p(\mathcal{C}_k)}{\sum_{\mathcal{C}_j} p(\mathbf{x}|\mathcal{C}_j) p(\mathcal{C}_j)}$$

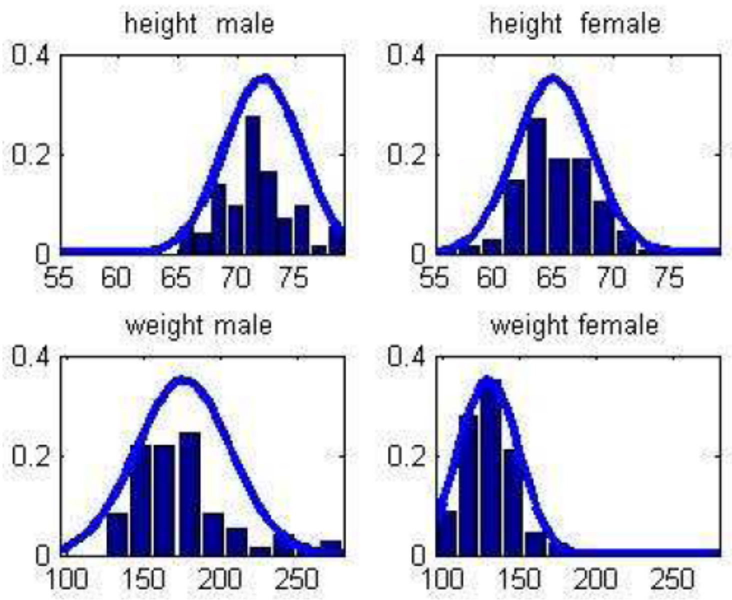
normalising constant

Bayesian classifier continuous

Independence assumption for $\mathbf{x} \in \mathbb{R}^d$

$$x_j | \mathcal{C}_k \sim \mathcal{N}(\mu_{j,k}, \sigma_{j,k})$$

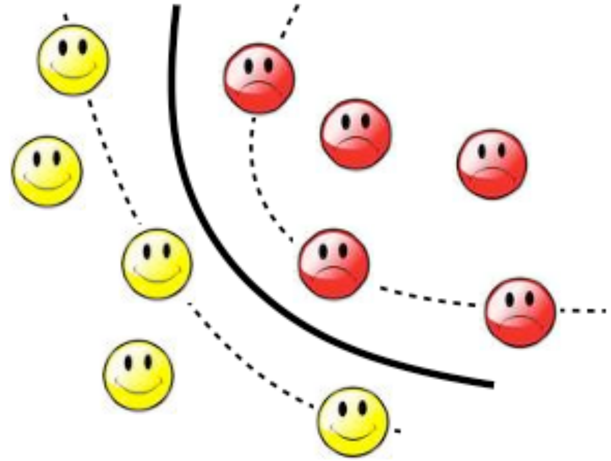
$$p(\mathbf{x}|\mathcal{C}_k) = \prod_{j=1}^{d} \frac{1}{\sqrt{2\pi\sigma_{j,k}^2}} \exp\left(-\frac{1}{2\sigma_{j,k}^2}(x_j - \mu_{j,k})^2 \right)$$



$$\Sigma_k = \begin{pmatrix} \sigma_{1,k}^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_{d,k}^2 \end{pmatrix}$$

Naïve Bayes Summary

- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Independence assumption may not hold for some attributes
  - Use other techniques such as Bayesian Networks (BN)



Evaluating classification

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, scalability, etc.
- Confusion Matrix:

	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)
b: FN (false negative)
c: FP (false positive)
d: TN (true negative)

Metrics for Performance Evaluation

	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

- Most widely-used metric:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Limitation of Accuracy

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10

- If model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %
  - Accuracy is misleading because model does not detect any class 1 example

Cost Matrix

	PREDICTED CLASS	
ACTUAL CLASS		
C(i|j)	**Class=Yes**	**Class=No**
Class=Yes	C(Yes|Yes)	C(No|Yes)
Class=No	C(Yes|No)	C(No|No)

C(i|j): Cost of misclassifying class j example as class i

Computing Cost of Classification

Cost Matrix	PREDICTED CLASS		
	C(i|j)	+	-
ACTUAL CLASS	+	-1	100
	-	1	0

Model M_1	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	150	40
	-	60	250

Model M_2	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	250	45
	-	5	200

Accuracy = 80%

Cost = 3910

Accuracy = 90%

Cost = 4255

Cost vs Accuracy

THE UNIVERSITY OF SYDNEY

Count	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

Cost	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	p	q
	Class=No	q	p

Accuracy is proportional to cost if

1. C(Yes | No) = C(No | Yes) = q
2. C(Yes | Yes) = C(No | No) = p

$N = a + b + c + d$

$\text{Accuracy} = (a + d)/N$

$$\begin{aligned}\text{Cost} &= p\,(a + d) + q\,(b + c) \\ &= p\,(a + d) + q\,(N - a - d) \\ &= q\,N - (q - p)(a + d) \\ &= N\,[q - (q\text{-}p) \times \text{Accuracy}]\end{aligned}$$

Cost-Sensitive Measures

$$\text{Precision (p)} = \frac{a}{a+c}$$

$$\text{Recall (r)} = \frac{a}{a+b}$$

$$\text{F - measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

- Precision is biased towards C(Yes | Yes) & C(Yes | No)
- Recall is biased towards C(Yes | Yes) & C(No | Yes)
- F-measure is biased towards all except C(No | No)

$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

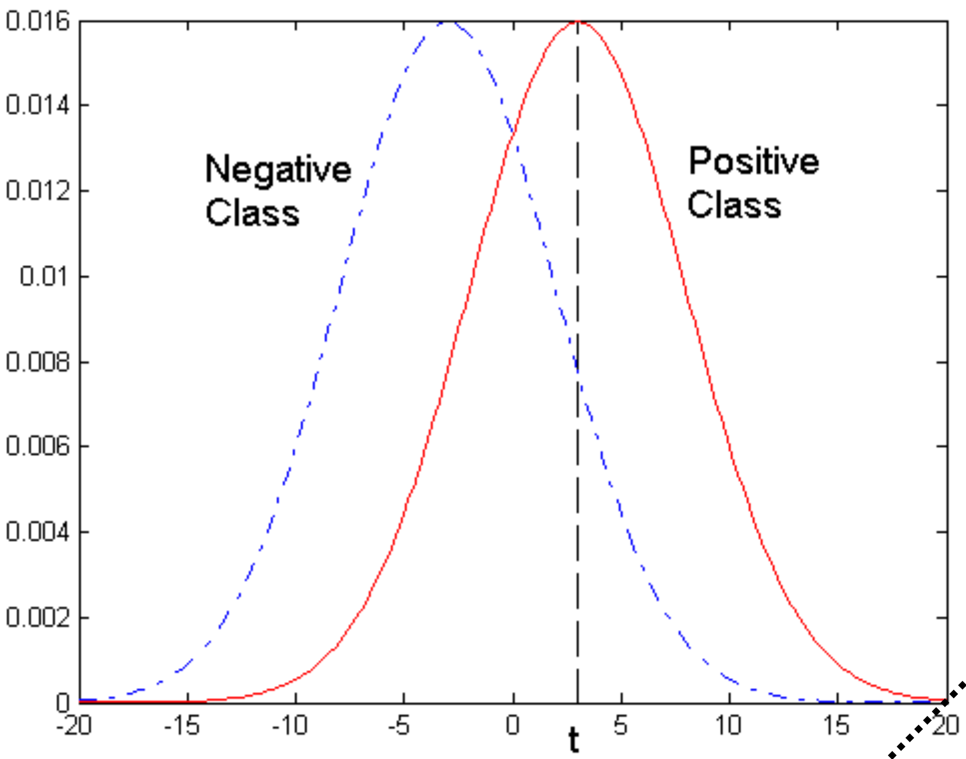
ROC (Receiver Operating Characteristic)

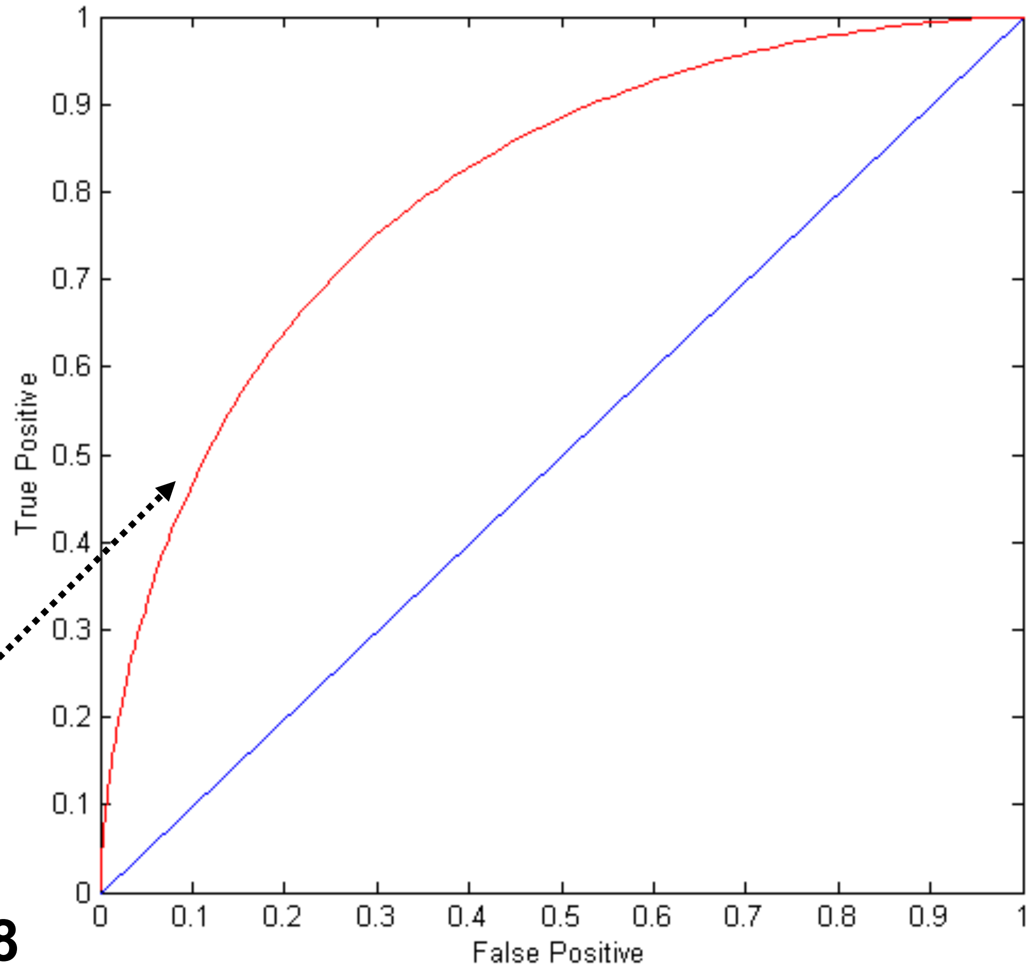
THE UNIVERSITY OF SYDNEY

- Developed in 1950s for signal detection theory to analyse noisy signals
  - Characterise the trade-off between positive hits and false alarms
- ROC curve plots TP (on the y-axis) against FP (on the x-axis)
- Performance of each classifier represented as a point on the ROC curve
- changing the threshold of algorithm, sample distribution or cost matrix changes the location of the point

ROC Curve

- 1-dimensional data set containing 2 classes (positive and negative)
- any points located at x > t is classified as positive





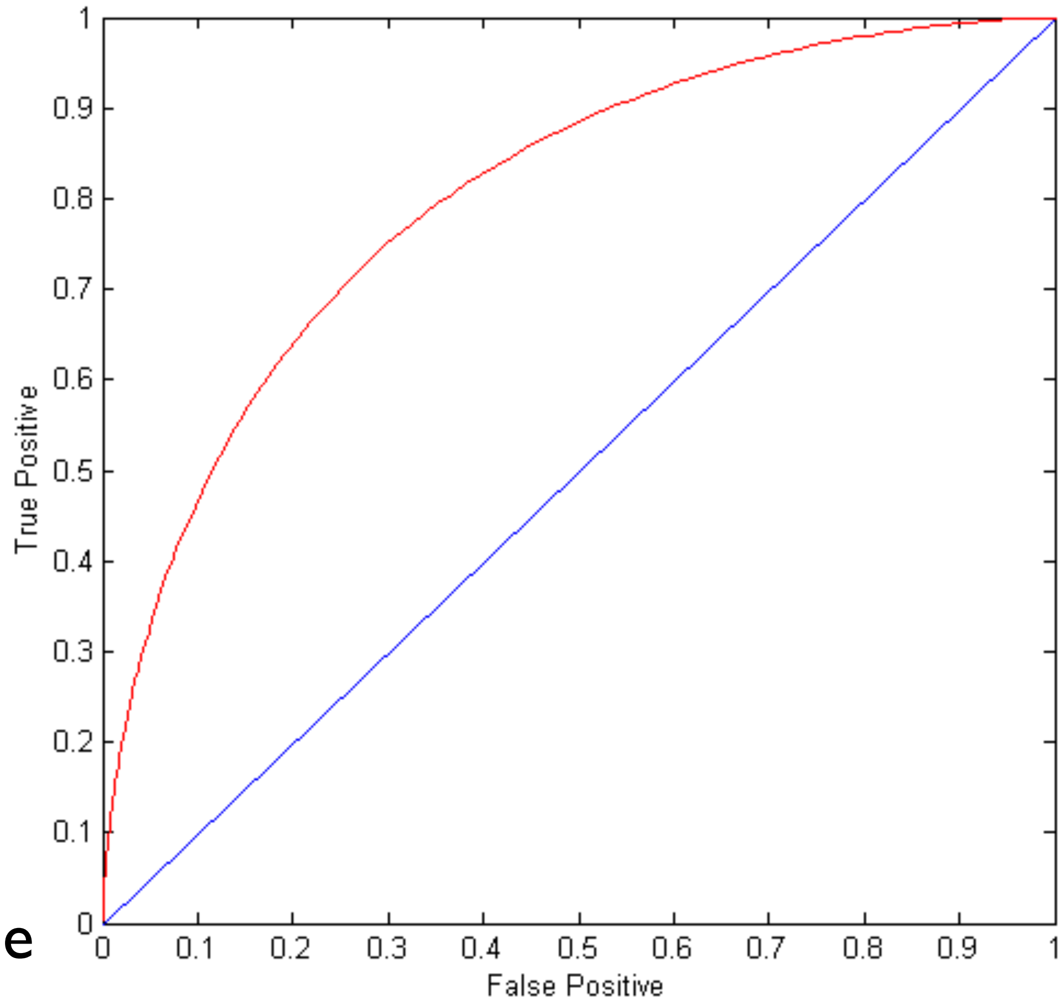
At threshold t:

TP=0.5, FN=0.5, FP=0.12, FN=0.88

ROC Curve

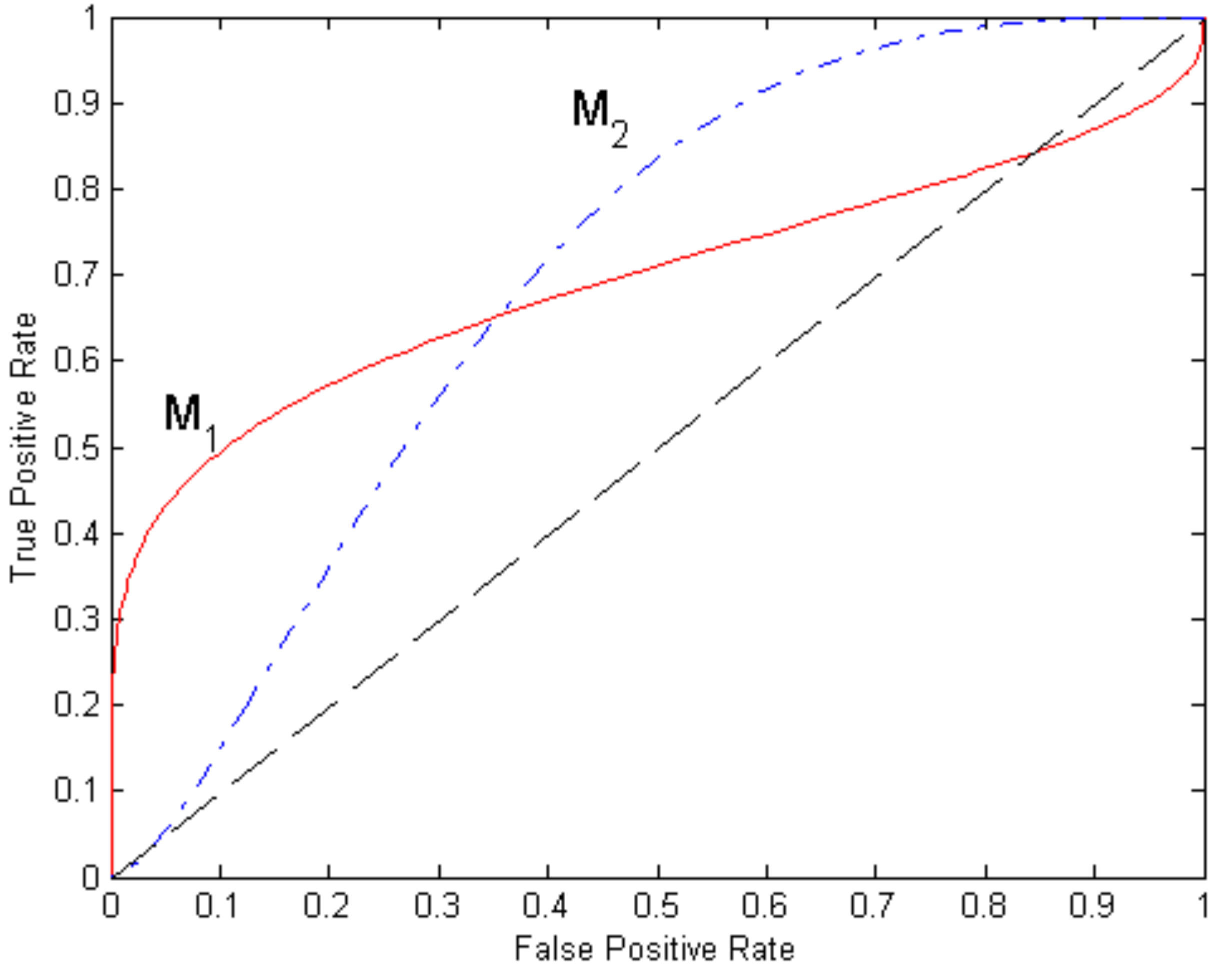
(TP,FP):

- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (1,0): ideal



- Diagonal line:
  - Random guessing
  - Below diagonal line:
    - prediction is opposite of the true class

Using ROC for Model Comparison



- No model consistently outperform the other
- M_1 is better for small FPR
- M_2 is better for large FPR

- Area Under the ROC curve
  - Ideal:
    - Area = 1
  - Random guess:
    - Area = 0.5

How to construct a ROC curve

THE UNIVERSITY OF SYDNEY

Instance	$P(+\|A)$	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Use classifier that produces posterior probability for each test instance $P(+ \mid A)$
- Sort the instances according to $P(+ \mid A)$ in decreasing order
- Apply threshold at each unique value of $P(+ \mid A)$
- Count the number of TP, FP, TN, FN at each threshold
- TP rate, TPR = TP/(TP+FN)
- FP rate, FPR = FP/(FP + TN)

How to construct a ROC curve

THE UNIVERSITY OF SYDNEY

Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve: