Lancaster University

SCC361: Artificial Intelligence

Week 3: Clustering and Classification

Dr Bryan M. Williams
School of Computing and Communications, Lancaster University
Office: InfoLab21 C46 Email: b.williams6@lancaster.ac.uk

Attendance Check-in

Be sure to check in to all timetabled sessions using Attendance Check-in

To check in:

- Check the **Attendance Hub** in iLancaster
- Click **Check In**
- Wait for the “You are checked in” confirmation page
- Here is a the demo

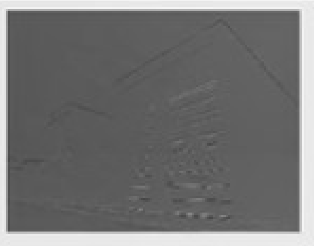
Please DO NOT leave a timetabled session without your attendance being registered

Lancaster University

Last Week: Feature Extraction

Sobel Filter

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$



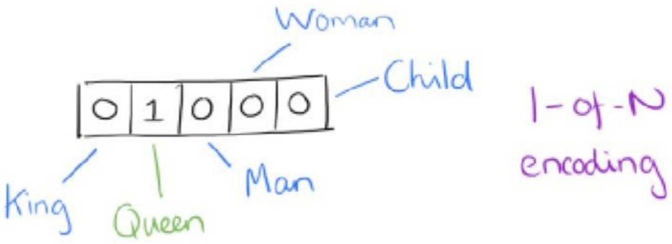
$$\mathbf{G} = \sqrt{\mathbf{G}_x{}^2 + \mathbf{G}_y{}^2}$$

- Term Frequency – Inverse Document Frequency
  - For a corpus of documents D:
    - *Term Frequency* (TF)
      - Frequency counts (log transformed)
      - $TF_{t,d} = \begin{cases} 1 + \log_{10} c(t,d) & if\ c(t,d)>0 \\ 0 & otherwise \end{cases}$
    - *Inverse Document Frequency* (IDF)
      - $|D|$ = # all documents
      - $|\{d \in D : t \in d\}|$ = # documents with t
      - $IDF_t = \log\left(\frac{|D|}{|\{d \in D: t \in d\}|}\right)$
      - Words like 'the' or '*of*' have low IDF
    - **TF-IDF:** $TF \times IDF$

	Document 1	Document 2	Document 3	Document 4	Document 5	Document 6	Document 7	Document 8
Term(s) 1	10	0	1	0	0	0	0	2
Term(s) 2	0	2	0	0	0	18	0	2
Term(s) 3	0	0	0	0	0	0	0	2
Term(s) 4	6	0	0	4	6	0	0	0
Term(s) 5	0	0	0	0	0	0	0	2
Term(s) 6	0	0	1	0	0	1	0	0
Term(s) 7	0	1	8	0	0	0	0	0
Term(s) 8	0	0	0	0	0	3	0	0

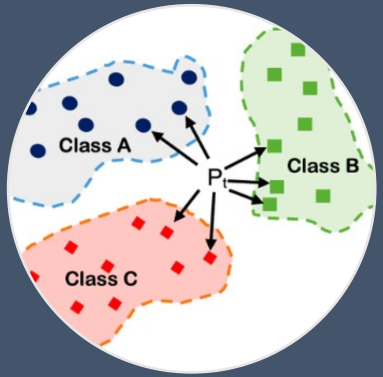
Word Vector (Passage Vector)

Document Vector

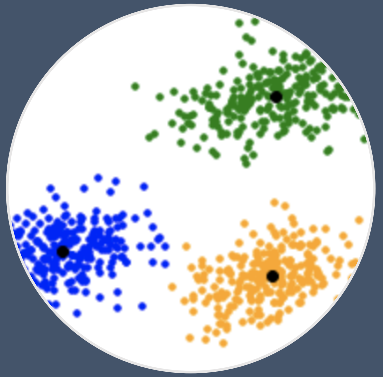


word	features				
king	1	0	0	0	0
queen	0	1	0	0	0
man	0	0	1	0	0
woman	0	0	0	1	0
child	0	0	0	0	1

Next Four Weeks: Fundamental ML

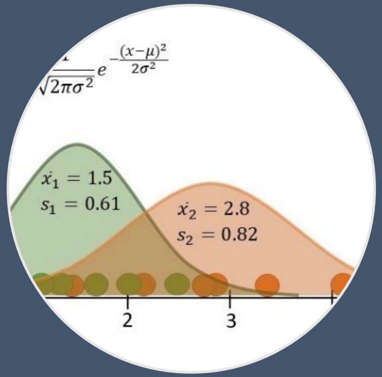


KNN



K-Means

Decision Trees



Naïve Bayes

Generic Algorithms

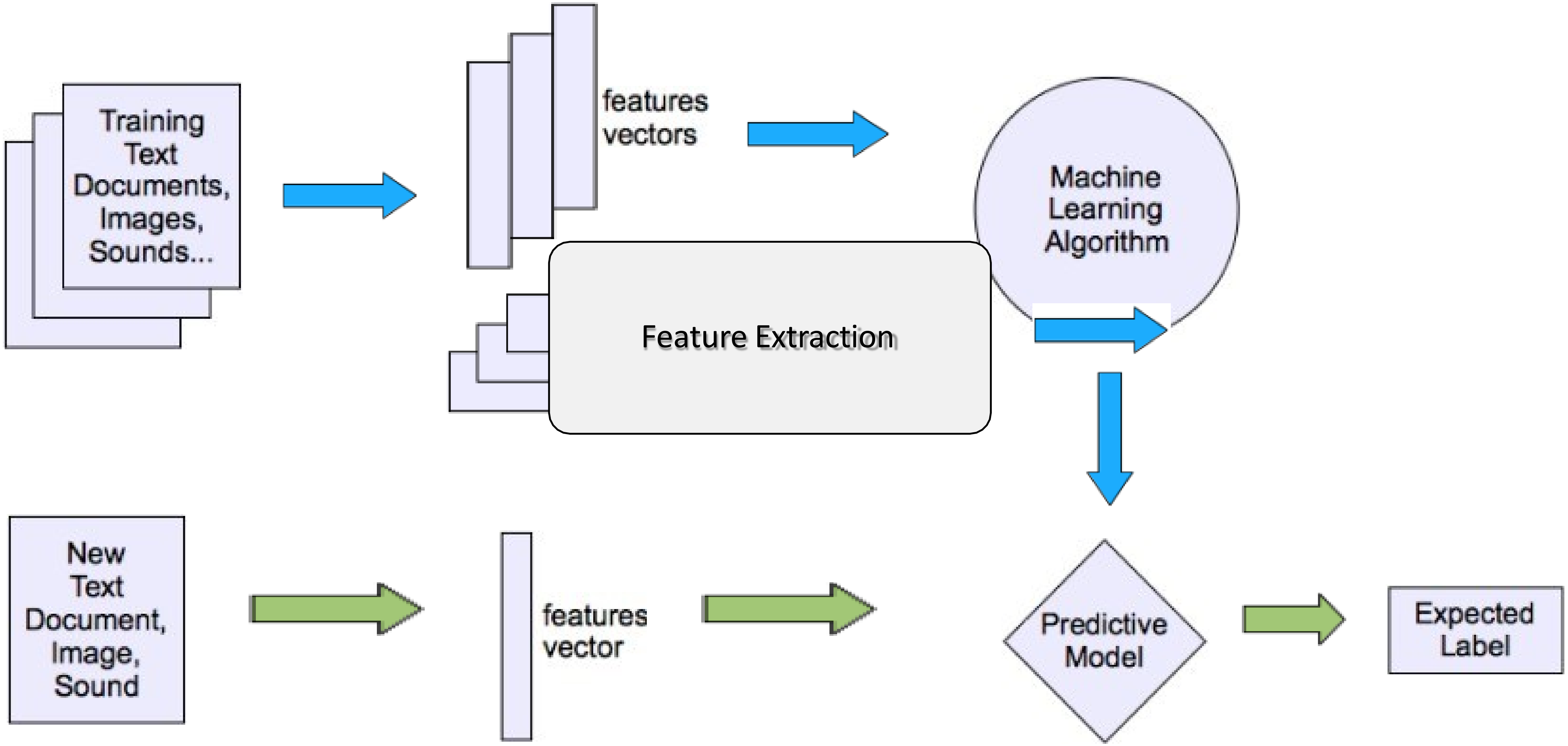
Expected Learning Outcomes

By the end of this week:

- Distinguish between classification and clustering as supervised and unsupervised learning methods
- Demonstrate how clustering algorithms work
- Demonstrate how classification algorithms work
- Understand their typical application scenarios

Machine Learning

Machine Learning Paradigm



Machine Learning Paradigm

Training Data
The observed set of example data

Test Data
Previously unseen data generated by the same process as the training data

Training process
Inferring or learning something about the process that generated the training data

Prediction
Using the inference to make predictions about the test data

Evaluation
Comparing the prediction outputs with the correct test data outputs

Machine Learning Paradigm

Training Data
The observed set of example data

Validation Data
Data tested during the training process

Test Data
Previously unseen data generated by the same process as the training data

Training process
Inferring or learning something about the process that generated the training data

Prediction
Using the inference to make predictions about the test data

Evaluation
Comparing the prediction outputs with the correct test data outputs

Supervised vs Unsupervised

Supervised learning

- Given a set of feature-label pairs, find a rule that predicts the label associated with a previously unseen input
  - Classification
  - Regression

Unsupervised learning

- Given a set of feature vectors (without labels) group them into some "natural clusters"
  - Clustering
  - Association

Classification

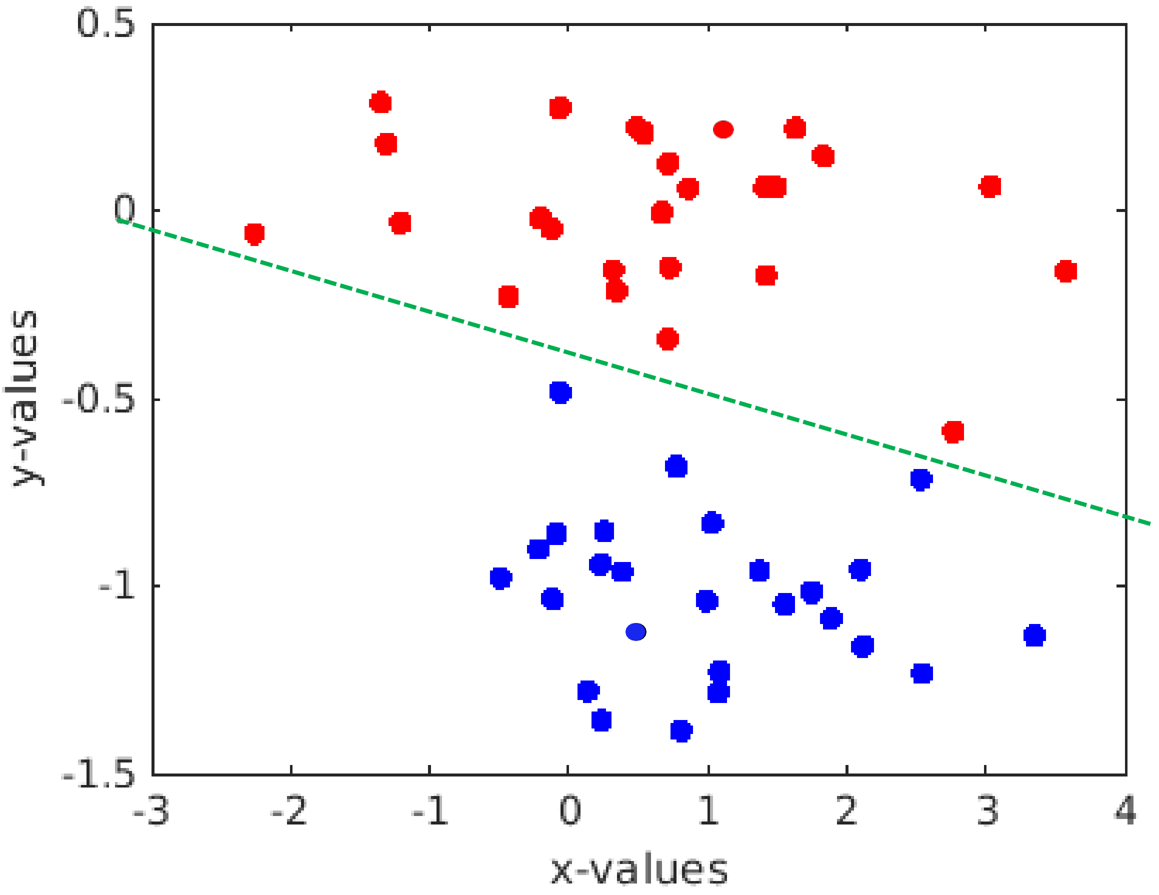
Classification

A supervised learning concept used in building machine learning models that categorise data items into classes

Classifier is trained to specify which of k categories some input belongs to

A classifier is a function:

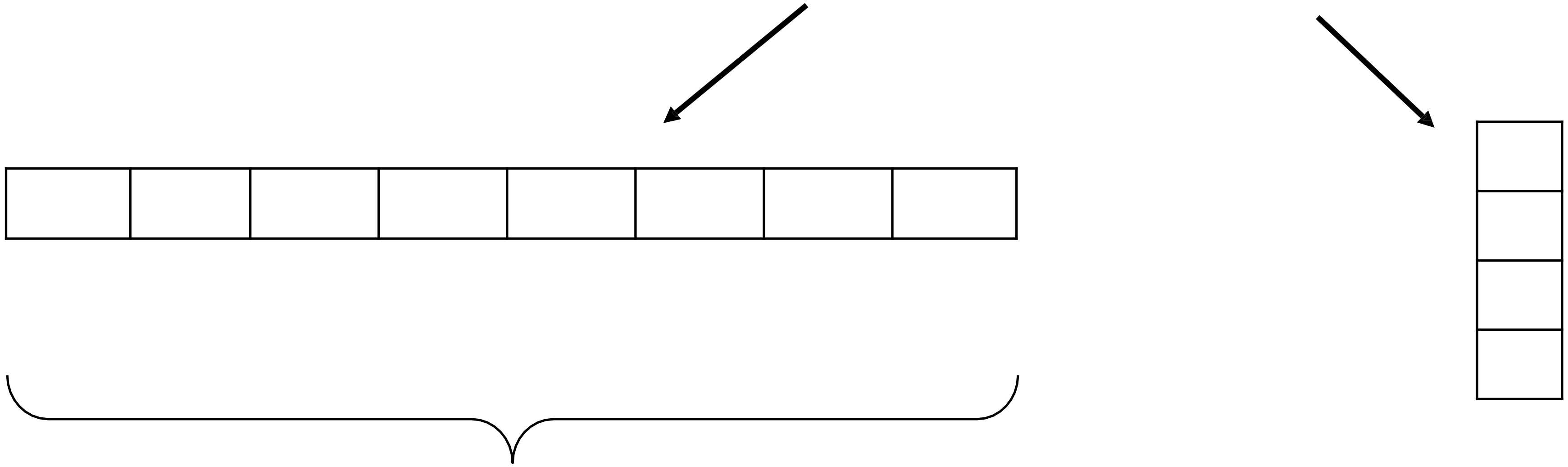
$$f: \mathbb{R}^n \rightarrow \{1, \ldots, c\}$$



Classification

Aim: find f such that $f(x) = y \in \{1, \dots, c\}$ where
x = feature vector and
y = class label

$$f: \mathbb{R}^n \to \{1, \dots, c\}$$

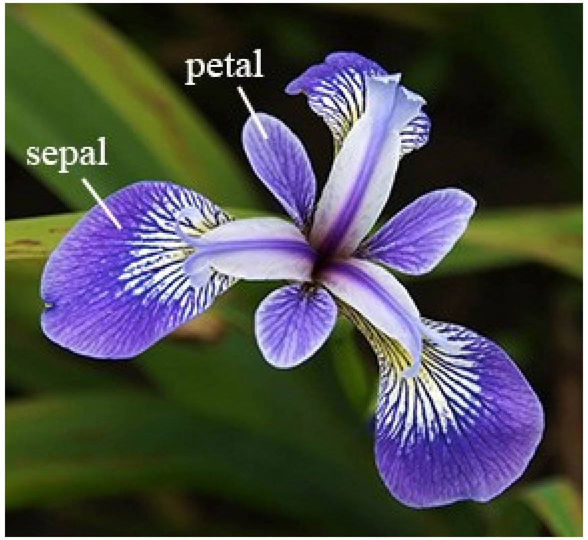


Feature vectors

Classes, labels, categories

Iris Dataset Example

Aim: find f such that $f(x) = y \in \{1, \ldots, c\}$ where
x = feature vector: values of measurements
y = class label: names of species



$$f: \mathbb{R}^n \to \{1, \ldots, c\}$$

$$c = 3$$

$$n = 150 \times 4$$

Sepal Length	Sepal Width	Petal Length	Petal Width

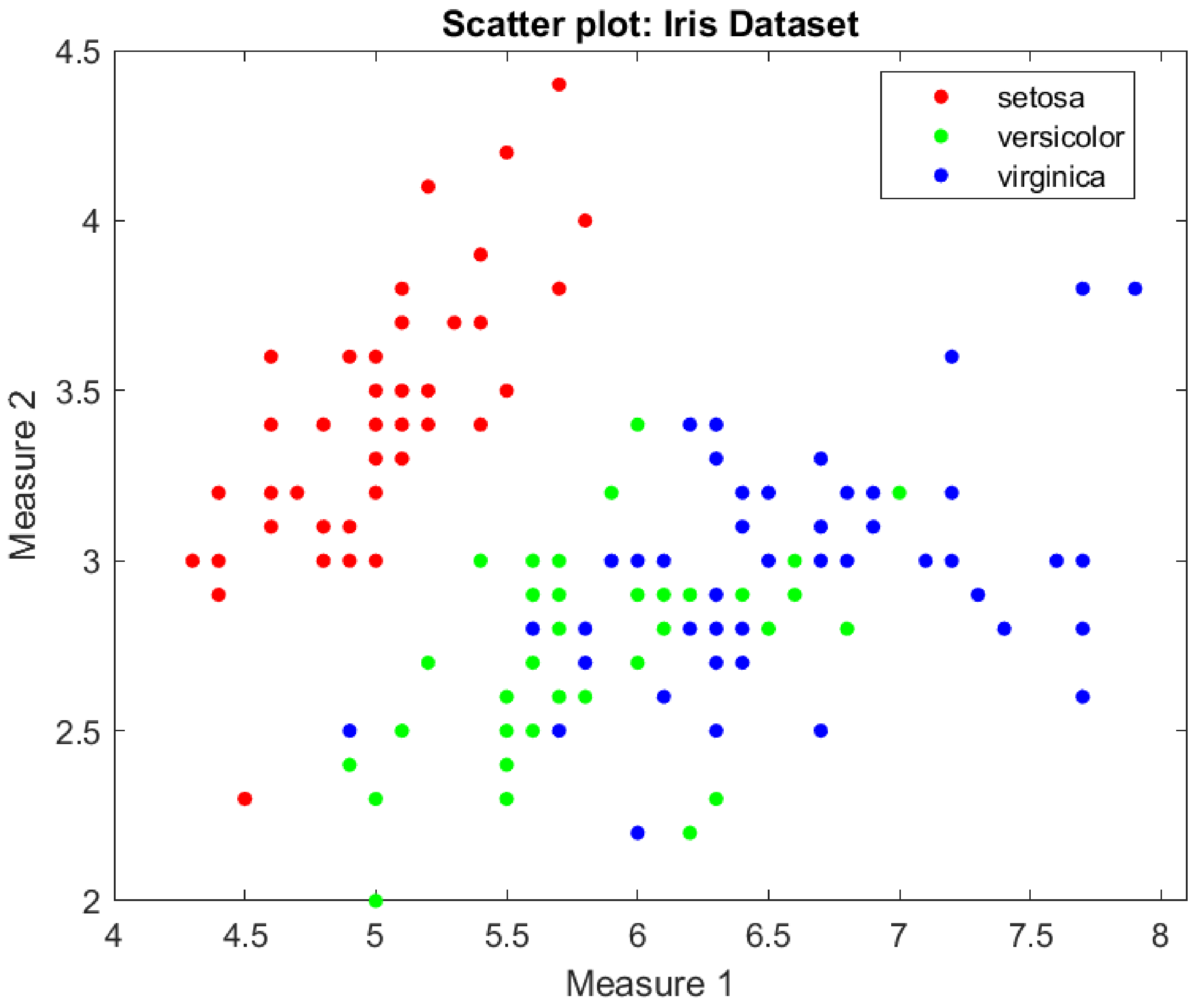
Feature vectors

setosa
versicolor
virginica

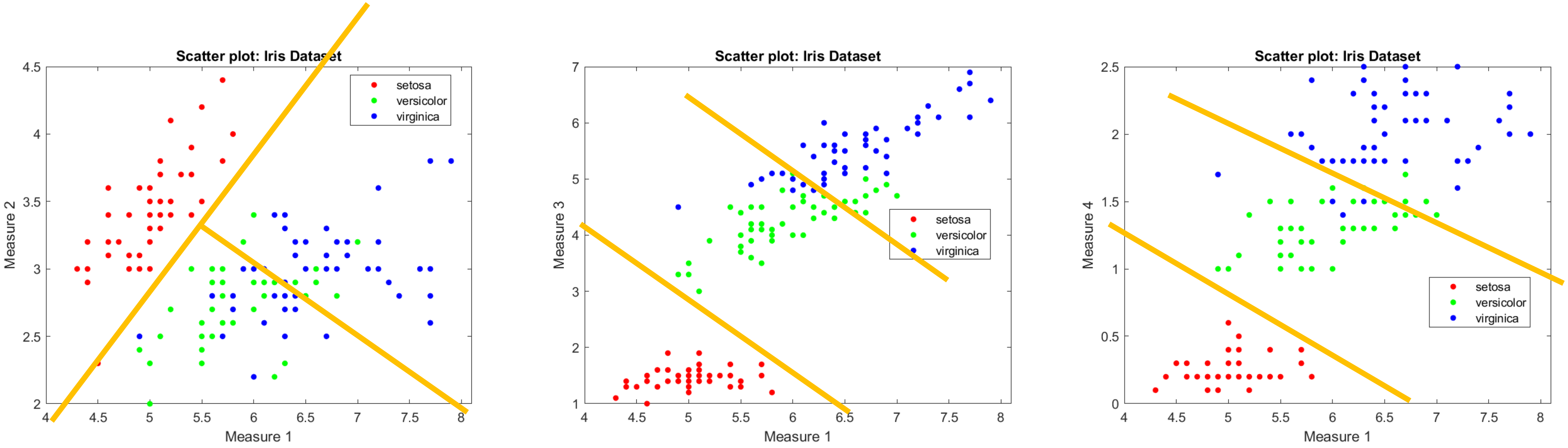
Classes, labels, categories

Iris Dataset Example

Measure1	Measure2	Measure3	Measure4	Class1	Class2
5.1	3.5	1.4	0.2	1	setosa
4.9	3	1.4	0.2	1	setosa
4.7	3.2	1.3	0.2	1	setosa
4.6	3.1	1.5	0.2	1	setosa
5	3.6	1.4	0.2	1	setosa
5.4	3.9	1.7	0.4	1	setosa
4.6	3.4	1.4	0.3	1	setosa
5	3.4	1.5	0.2	1	setosa
4.4	2.9	1.4	0.2	1	setosa
4.9	3.1	1.5	0.1	1	setosa
5.4	3.7	1.5	0.2	1	setosa
4.8	3.4	1.6	0.2	1	setosa
4.8	3	1.4	0.1	1	setosa
4.3	3	1.1	0.1	1	setosa
5.8	4	1.2	0.2	1	setosa
5.7	4.4	1.5	0.4	1	setosa
5.4	3.9	1.3	0.4	1	setosa
5.1	3.5	1.4	0.3	1	setosa
5.7	3.8	1.7	0.3	1	setosa
5.1	3.8	1.5	0.3	1	setosa
5.4	3.4	1.7	0.2	1	setosa
5.1	3.7	1.5	0.4	1	setosa
4.6	3.6	1	0.2	1	setosa
5.1	3.3	1.7	0.5	1	setosa
4.8	3.4	1.9	0.2	1	setosa
7	3.2	4.7	1.4	2	versicolor
6.4	3.2	4.5	1.5	2	versicolor
6.9	3.1	4.9	1.5	2	versicolor
5.5	2.3	4	1.3	2	versicolor



Iris Dataset Example

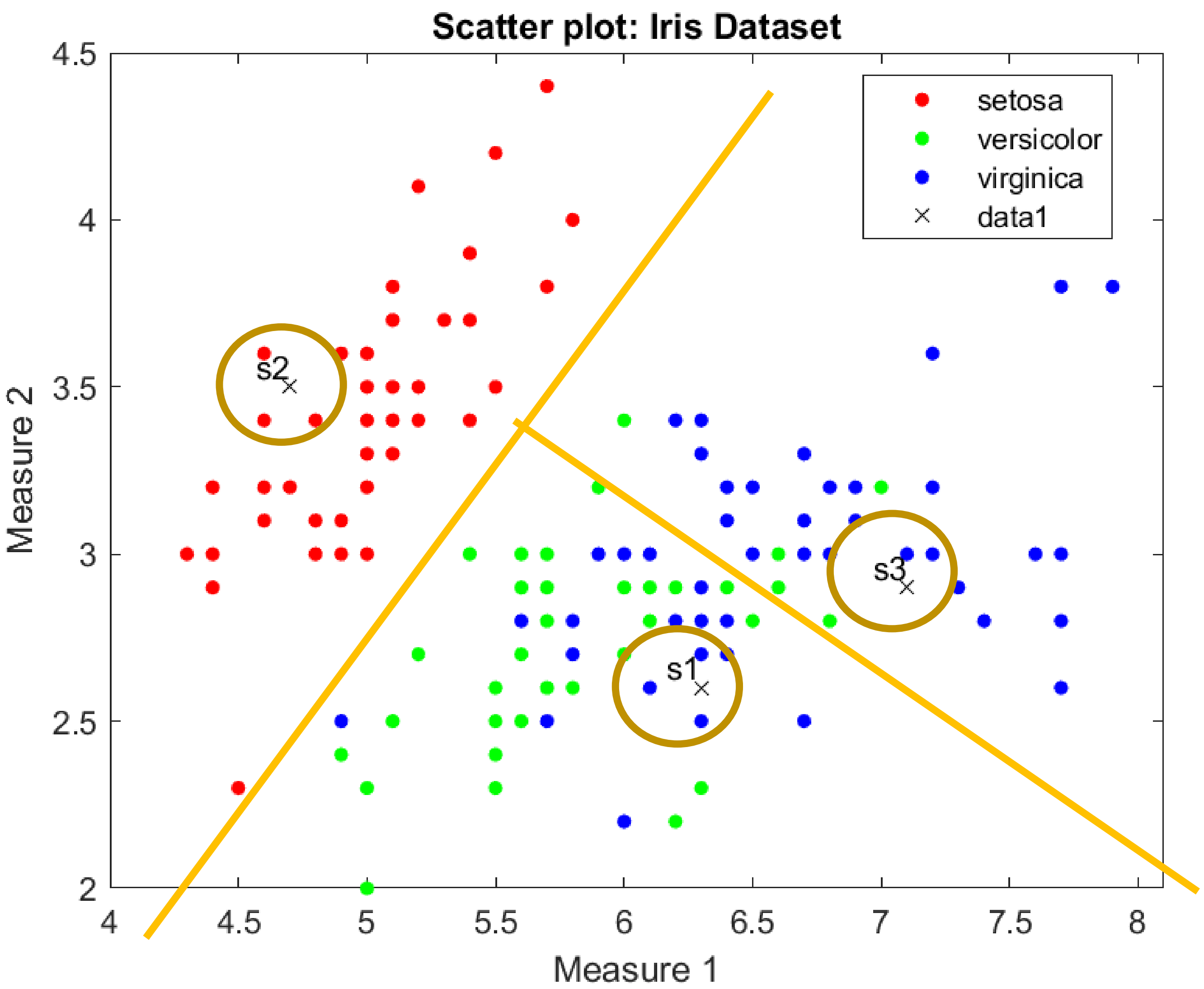


Iris Dataset Example

2.5: Classification

Add the following samples to the graphs created in Section 2.2 and use this to estimate the class that they belong to.

	Measure 1	Measure 2	Measure 3	Measure 4
Sample 1	6.3	2.6	4.1	1.2
Sample 2	4.7	3.5	1.5	0.3
Sample 3	7.1	2.9	5.5	2.1



k-Nearest Neighbour

K-Nearest Neighbour

Car

???

Bicycle

K-Nearest Neighbour

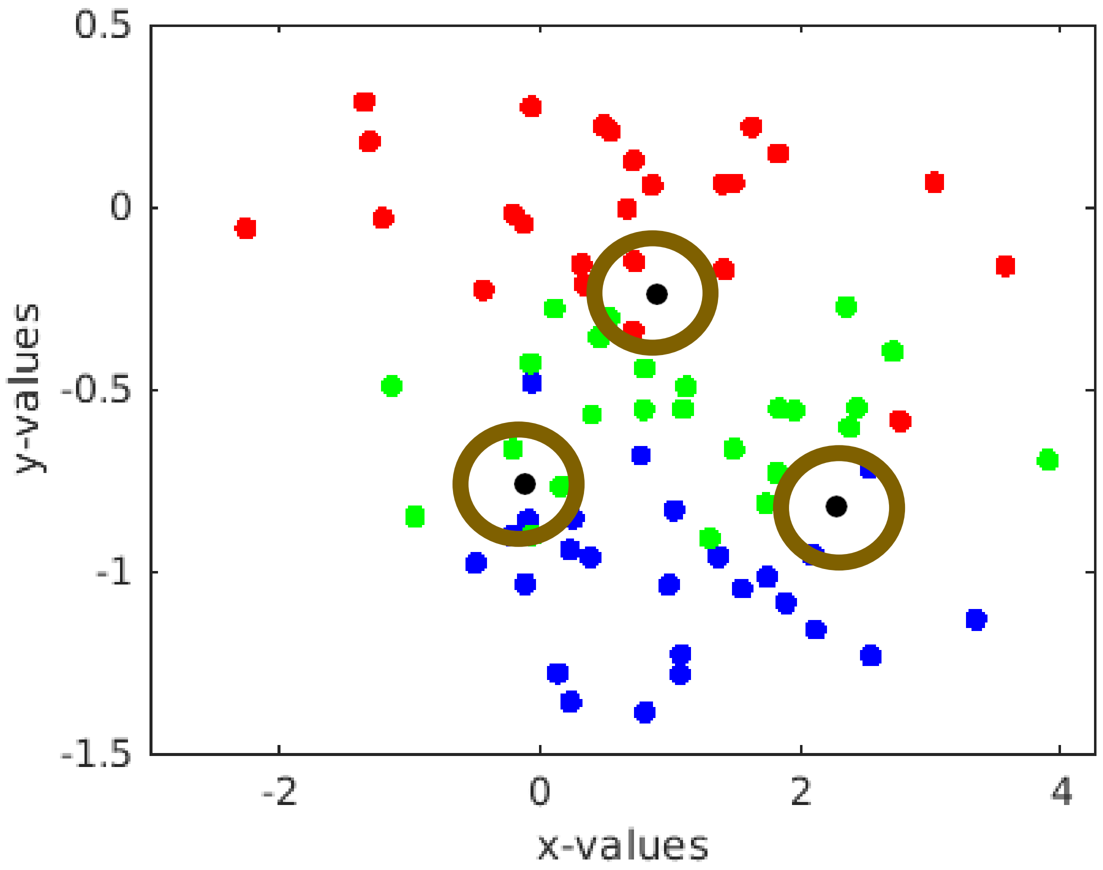
Given **training data** and **a test point**

Look at the k most similar examples

Assign the majority class label $k =$ number of nearest neighbours to search for

Special case: $k = 1$

- 1 nearest neighbour

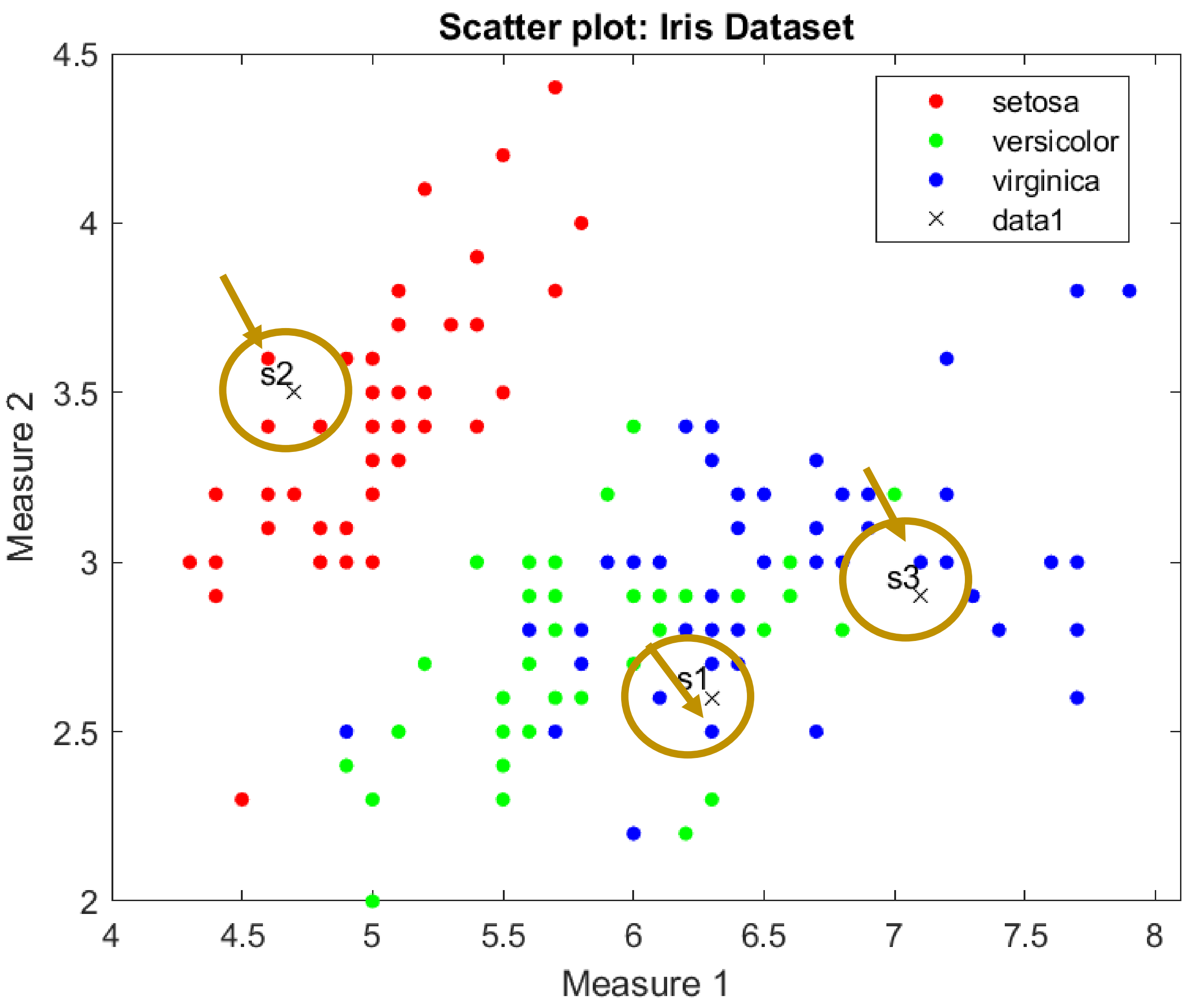


Iris Dataset Example

2.5: Classification

Add the following samples to the graphs created in Section 2.2 and use this to estimate the class that they belong to.

	Measure 1	Measure 2	Measure 3	Measure 4
Sample 1	6.3	2.6	4.1	1.2
Sample 2	4.7	3.5	1.5	0.3
Sample 3	7.1	2.9	5.5	2.1

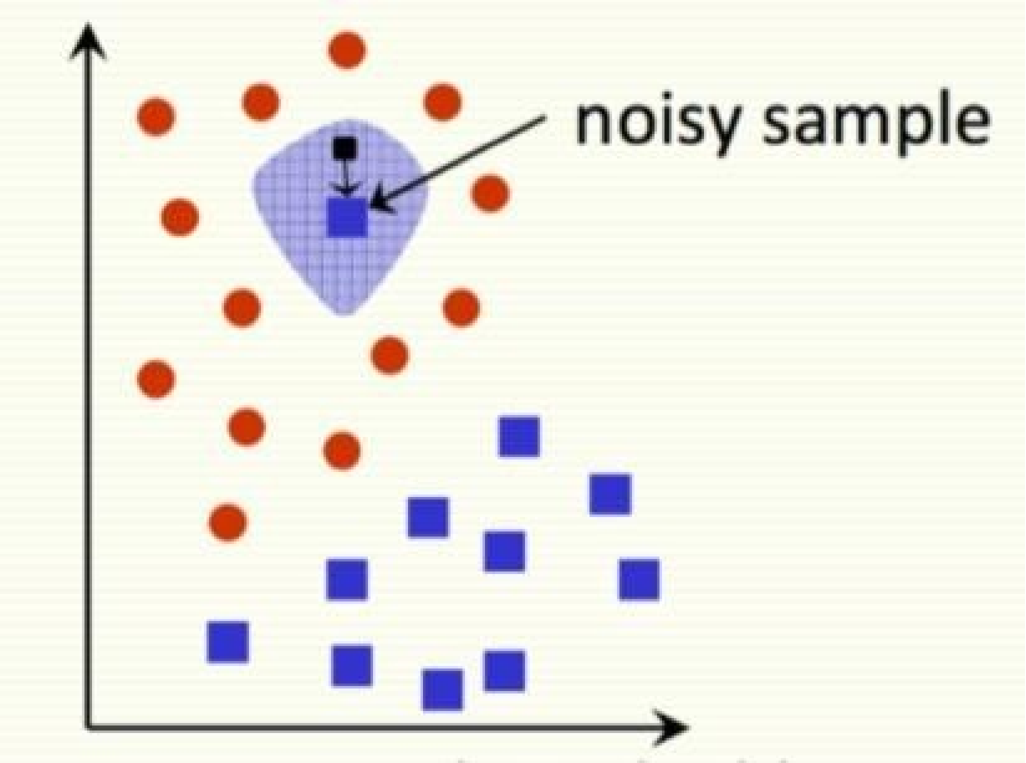


1-Nearest Neighbour

1-NN is sensitive to mis-labelled data

Every example in the blue shaded area will be misclassified as the blue class

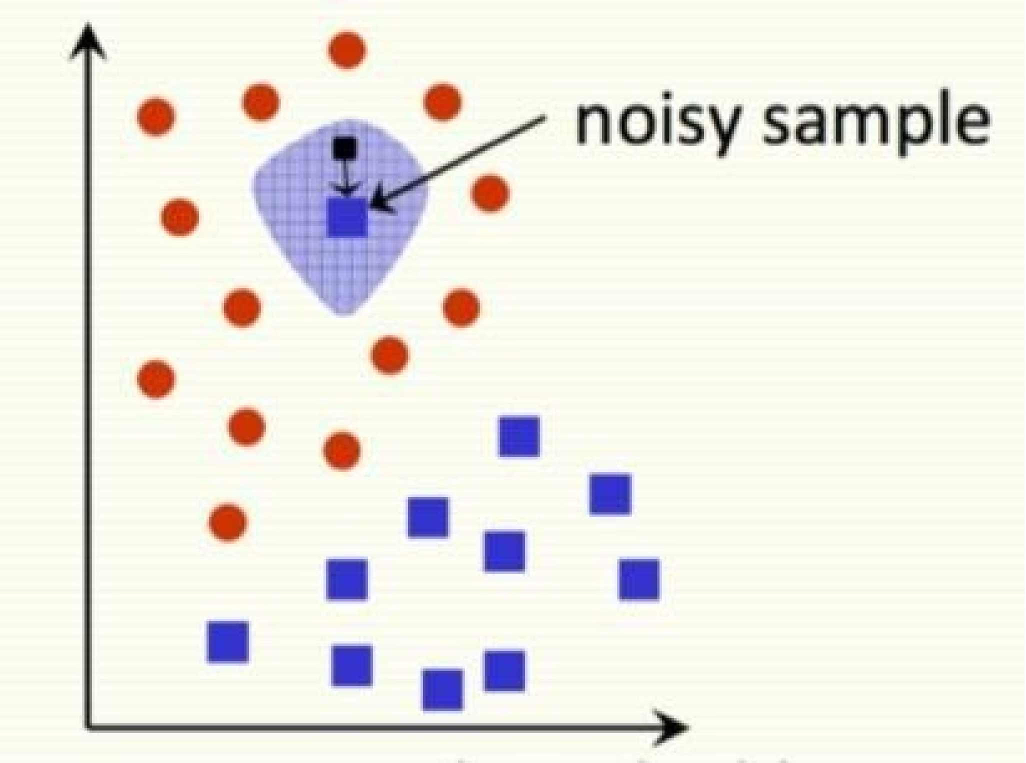
Solution? Increase k



3-Nearest Neighbour

3-NN reduces the classification error

Every example in the blue shades will now be classified correctly as the red class



1-Nearest Neighbour

Iris Example

Using 1-Nearest Neighbour and all of the measures, can we determine the species of the 4th row?

Sepal Length (m_1)	Sepal Width (m_2)	Petal Length (m_3)	Petal Width (m_4)	Species
2.3	4.3	1.2	2.4	setosa
3.4	6.4	4.5	2.3	Versicolor
6.4	4.2	2.1	1.2	virginica
2.5	3.5	2.7	2.1	???

1-Nearest Neighbour

Iris Example

Using 1-Nearest Neighbour and all of the measures, can we determine the species of the 4th row?

Distance: Euclidean

Distance to 4th Row	Sepal Length (m_1)	Sepal Width (m_2)	Petal Length (m_3)	Petal Width (m_4)	Species
1.73	2.3	4.3	1.2	2.4	Setosa
3.54	3.4	6.4	4.5	2.3	Versicolor
4.11	6.4	4.2	2.1	1.2	virginica
0	2.5	3.5	2.7	2.1	???

$$\|a - b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$$

Summation

Let $\mathbf{u} \in \mathbb{R}^n$ and assume $1 \leq a, b \leq n$

i.e. a and b and greater than or equal to 1 and less than or equal to n

The summation of $\mathbf{u}$ from a to b

$$\sum_{x=a}^{b} \mathbf{u}[x] = \mathbf{u}[a] + \mathbf{u}[a+1] + \cdots + \mathbf{u}[b]$$

Can think of this as a *for* loop:

```
count = 0;
for x = a:b
    count = count + u(x);
end
```

or we could form a vector and use a summation

```
count = sum(u(a:b))
```

Summation

Let $\mathbf{u} \in \mathbb{R}^n$ and $A = \{1,2, \dots, n\}$

A is a set of values

The summation of $\mathbf{u}$ over the set A

$$\sum_{x \in A} \mathbf{u}[x] = \mathbf{u}[1] + \mathbf{u}[2] + \cdots + \mathbf{u}[n]$$

Can think of this as a *for* loop:

```
count = 0;
for x = 1:length(A)
    count = count + u(A(x));
end
```

or we could form a vector and use a summation

```
count = sum(u(A))
```

1-Nearest Neighbour

Iris Example

Using 1-Nearest Neighbour and all of the measures, can we determine the species of the 4th row?

Distance: Euclidean

Distance to 4th Row	Sepal Length (m_1)	Sepal Width (m_2)	Petal Length (m_3)	Petal Width (m_4)	Species
1.73	2.3	4.3	1.2	2.4	Setosa
3.54	3.4	6.4	4.5	2.3	Versicolor
4.11	6.4	4.2	2.1	1.2	virginica
0	2.5	3.5	2.7	2.1	???

$$\|a - b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$$

1-Nearest Neighbour

Iris Example

Using 1-Nearest Neighbour and all of the measures, can we determine the species of the 4th row?

Distance: Euclidean

Distance to 4th Row	Sepal Length (m_1)	Sepal Width (m_2)	Petal Length (m_3)	Petal Width (m_4)	Species
1.73	2.3	4.3	1.2	2.4	Setosa
3.54	3.4	6.4	4.5	2.3	Versicolor
4.11	6.4	4.2	2.1	1.2	virginica
0	2.5	3.5	2.7	2.1	???

$$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$$

1-Nearest Neighbour

Iris Example

Using 1-Nearest Neighbour and all of the measures, can we determine the species of the 4th row?

Distance: Euclidean

Distance to 4th Row	Sepal Length (m_1)	Sepal Width (m_2)	Petal Length (m_3)	Petal Width (m_4)	Species
1.73	2.3	4.3	1.2	2.4	Setosa
3.54	3.4	6.4	4.5	2.3	Versicolor
4.11	6.4	4.2	2.1	1.2	virginica
0	2.5	3.5	2.7	2.1	???

$$\|a - b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$$

1-Nearest Neighbour

Iris Example

Using 1-Nearest Neighbour and all of the measures, can we determine the species of the 4th row?

Distance: Euclidean

Distance to 4th Row	Sepal Length (m_1)	Sepal Width (m_2)	Petal Length (m_3)	Petal Width (m_4)	Species
1.73	2.3	4.3	1.2	2.4	Setosa
3.54	3.4	6.4	4.5	2.3	Versicolor
4.11	6.4	4.2	2.1	1.2	virginica
0	2.5	3.5	2.7	2.1	???

$$\|a - b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$$

1-Nearest Neighbour

Iris Example

Using 1-Nearest Neighbour and all of the measures, can we determine the species of the 4th row?

Distance: Euclidean

Distance to 4th Row	Sepal Length (m_1)	Sepal Width (m_2)	Petal Length (m_3)	Petal Width (m_4)	Species
1.73	2.3	4.3	1.2	2.4	Setosa
3.54	3.4	6.4	4.5	2.3	Versicolor
4.11	6.4	4.2	2.1	1.2	virginica
0	2.5	3.5	2.7	2.1	???

$$\|a - b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$$

ED = sqrt(sum((a-b).^2))

1-Nearest Neighbour

Iris Example

Using 1-Nearest Neighbour and all of the measures, can we determine the species of the 4th row?

Distance: Euclidean

What if we only use measures m_3 and m_4

Distance to 4th Row	Sepal Length (m_1)	Sepal Width (m_2)	Petal Length (m_3)	Petal Width (m_4)	Species
1.53	2.3	4.3	1.2	2.4	Setosa
1.81	3.4	6.4	4.5	2.3	Versicolor
1.08	6.4	4.2	2.1	1.2	virginica
0	2.5	3.5	2.7	2.1	???

$$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$$

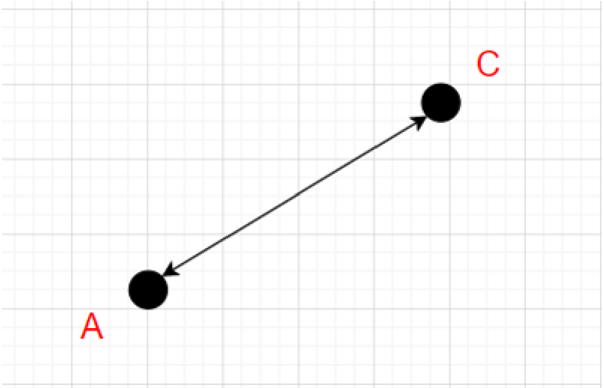
Distance Metrics

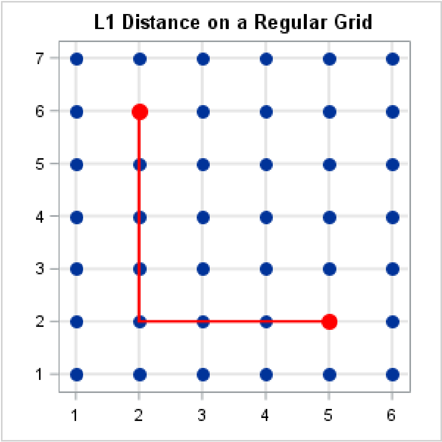
Common Distance Metrics

Name	Formula
Euclidean Distance	$\lVert a-b \rVert_2 = \sqrt{\sum_i (a_i - b_i)^2}$
Square Euclidean Distance	$\lVert a-b \rVert_2^2 = \sum_i (a_i - b_i)^2$
Manhattan Distance	$\lVert a-b \rVert_1 = \sum_i \lvert a_i - b_i \rvert$
Maximum Distance	$\lVert a-b \rVert_\infty = \max_i \lvert a_i - b_i \rvert$
Mahalanobis Distance	$\sqrt{(a-b)^\top S^{-1}(a-b)}$ where S is the Covariance matrix
Cosine Distance	$\frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}$
Hamming Distance	$\sum_i^n \begin{cases} 1 & \text{if } a_i \neq b_i \\ 0 & \text{otherwise} \end{cases}$

Common Distance Metrics

Name	Formula
Euclidean Distance	$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
Square Euclidean Distance	$\|a-b\|_2^2 = \sum_i (a_i - b_i)^2$
Manhattan Distance	$\|a-b\|_1 = \sum_i \lvert a_i - b_i \rvert$
Maximum Distance	$\|a-b\|_\infty = \max_i \lvert a_i - b_i \rvert$

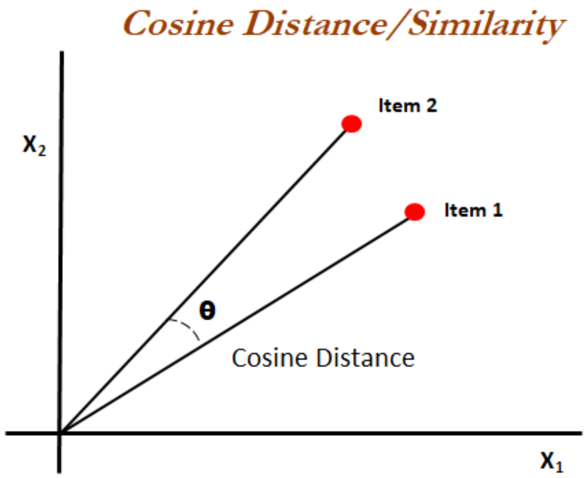


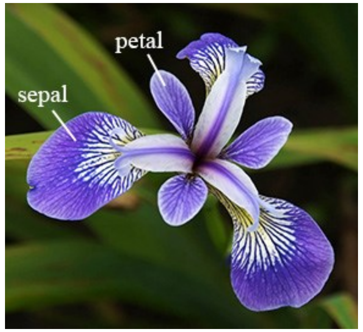


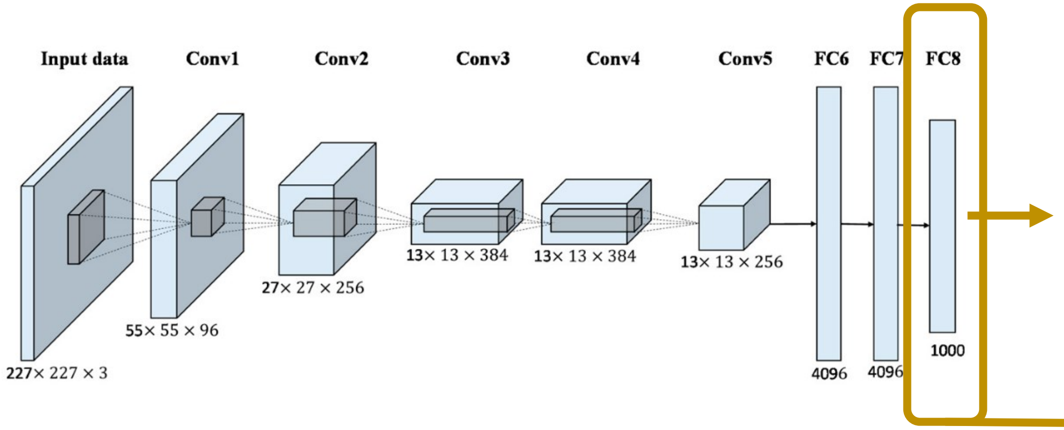
Common Distance Metrics: Cosine Distance

Cosine Distance between a and b

$$\frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}$$







Decision?

Extract Feature Vector

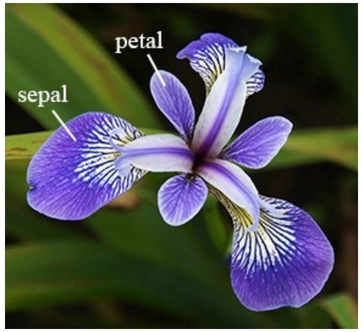
Common Distance Metrics: Bray-Curtis Distance

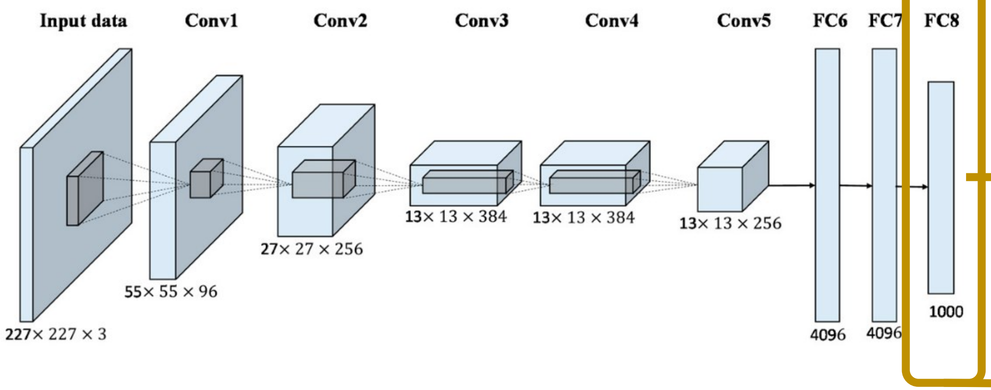
Bray-Curtis Distance between a and b

$$a = \{a_1, a_2, a_3, \dots, a_{|a|}\}$$
$$b = \{b_1, b_2, b_3, \dots, b_{|b|}\}$$

$$1 - 2\frac{\boxed{\sum_i \min(a_i, b_i)}}{|a| + |b|}$$

$$\boxed{\mathrm{sum}\big(\min(a_1, b_1), \min(a_2, b_2), \min(a_3, b_3), \dots, \min\big(a_{|a|}, b_{|b|}\big)\big)}$$





Decision?

Extract Feature Vector

Common Distance Metrics: Hamming Distance

How many letters are different between two words

C	A	T	Total Difference
V	E	T	
1	1	0	2

Hamming Distance = $\sum_{i}^{n} \begin{cases} 1 & \text{if } a_i \neq b_i \\ 0 & \text{otherwise} \end{cases}$

NB: We must have $|a| = |b|$

A	R	T	I	F	I	C	I	A	L			Total Difference
I	N	T	E	L	L	I	G	E	N	C	E	?

Common Distance Metrics: Levenshtein Distance

The number of changes needed to change word a into word b

C	A	T	Number of Changes
V	A	T	1
V	E	T	2

Common Distance Metrics: Levenshtein Distance

The number of changes needed to change word a into word b

A	R	T	I	F	I	C	I	A	L			Number of Changes
I	N	T	E	L	L	I	G	E	N	C	E	?

Levenshtein Distance:

$$\operatorname{lev}(a,b) = \begin{cases} |a| & \text{if } |b| = 0 \\ |b| & \text{if } |a| = 0 \\ \operatorname{lev}\big(\operatorname{tail}(a), \operatorname{tail}(b)\big) & \text{if } \operatorname{head}(a) = \operatorname{head}(b) \\ 1 + \min \begin{cases} \operatorname{lev}(\operatorname{tail}(a), b) \\ \operatorname{lev}\big(a, \operatorname{tail}(b)\big) \\ \operatorname{lev}\big(\operatorname{tail}(a), \operatorname{tail}(b)\big) \end{cases} & \text{otherwise} \end{cases}$$

Common Distance Metrics: Levenshtein Distance

The number of changes needed to change word a into word b

C	A	T	Number of Changes
V	A	T	1
V	E	T	2

Levenshtein Distance:

$$\operatorname{lev}(a,b) = \begin{cases} |a| & \text{if } |b| = 0 \\ |b| & \text{if } |a| = 0 \\ \operatorname{lev}\big(\operatorname{tail}(a), \operatorname{tail}(b)\big) & \text{if } \operatorname{head}(a) = \operatorname{head}(b) \\ 1 + \min \begin{cases} \operatorname{lev}(\operatorname{tail}(a), b) \\ \operatorname{lev}\big(a, \operatorname{tail}(b)\big) \\ \operatorname{lev}\big(\operatorname{tail}(a), \operatorname{tail}(b)\big) \end{cases} & \text{otherwise} \end{cases}$$

Common Distance Metrics: Levenshtein Distance

$$
lev(cat, vet) = 1 + \min \begin{cases}
lev(at, vet) = 1 + \min \begin{cases}
lev(t, vet) = 1 + \min \begin{cases}
lev(, vet) = 3 \\
lev(t, et) = 1 + \min \begin{cases} lev(, et) = 2 \\ lev(t, t) = lev(,) = 0 \\ lev(, t) = 1 \end{cases} \\
lev(, et) = 2
\end{cases} \\
lev(at, et) = 1 + \min \begin{cases} lev(t, et) \\ lev(at, t) \\ lev(t, t) \end{cases} \\
lev(t, et) = 1 + \min \begin{cases} lev(, et) \\ lev(t, t) \\ lev(, t) \end{cases}
\end{cases} \\
lev(cat, et) = 1 + \min \begin{cases} lev(at, et) \\ lev(cat, t) \\ lev(at, t) \end{cases} \\
lev(at, et) = 1 + \min \begin{cases} lev(t, et) \\ lev(at, t) \\ lev(t, t) \end{cases}
\end{cases}
$$

Common Distance Metrics: Levenshtein Distance

Let $m = |a|$ and $n = |b|$

Create a $(m+1) \times (n+1)$ matrix d – here, column and row indices start at 0

Set

$$d(i,0) = i \text{ for } i = 0..m$$
$$d(0,j) = j \text{ for } j = 0..n$$

for i from 1 to m

for j from 1 to n

if $a[i] = b[j]$ then $sub = 0$, otherwise $sub = 1$

$$d[i,j] = \min \begin{cases} d[i-1,j]+1 \\ d[i,j-1]+1 \\ d[i-1,j-1]+sub \end{cases}$$

answer $= d[m,n]$

Example: $\text{lev}(vet, cat)$

		V	E	T
	0	1	2	3
C	1	1	2	3
A	2	2	2	3
T	3	3	3	2

Common Distance Metrics: Levenshtein Distance

Example: $\text{lev}(artificial, intelligence)$

		A	R	T	I	F	I	C	I	A	L
	0	1	2	3	4	5	6	7	8	9	10
I	1	1	2	3	3	4	4	5	6	7	8
N	2	2	2	3	4	4	5	5	6	7	8
T	3	3	3	2	3	4	5	6	6	7	8
E	4	4	4	3	3	4	5	6	7	7	8
L	5	5	5	4	4	4	5	6	7	8	7
L	6	6	6	5	5	5	5	6	7	8	8
I	7	7	7	6	5	6	5	6	6	7	8
G	8	8	8	7	6	6	6	6	7	7	8
E	9	9	9	8	7	7	7	7	7	8	8
N	10	10	10	9	8	8	8	8	8	8	9
C	11	11	11	10	9	9	9	9	9	9	9
E	12	12	12	11	10	10	10	10	10	10	10

Lancaster University

SCC361: Artificial Intelligence

Week 3: Clustering and Classification

Dr Bryan M. Williams
School of Computing and Communications, Lancaster University
Office: InfoLab21 C46 Email: b.williams6@lancaster.ac.uk

Be sure to check in to all timetabled sessions using Attendance Check-in

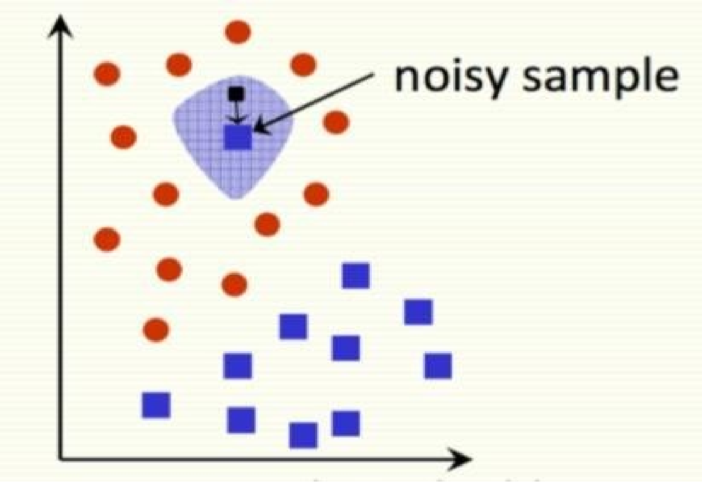
To check in:

- Check the **Attendance Hub** in iLancaster
- Click **Check In**
- Wait for the “You are checked in” confirmation page
- Here is a the demo

Please DO NOT leave a timetabled session without your attendance being registered

KNN and Distance

d	m_1	m_2	m_3	m_4	l
1.73	2.3	4.3	1.2	2.4	1
3.54	3.4	6.4	4.5	2.3	2
4.11	6.4	4.2	2.1	1.2	3
0	2.5	3.5	2.7	2.1	???



Name	Formula
Euclidean Distance	$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
Square Euclidean Distance	$\|a-b\|_2^2 = \sum_i (a_i - b_i)^2$
Manhattan Distance	$\|a-b\|_1 = \sum_i \lvert a_i - b_i \rvert$
Maximum Distance	$\|a-b\|_\infty = \max_i \lvert a_i - b_i \rvert$
Mahalanobis Distance	$\sqrt{(a-b)^\top S^{-1}(a-b)}$ where S is the Covariance matrix
Cosine Distance	$\dfrac{\sum_{i=1}^n a_i b_i}{\sqrt{\sum_{i=1}^n a_i^2}\sqrt{\sum_{i=1}^n b_i^2}}$
Hamming Distance	$\sum_i^n \begin{cases} 1 & \text{if } a_i \neq b_i \\ 0 & \text{otherwise} \end{cases}$

Application of KNN Classification

Image recognition
- Face detection, optical character recognition

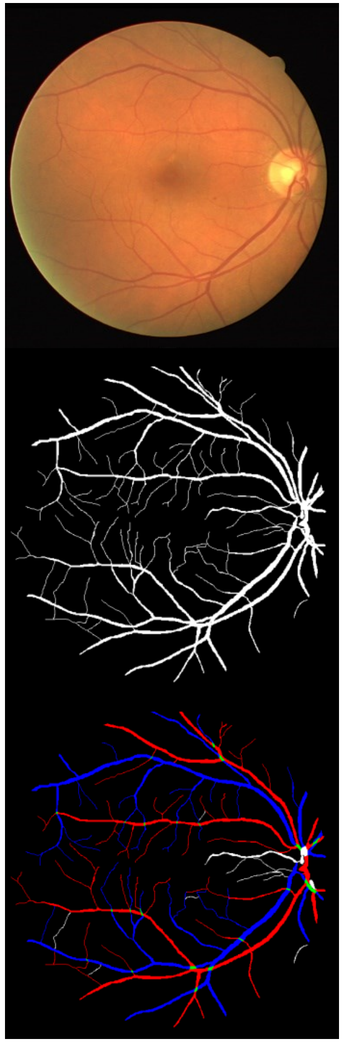
Sentiment analysis
- aka opinion mining e.g. politics, product reviews

Text classification
- e.g. classifying news to topics (technology, sports, entertainment)

Email Classification and Spam filtering
- sorting emails into appropriate folders and removing spams

Authorship attribution
- identifying writing styles of different authors.

sentiment	tweets
MIXED	168
NEGATIVE	1956
NEUTRAL	5472
POSITIVE	854



Measures of Success
and
Error Measures

Machine Learning Paradigm

Training Data
The observed set of example data

Test Data
Previously unseen data generated by the same process as the training data

Training process
Inferring or learning something about the process that generated the training data

Prediction
Using the inference to make predictions about the test data

Evaluation
Comparing the prediction outputs with the correct test data outputs

Evaluation

Suppose we have a method f and a test set $S_x = \{X_1, \ldots, X_n\}$ where $X_i = (\boxed{x_i}, \boxed{l_i})$

label

data

Our method estimates the label as

$$\hat{l_i} = f(x_i)$$

If the model is suitable, we should have

$$\hat{l_i} = l_i \text{ for all } i = 1, \ldots, n$$

$$Accuracy = \frac{Number\ of\ test\ elements\ where\ the\ estimated\ label\ is\ correct}{Number\ of\ test\ elements}$$

Accuracy

Accuracy is a measure of how close your estimate is to the actual value

For classification with categorical labels:

Accuracy = ratio of **correct estimates** to **all estimates**

$$Accuracy = \frac{Number\ of\ test\ elements\ where\ the\ estimated\ label\ is\ correct}{Number\ of\ test\ elements}$$

$$Accuracy = \frac{\sum_i b(l_i, \widehat{l_i})}{|S_x|} \text{ where } b(l_i, \widehat{l_i}) = \begin{cases} 1 \text{ if } l_i = \widehat{l_i} \\ 0 \ if \ l_i \neq \widehat{l_i} \end{cases}$$

Accuracy

Example

x_1:  $l_1 = \text{"dog"}$ $\quad \hat{l}_1 = f(x_1) = \text{"dog"}$ $\quad b(l_1, \hat{l}_1) = 1$

x_2:  $l_2 = \text{"dog"}$ $\quad \hat{l}_2 = f(x_2) = \text{"dog"}$ $\quad b(l_2, \hat{l}_2) = 1$

x_3:  $l_3 = \text{"cat"}$ $\quad \hat{l}_3 = f(x_3) = \text{"cat"}$ $\quad b(l_3, \hat{l}_3) = 1$

x_4:  $l_4 = \text{"cat"}$ $\quad \hat{l}_4 = f(x_4) = \text{"dog"}$ $\quad b(l_4, \hat{l}_4) = 0$

x_5:  $l_5 = \text{"cat"}$ $\quad \hat{l}_5 = f(x_5) = \text{"cat"}$ $\quad b(l_5, \hat{l}_5) = 1$

$$Accuracy = \frac{\sum_i b(l_i, \hat{l}_i)}{|S_x|}$$

$$|S_x| = 5$$

$$\sum_i b(l_i, \hat{l}_i) = 1 + 1 + 1 + 0 + 1$$
$$= 4$$

$$Accuracy = \frac{4}{5} = 0.8 = 80\%$$

Accuracy

Non-categorical

What if model output is not categorical?

Example: determine location of centre of pupil

More meaningful: **distance** such as L^2-norm:

$$Accuracy = \|l - \hat{l}\|_2 = \sqrt{(x-\hat{x})^2 + (y-\hat{y})^2}$$

E.g. $l = (95,101)$, $\hat{l} = (100,103)$

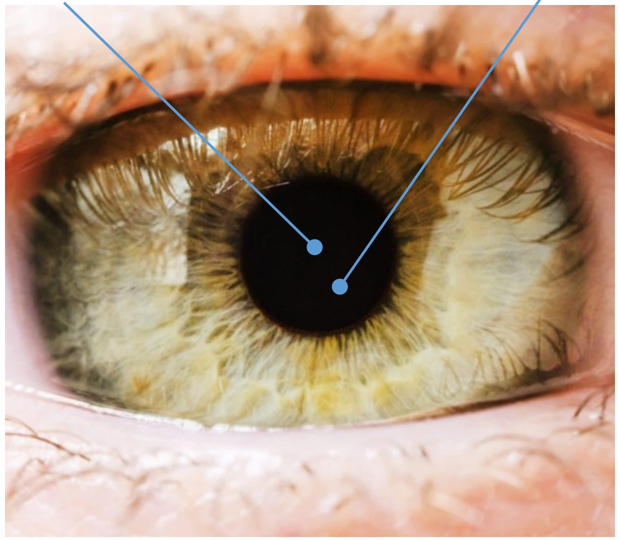
$$Accuracy = \|l - \hat{l}\|_2 = \sqrt{(95-100)^2 + (101-103)^2}$$
$$= \sqrt{(-5)^2 + (-2)^2} = \sqrt{25+4} = \sqrt{29} \approx 5.385$$

Actual Centre

$l = (x, y)$

Predicted Centre

$\hat{l} = (\hat{x}, \hat{y})$



Confusion Matrices

- Accuracy is important as an **overall** view.

Example:

- Suppose we are developing a new automated technique for Glaucoma diagnosis
- Our technique is 95% accurate on a test set of 10,000 people
- It is wrong for 5%
  - But which ones???

Healthy eyes

Periphial vision loss due to glaucoma

Confusion Matrices

- Confusion Matrices give us more insight.
  - **TP**: We predicted **positive** and we were **right**
  - **TN**: We predicted **negative** and we were **right**
  - **FP**: We predicted **positive** and we were **wrong**
  - **FN**: We predicted **negative** and we were **wrong**

		True Class	
		Positive	Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

Examples:

- Positive = Glaucoma, Negative = Healthy
- Positive = Cats, Negative = Dogs

Confusion Matrices

Glaucoma (GL) Diagnosis Example:

- Our technique is 95% accurate on a test set of 10,000 people
- Our test set includes
  - 500 GL Patients
  - 9500 Non-GL Patients
- Scenario 1: Our results:
  - All GL Patients diagnosed correctly
  - 9000 Healthy patients diagnosed correctly
  - 500 Healthy patients mis-diagnosed

		True Class	
Predicted Class		GL	Healthy
	GL	500	500
	Healthy	0	9000

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = 0.95$$

Confusion Matrices

Glaucoma (GL) Diagnosis Example:

- Our technique is 95% accurate on a test set of 10,000 people
- Our test set includes
  - 500 GL Patients
  - 9500 Non-GL Patients
- Scenario 2: Our results:
  - All Healthy Patients diagnosed correctly
  - All DL patients mis-diagnosed

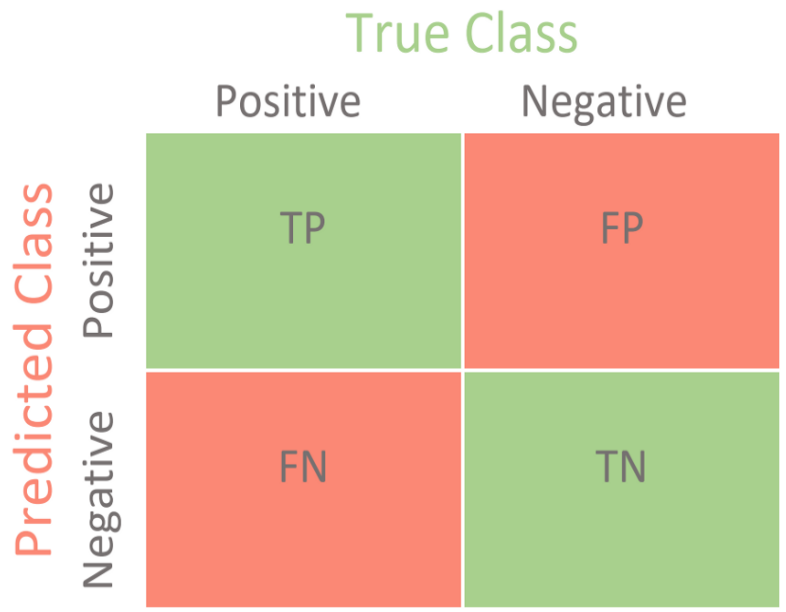
	True Class		
Predicted Class		GL	Healthy
	GL	0	0
	Healthy	500	9500

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = 0.95$$

Confusion Matrices

What other measures can we consider?

- **Sensitivity**, recall, hit rate, true positive rate
- **Specificity**, selectivity, true negative rate
- **Precision**, positive predicted value
- **Negative predictive value**
- Miss rate, false negative rate
- Fall out, false positive rate
- False discovery rate
- False omission rate
- **F1 score**, dice coefficient



Sensitivity and Specificity

Sensitivity: Probability of *positive* outcome if truly positive

$$Sensitivity = \frac{TP}{TP + FN} = \frac{TP}{|Positives|}$$

Specificity: Probability of *negative* outcome if truly *negative*

$$Specificity = \frac{TN}{TN + FP} = \frac{TN}{|Negatives|}$$

		True Class	
		Positive	Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

Sensitivity and Specificity

Example

Scenario 1	True Class		
Predicted Class		GL	Healthy
	GL	500	500
	Healthy	0	9000

$$Sensitivity = \frac{TP}{TP + FN} = \frac{500}{500 + 0} = 1$$

$$Specificity = \frac{TN}{TN + FP} = \frac{9000}{9000 + 500} \approx 0.95$$

Scenario 2	True Class		
Predicted Class		GL	Healthy
	GL	0	0
	Healthy	500	9500

$$Sensitivity = \frac{TP}{TP + FN} = \frac{0}{500 + 0} = 0$$

$$Specificity = \frac{TN}{TN + FP} = \frac{9500}{9500 + 0} = 1$$

Sensitivity and Specificity

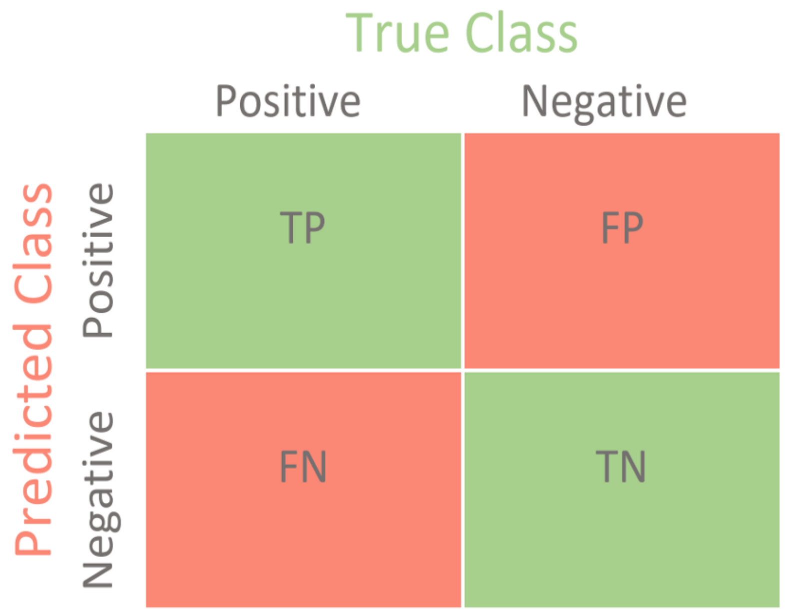
Sensitivity: Probability of *positive* outcome if truly positive

$$Sensitivity = \frac{TP}{TP + FN} = \frac{TP}{|Positives|}$$

Specificity: Probability of *negative* outcome if truly *negative*

$$Specificity = \frac{TN}{TN + FP} = \frac{TN}{|Negatives|}$$

- Usually consider sensitivity and specificity together.
- Aim for high sensitivity and high specificity.
- Can use trade-off thresholds.



Precision and Negative Predicted Value (NPV)

Precision: Proportion of *truly positive* outcomes to *positive predictions*

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{|Predicted\ Positives|}$$

NPV: Proportion of *truly negative* outcomes to *predicted negative*

$$NPV = \frac{TN}{TN + FN} = \frac{TN}{|Predicted\ Negatives|}$$

		True Class	
		Positive	Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

Precision and Negative Predicted Value (NPV)

Lancaster University

Example

Scenario 1	True Class	
Predicted Class	GL	Healthy
GL	500	500
Healthy	0	9000

$$Precision = \frac{TP}{TP + FP} = \frac{500}{500 + 500} = 0.5$$

$$NPV = \frac{TN}{TN + FN} = \frac{9000}{9000 + 0} = 1$$

Scenario 2	True Class	
Predicted Class	GL	Healthy
GL	0	0
Healthy	500	9500

$$Precision = \frac{TP}{TP + FP} = \frac{0}{0 + 0} = undefined$$

$$NPV = \frac{TN}{TN + FN} = \frac{9500}{9500 + 500} = 0.95$$

Precision and Negative Predicted Value (NPV)

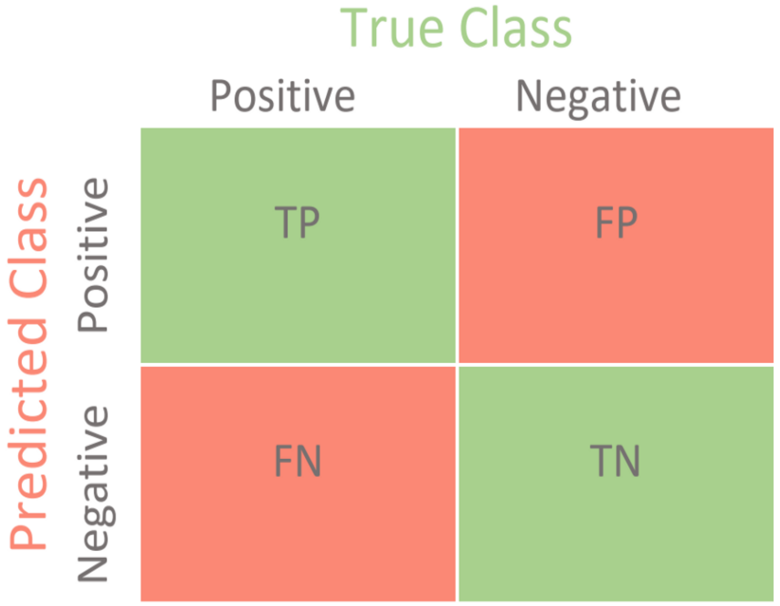
Precision: Proportion of *truly positive* outcomes to *positive predictions*

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{|Predicted\ Positives|}$$

NPV: Proportion of *truly negative* outcomes to *predicted negative*

$$NPV = \frac{TN}{TN + FN} = \frac{TN}{|Predicted\ Negatives|}$$

- Should consider precision and NPV together.
- Aim for high precision and NPV.
- Often consider **precision and recall** (sensitivity) together



F_1 Score

It can be difficult to compare a pair of values

Aim to reduce to a single value

We use harmonic mean when interested in average *rate*

F_1 score is the harmonic mean of precision and recall

$$F_1 = \frac{2}{recall^{-1} + precision^{-1}} = 2\,\frac{precision \cdot recall}{precision + recall}$$

$$= \frac{2TP}{2TP + FP + FN}$$

For our examples:

Example 1: $F_1 = \frac{2 \cdot 500}{2 \cdot 500 + 500 + 0} = \frac{1000}{1500} \approx 0.66$

Example 2: $F_1 = \frac{2 \cdot 0}{2 \cdot 0 + 0 + 500} = 0$

Confusion Matrices

Back to our example

Measure	Scenario 1	Scenario 2
TP	500	0
TN	9000	9500
FP	500	0
FN	0	500
Accuracy	0.95	0.95
Sensitivity	1	0
Specificity	0.947	1
Precision	0.5	Undefined
NPV	1	0.95
F1 Score	0.667	0

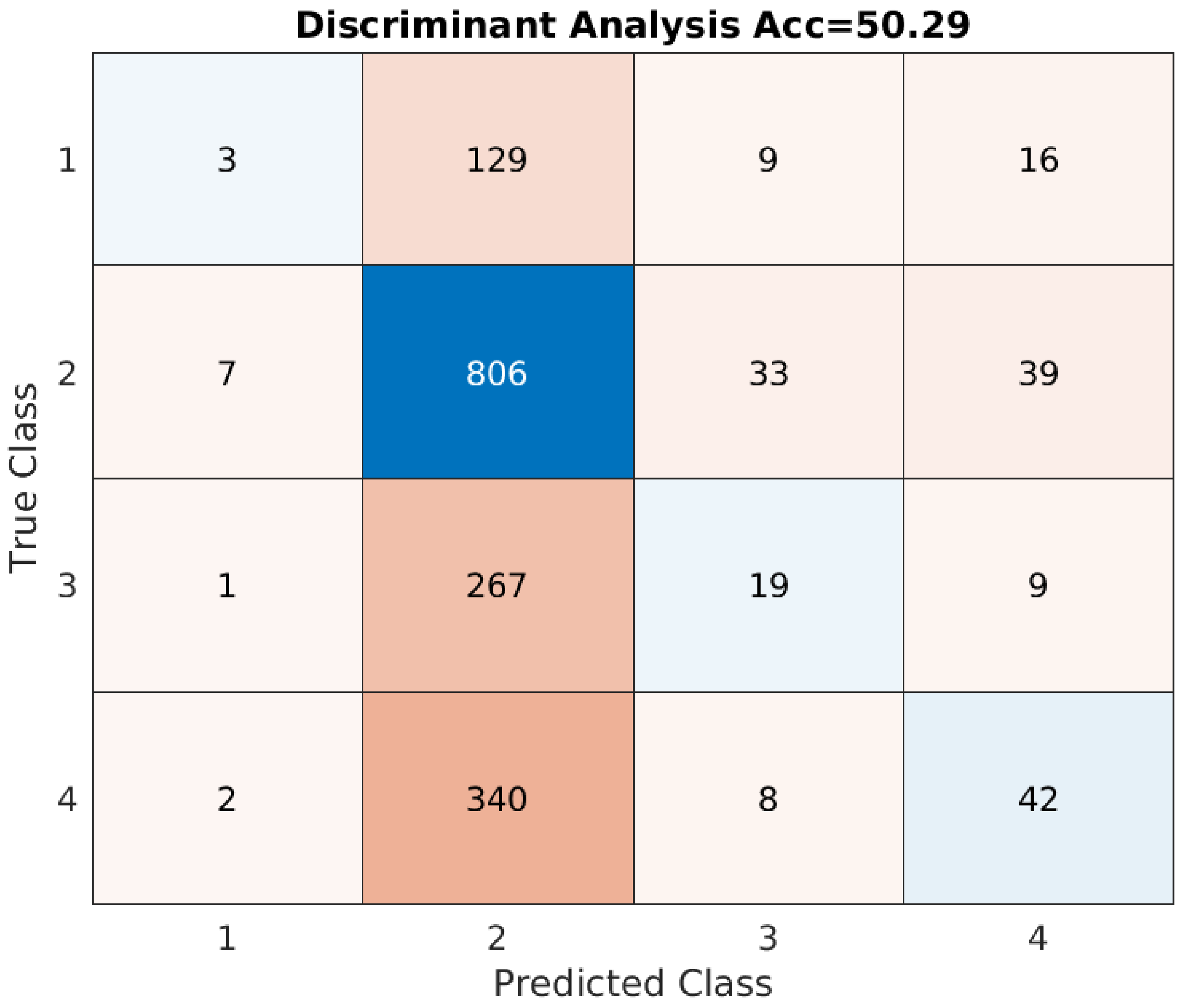
Scenario 1	**True Class**		
Predicted Class		GL	Healthy
	GL	500	500
	Healthy	0	9000

Scenario 2	**True Class**		
Predicted Class		GL	Healthy
	GL	0	0
	Healthy	500	9500

Non-Binary Confusion Matrices

Compute accuracy in the same way.

There exist generalised versions of F1 Score etc but this is not covered here.



Example: Security System

False Acceptance Rate:

$$FAR = \frac{Incorrect\ Authentication}{Total\ Attempts}$$

False Rejection Rate:

$$FRR = \frac{Incorrect\ Rejection}{Total\ Attempts}$$

Aim: minimise both FAR and FRR.

Equal Error Rate: when FAR = FRR

Example: Clinical Test

So how to choose hyperparameters?

Best Accuracy	Best F_1 Score	Best Youden Index	Best Precision
Threshold: 0.67	Threshold: 0.59	Threshold: 0.59	Threshold: 0.83
Accuracy: 0.80	Accuracy: 0.73	Accuracy: 0.73	Accuracy: 0.76
Sensitivity: 0.49	Sensitivity: 0.80	Sensitivity: 0.80	Sensitivity: 0.07
Specificity: 0.90	Specificity: 0.70	Specificity: 0.70	Specificity: 1.00
Precision: 0.64	Precision: 0.48	Precision: 0.48	Precision: 0.92
F_1 Score: 0.55	F_1 Score: 0.60	F_1 Score: 0.60	F_1 Score: 0.13

At 0.95 sensitivity, we have: 0.41 specificity

Clustering

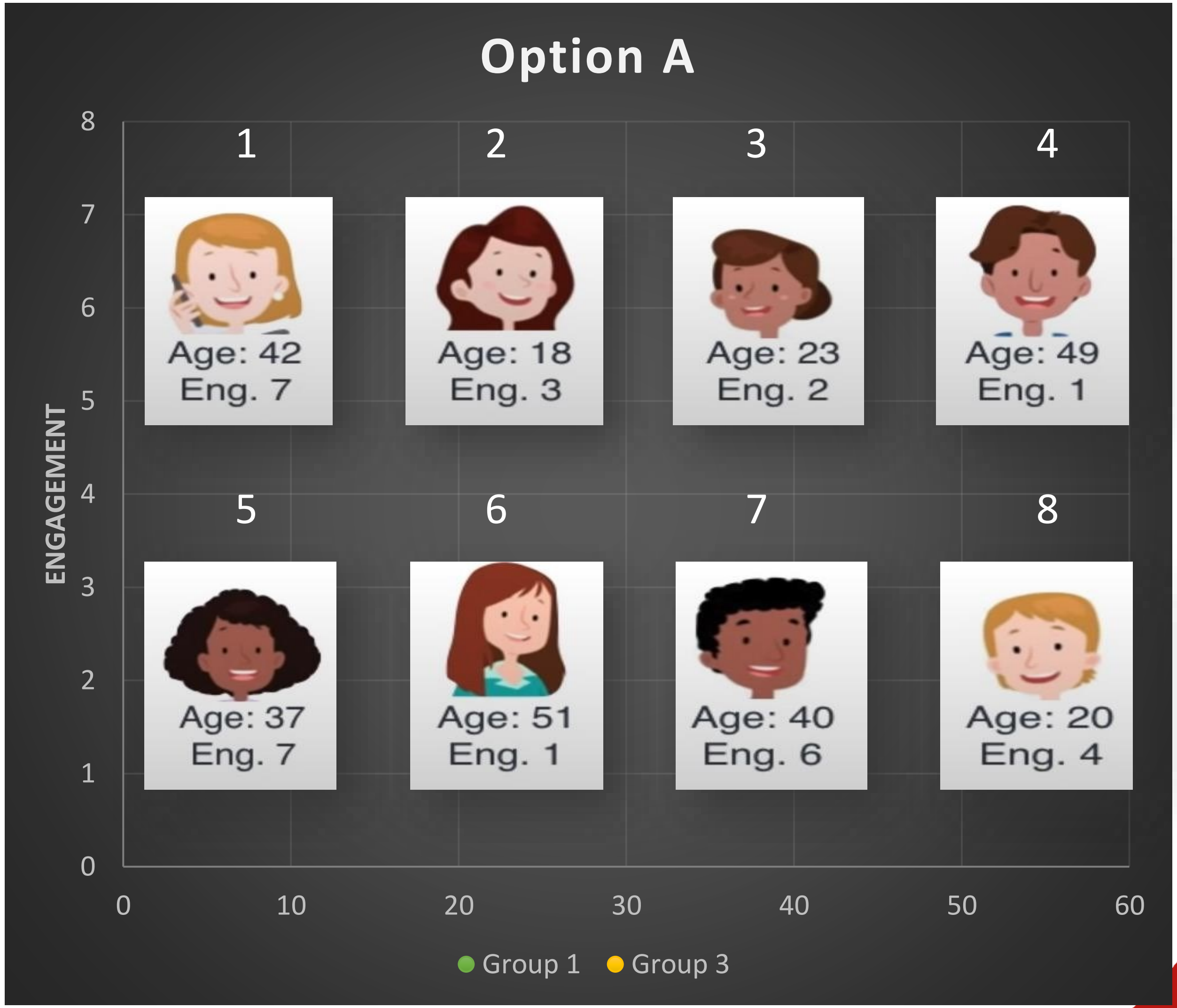
Lancaster University

Customer Engagement Data

Here are the **ages** (in years) and **engagements** (in days/weeks) of our customers that use our app.

If we have to put them into three groups to effectively serve them, how should we do that?

A	{1,5,6} {4,8} {2,3,7}
B	{1,8,3} {4,7,2} {5,6}
C	{2,8,3} {5,7,1} {4,6}
D	{3,7,8} {4,1} {2,5,6}

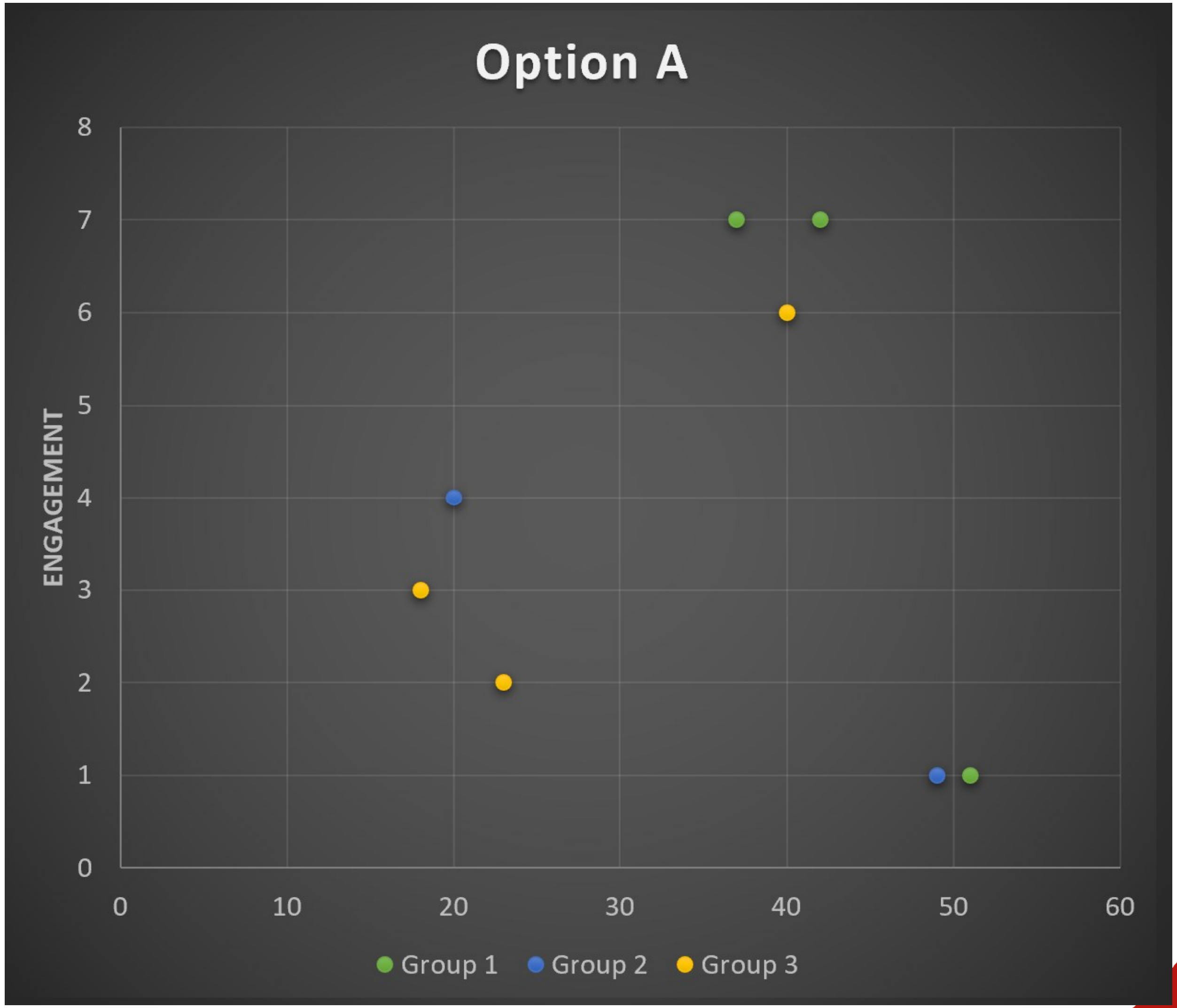


Customer Engagement Data

Here are the **ages** (in years) and **engagements** (in days/weeks) of our customers that use our app.

If we have to put them into three groups to effectively serve them, how should we do that?

A	{1,5,6} {4,8} {2,3,7}
B	{1,8,3} {4,7,2} {5,6}
C	{2,8,3} {5,7,1} {4,6}
D	{3,7,8} {4,1} {2,5,6}

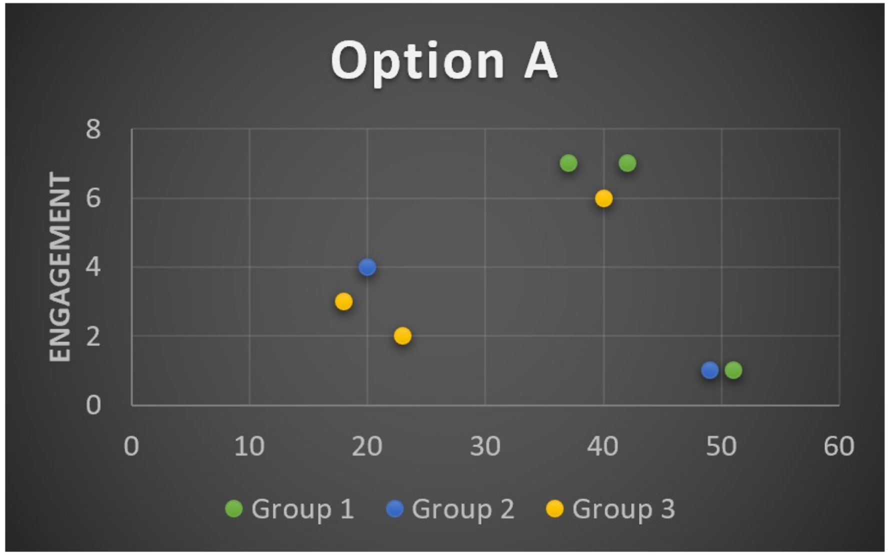


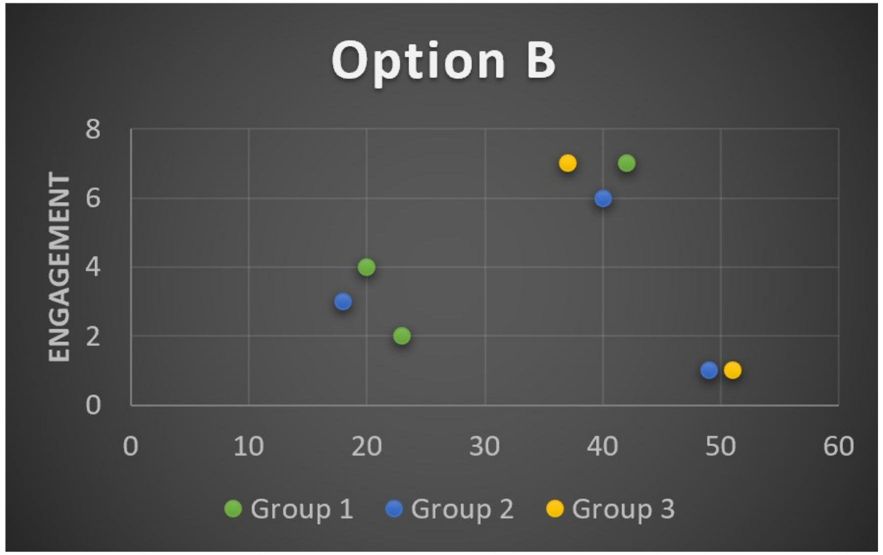
Customer Engagement Data

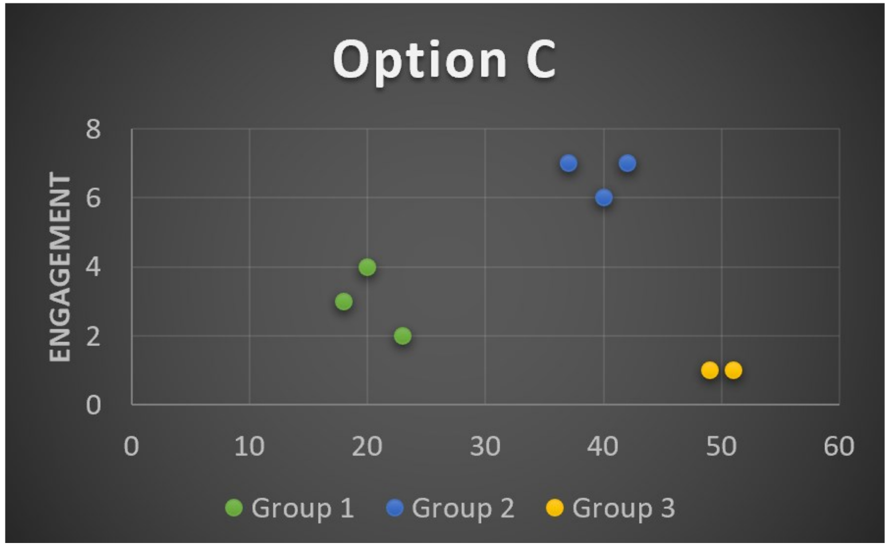
Here are the **ages** (in years) and **engagements** (in days/weeks) of our customers that use our app.

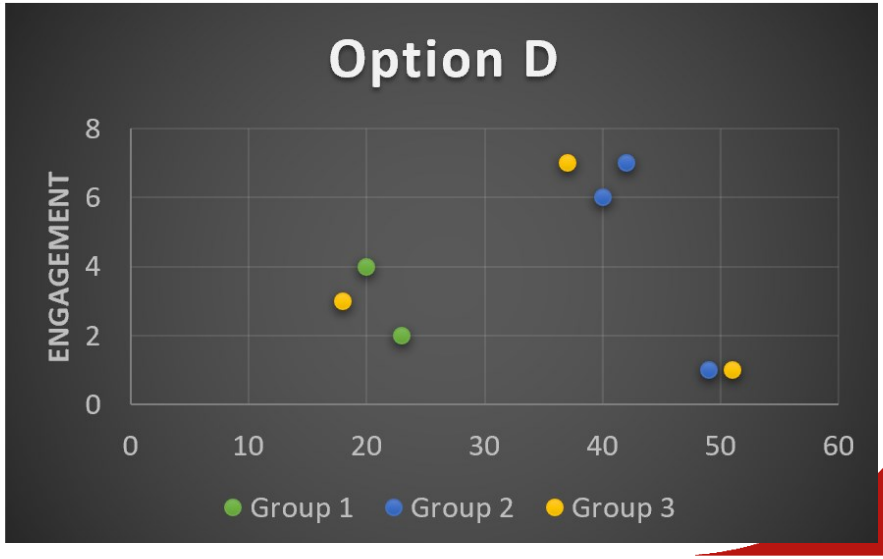
If we have to put them into three groups to effectively serve them, how should we do that?

A	{1,5,6} {4,8} {2,3,7}
B	{1,8,3} {4,7,2} {5,6}
C	{2,8,3} {5,7,1} {4,6}
D	{3,7,8} {4,1} {2,5,6}









Learning from Data

ML Goal

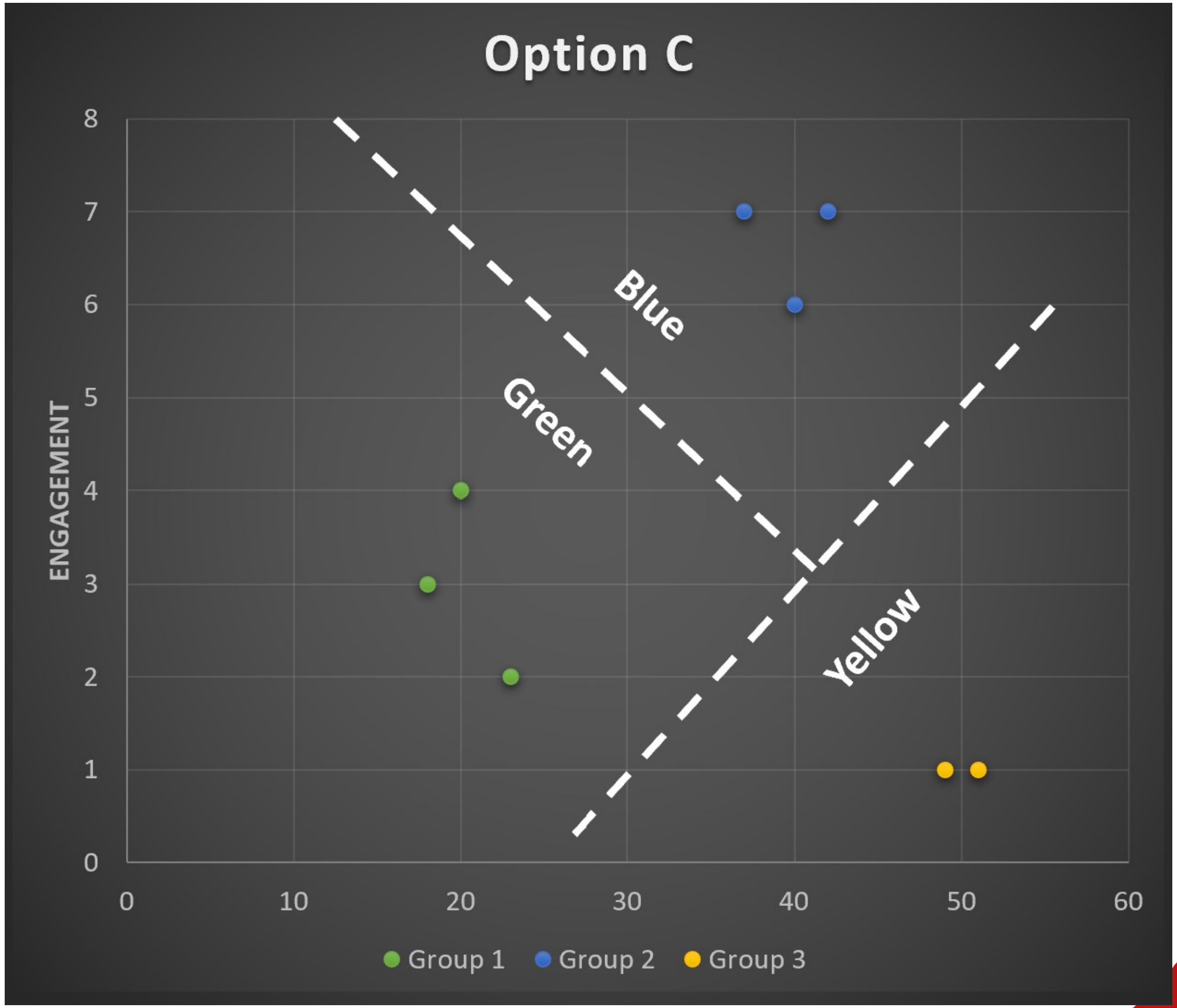
Learn from observed data in two ways

1: Clustering

- Identify meaningful patterns, clusters or groups in observed data points

2: Classification

- Classify or categorise new data points into one of the identified groups



What is Clustering?

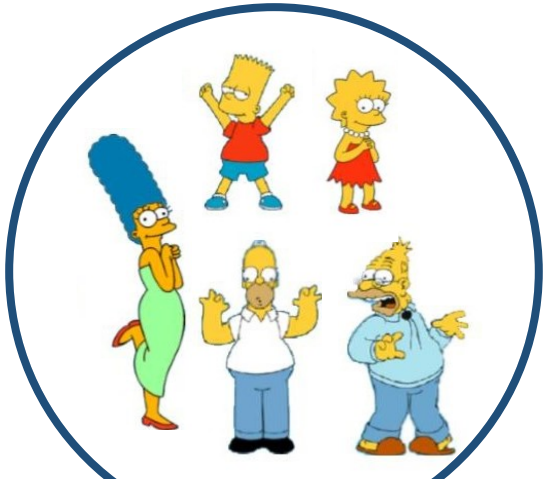
Grouping data into “clusters”

Optimisation with constraints:
- Number of clusters
- Minimum distance between clusters

Reduce dissimilarity between members in a cluster

Two common methods:

Hierarchical | K-Means



Simpson Family

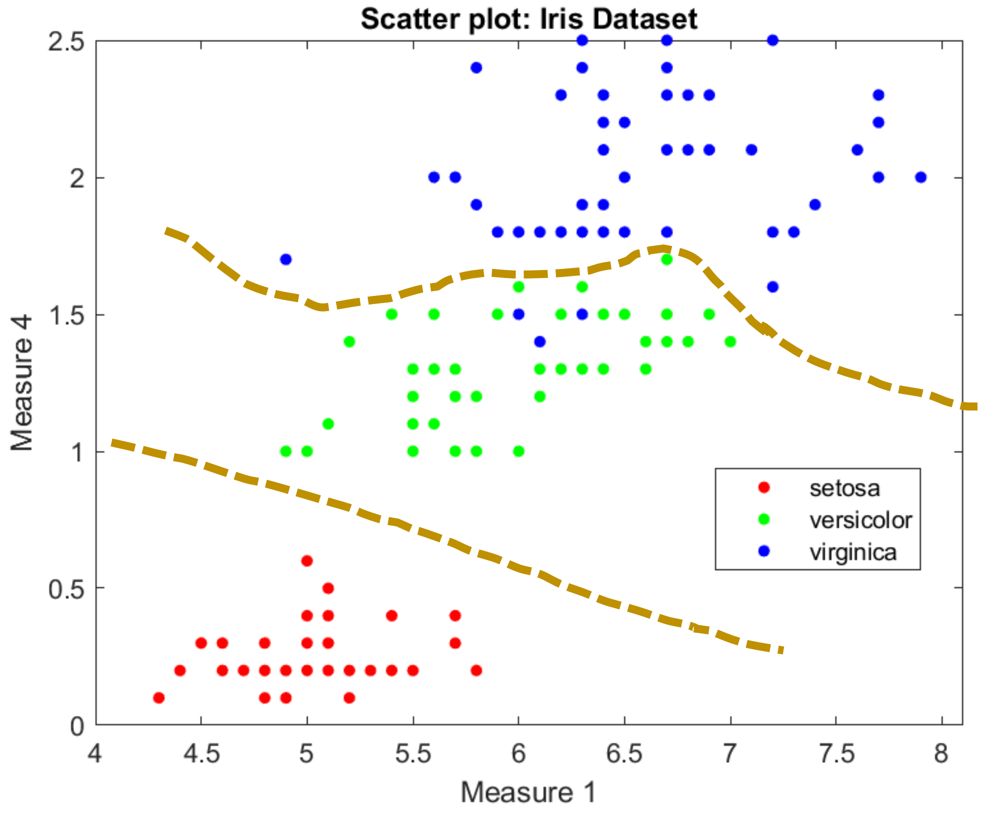
VS

Non-Simpson Family

K-Means Clustering

K-Means clustering is an **optimisation problem**

Aim:
Minimize the within-cluster sum of squares (WCSS) i.e. variance



Given n observations $(x_1, x_2, \dots, x_n)$, where each is a d-dimensional real vector, K-Means partitions the n observations into k sets S where $k \leq n$:

$$S = \{S_1, S_2, \dots, S_k\}$$

Such that the variance of each subset S_i is minimised.

K-Means Clustering

Lancaster University

K-Means clustering is an **optimisation problem**

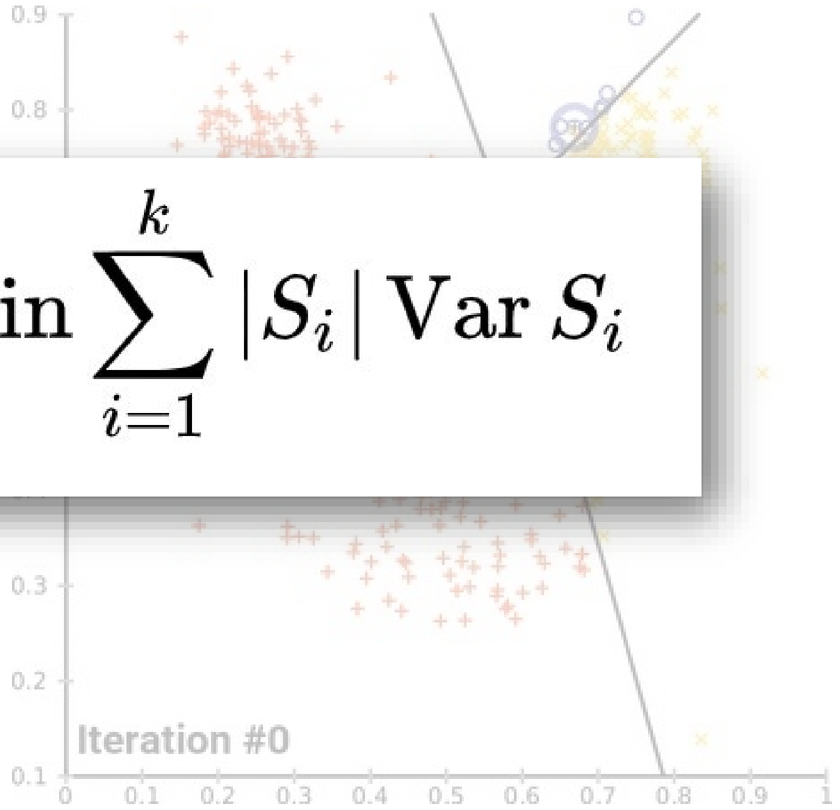
Aim:
Mi
(W

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \left\| \mathbf{x} - \boldsymbol{\mu}_i \right\|^2 = \underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i$$

Giv
a d
n observations into k sets S where $k \leq n$:

$$S = \{S_1, S_2, \dots, S_k\}$$

Such that the variance of each subset S_i is minimised.



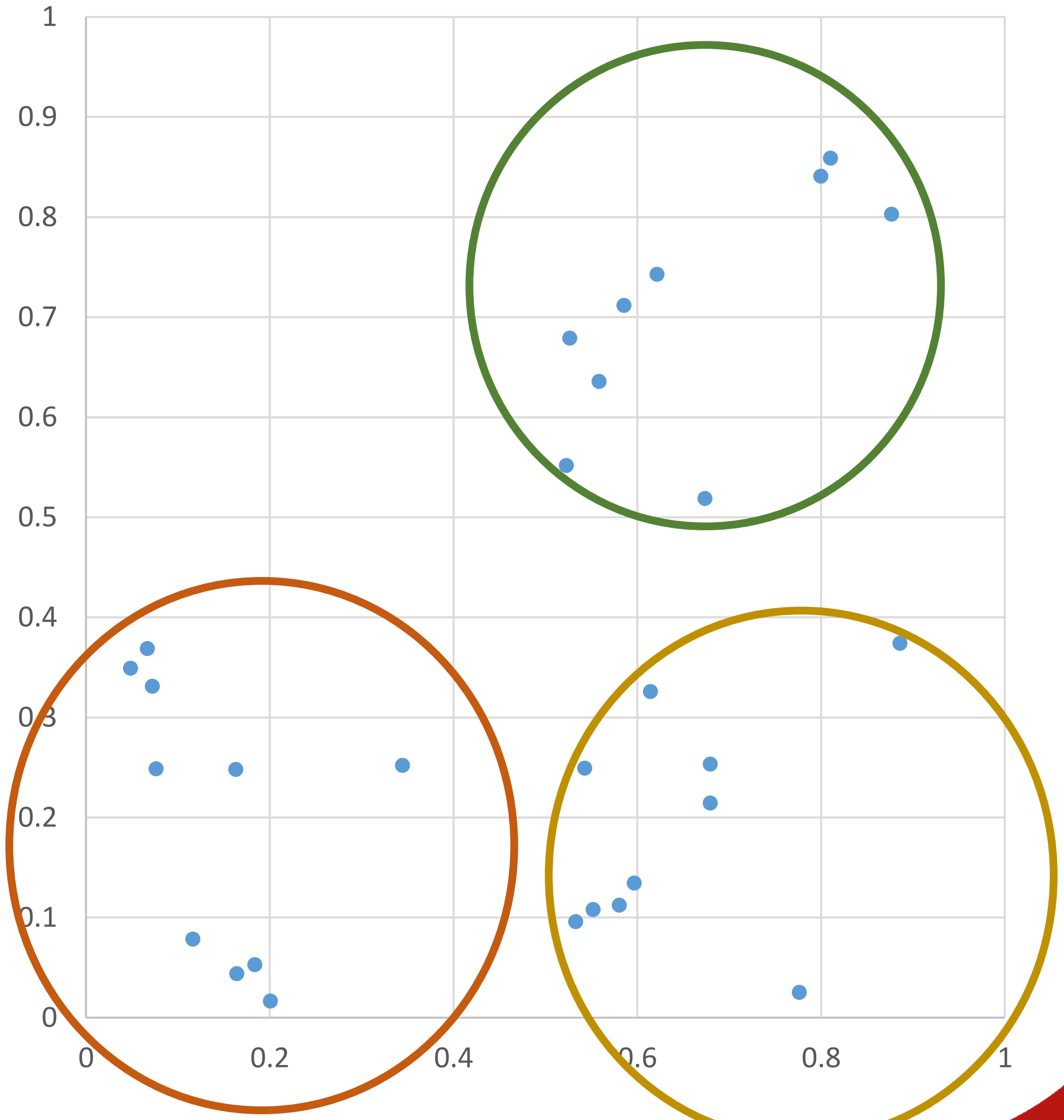
Source: K-Means Clustering on Wikipedia

How K-Means Works

Lancaster University

Algorithm:

- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
    - Create k clusters by assigning each point to closest centroid
        - Calculate distance from centroids
        - Find minimum distance
        - Copy label
    - compute k new centroids by averaging points in each cluster
    - if centroids don't change:
        - stop

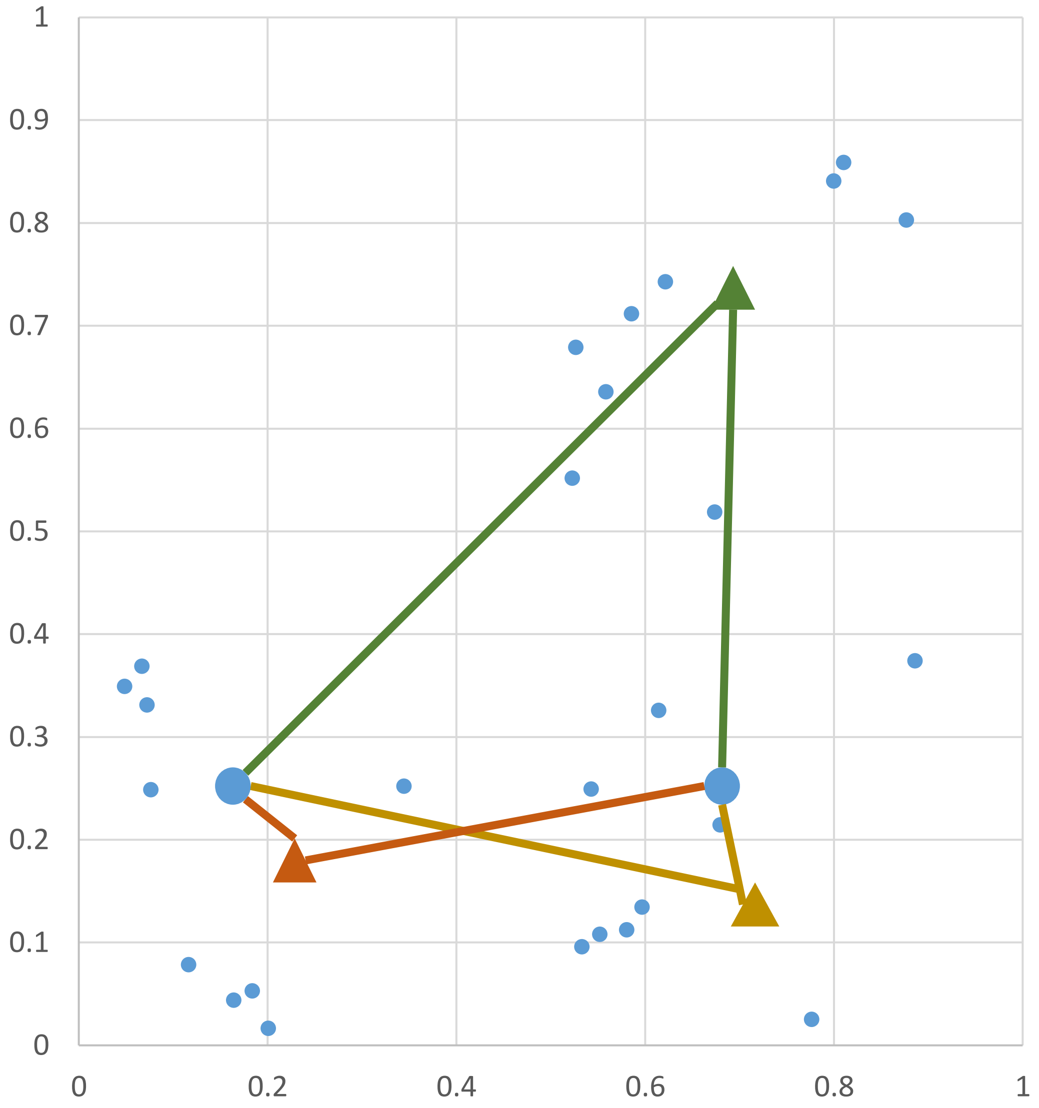


How K-Means Works

Algorithm:

- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
  - Create k clusters by assigning each point to closest centroid
    - Calculate distance from centroids
    - Find minimum distance
    - Copy label
  - compute k new centroids by averaging points in each cluster
  - if centroids don't change:
    - stop

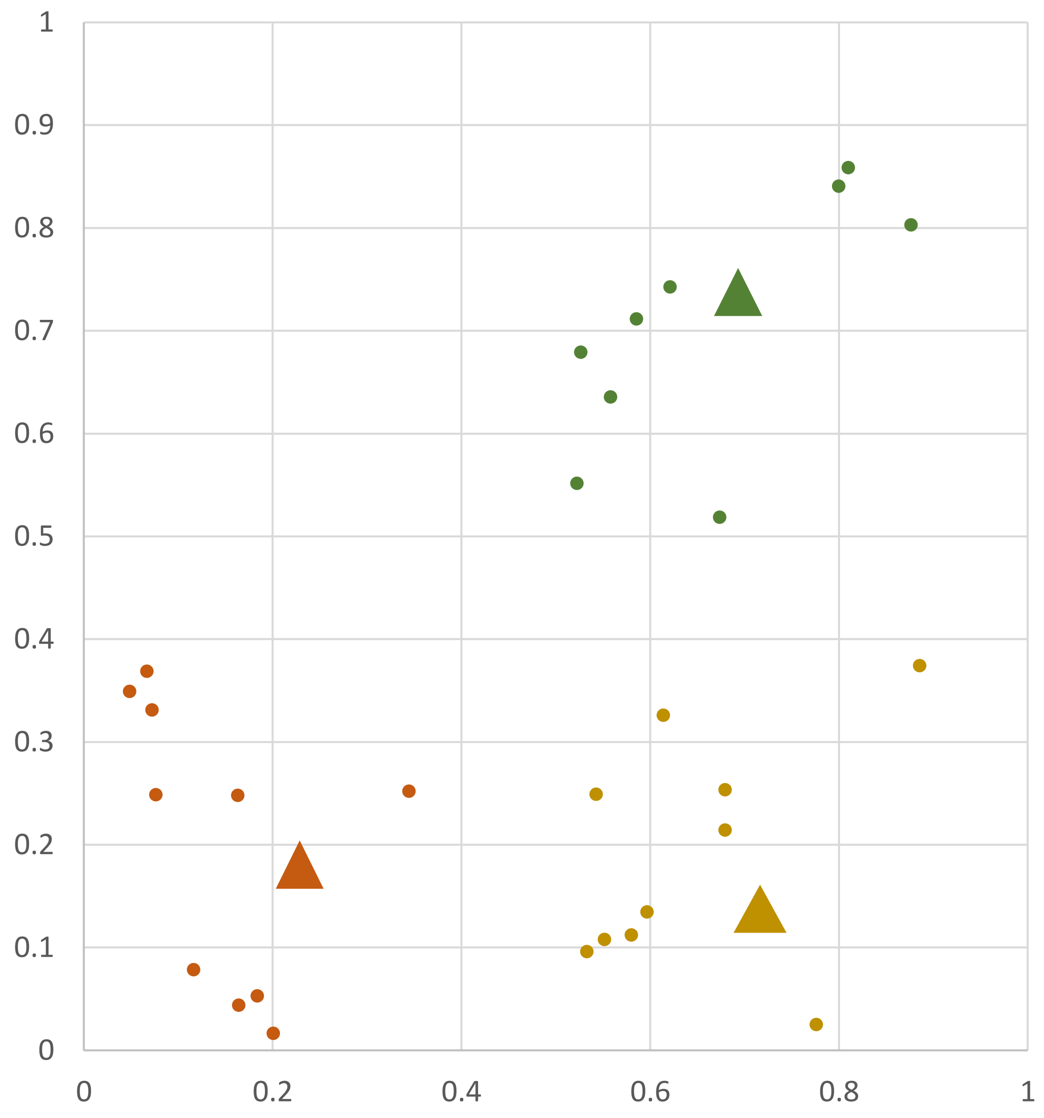
kNN



How K-Means Works

Algorithm:

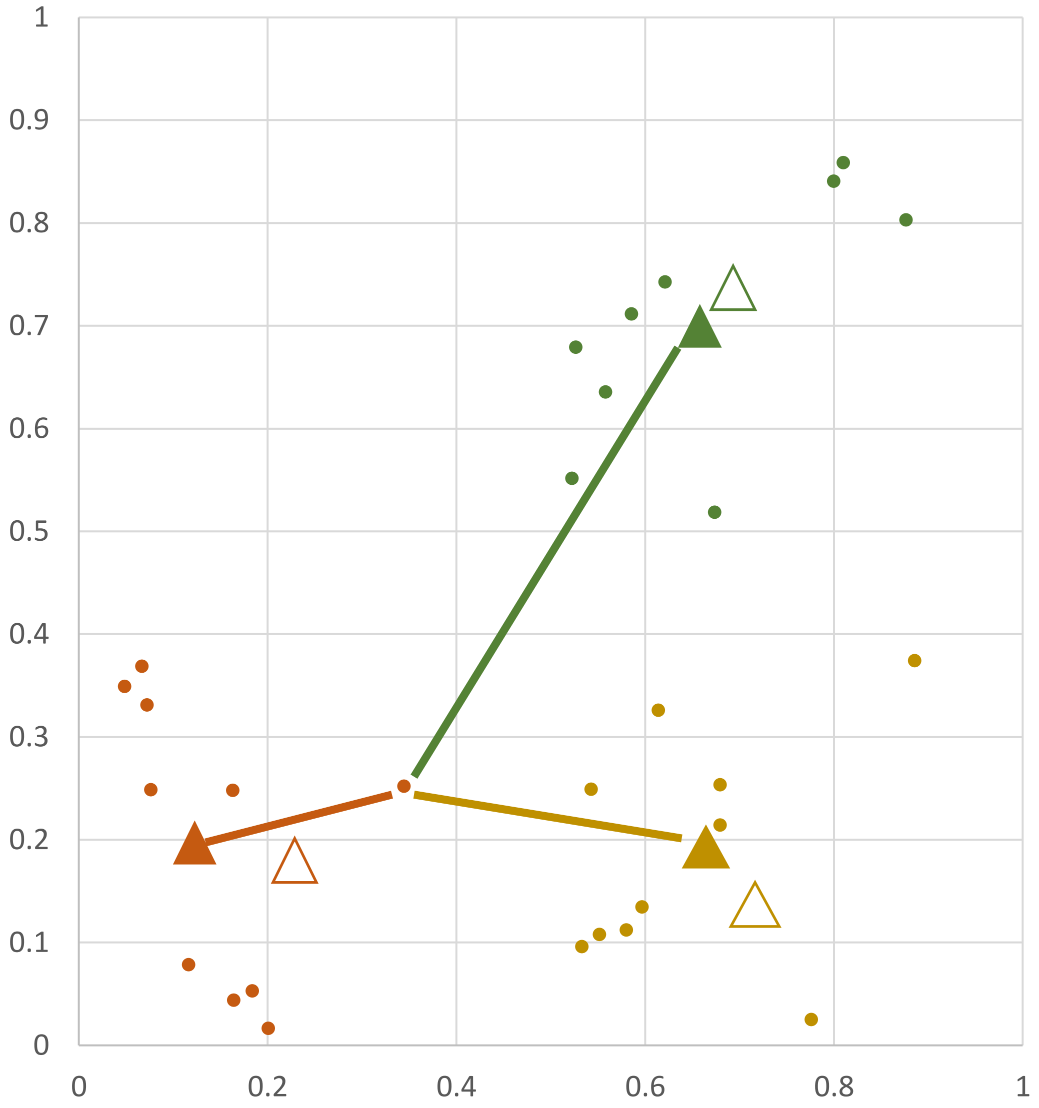
- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
    - Create k clusters by assigning each point to closest centroid
        - Calculate distance from centroids
        - Find minimum distance
        - Copy label
    - compute k new centroids by averaging points in each cluster
    - if centroids don't change:
        - stop



How K-Means Works

Algorithm:

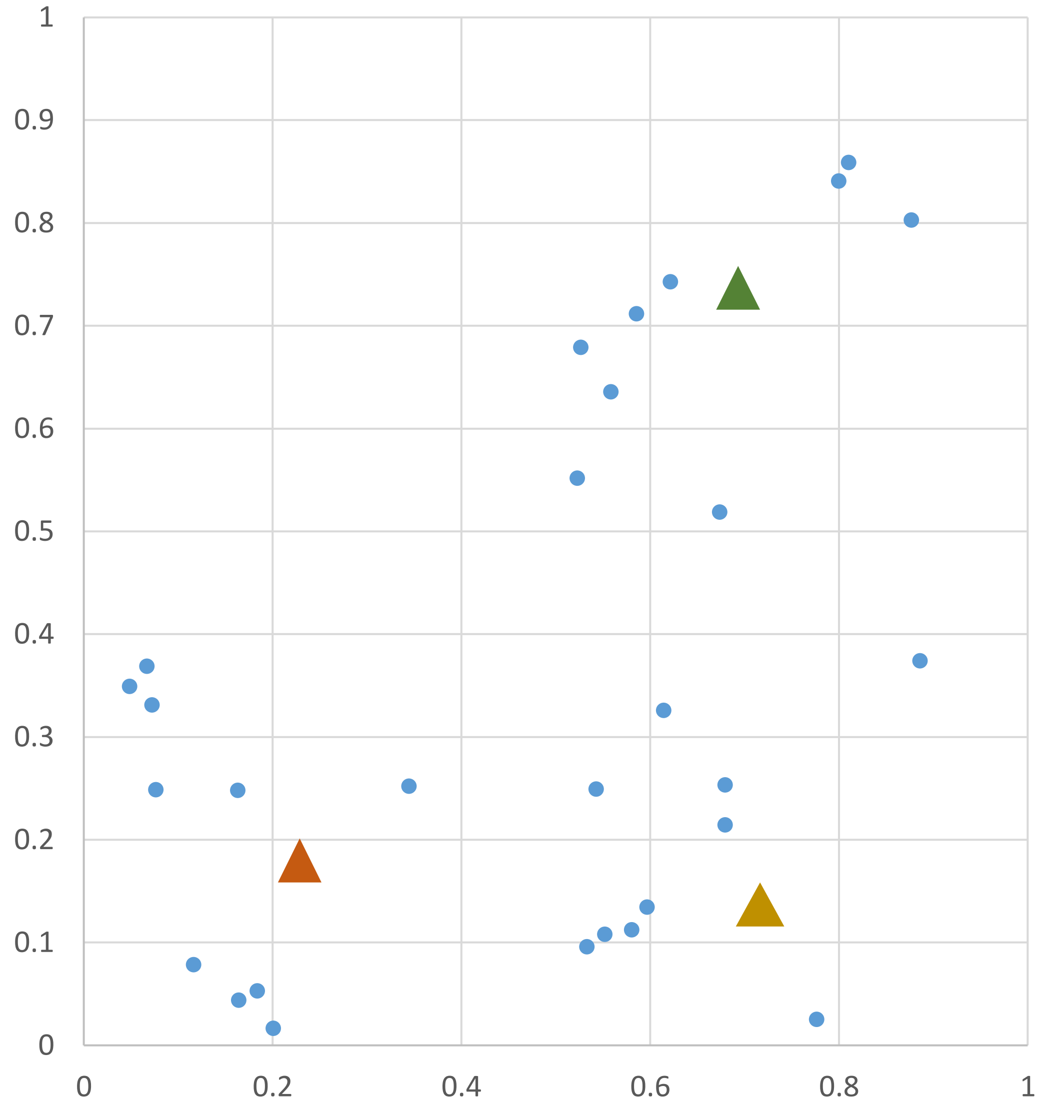
- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
    - Create k clusters by assigning each point to closest centroid
        - Calculate distance from centroids
        - Find minimum distance
        - Copy label
    - compute k new centroids by averaging points in each cluster
    - if centroids don't change:
        - stop



Changing the Initialisation

Algorithm:

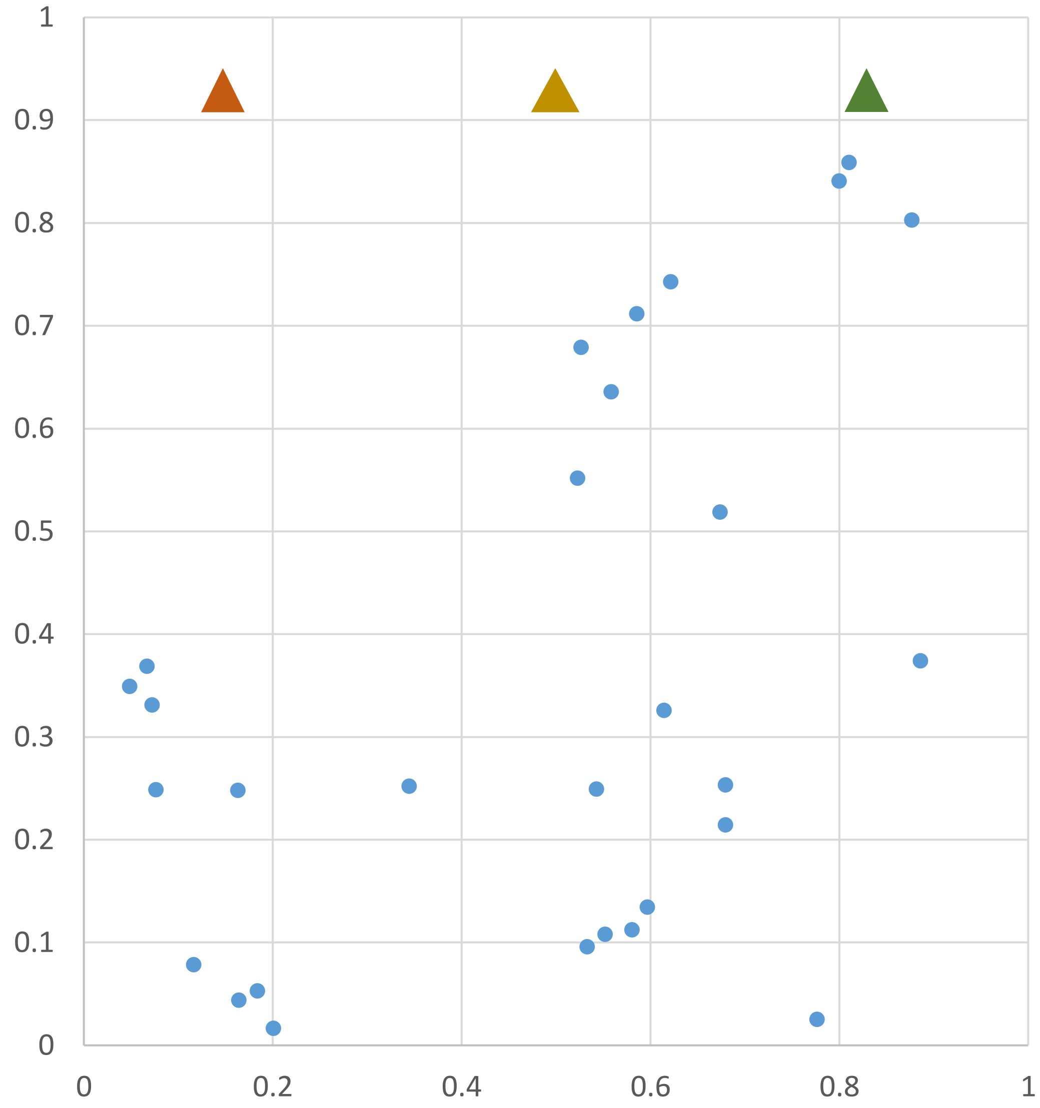
- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
    - Create k clusters by assigning each point to closest centroid
        - Calculate distance from centroids
        - Find minimum distance
        - Copy label
    - compute k new centroids by averaging points in each cluster
    - if centroids don't change:
        - stop



Changing the Initialisation

Algorithm:

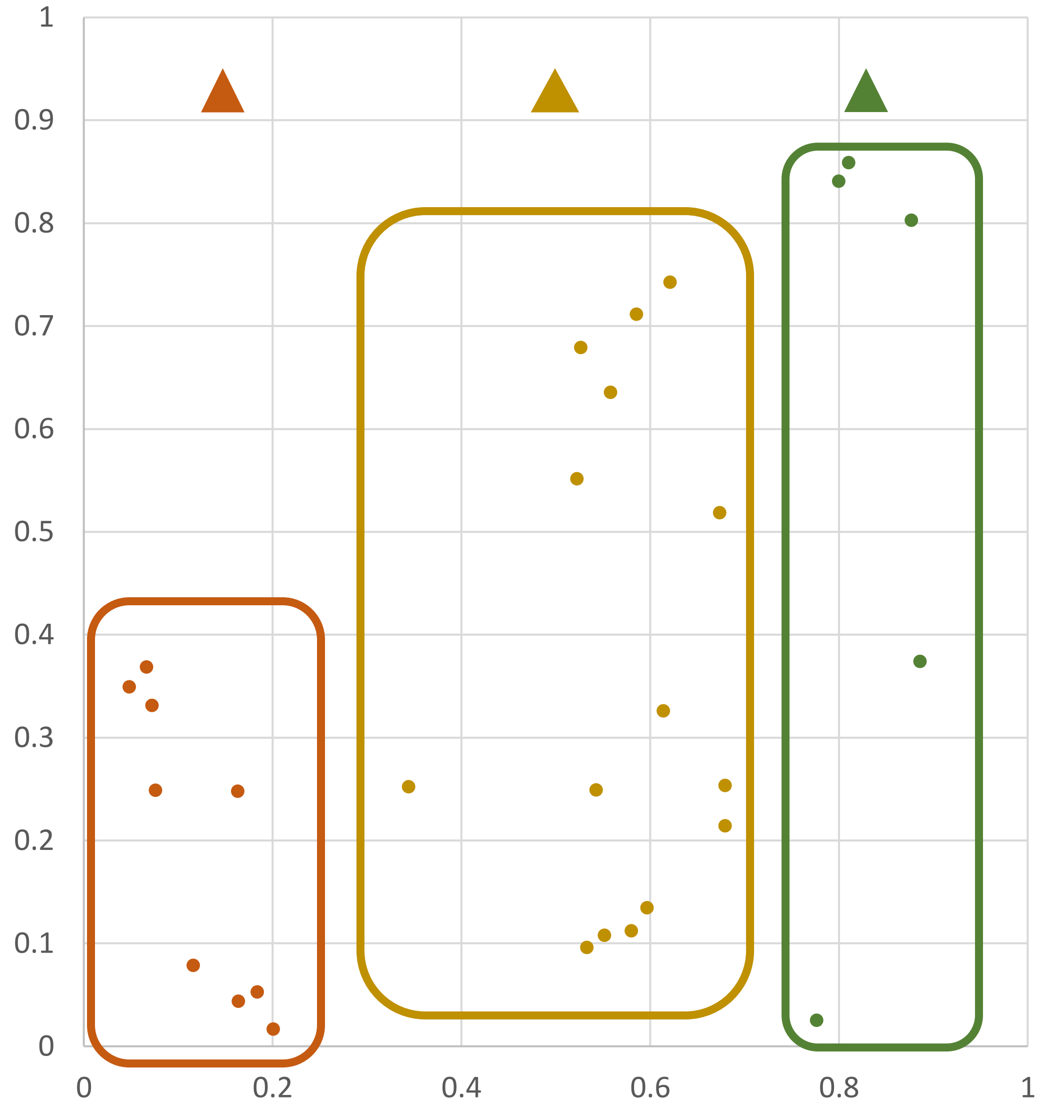
- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
  - Create k clusters by assigning each point to closest centroid
    - Calculate distance from centroids
    - Find minimum distance
    - Copy label
  - compute k new centroids by averaging points in each cluster
  - if centroids don't change:
    - stop



Changing the Initialisation

Algorithm:

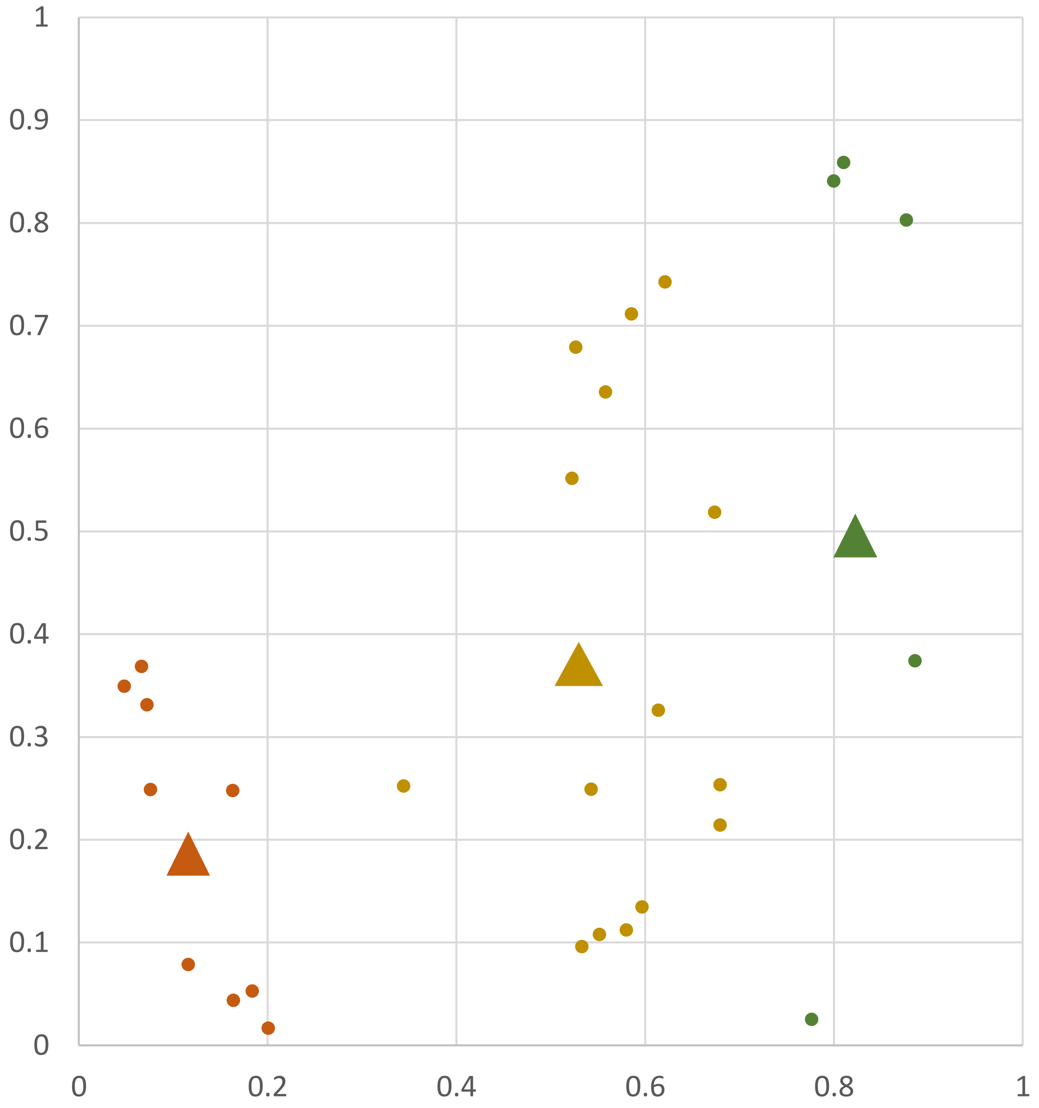
- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
  - Create k clusters by assigning each point to closest centroid
    - Calculate distance from centroids
    - Find minimum distance
    - Copy label
  - compute k new centroids by averaging points in each cluster
  - if centroids don't change:
    - stop



Changing the Initialisation

Algorithm:

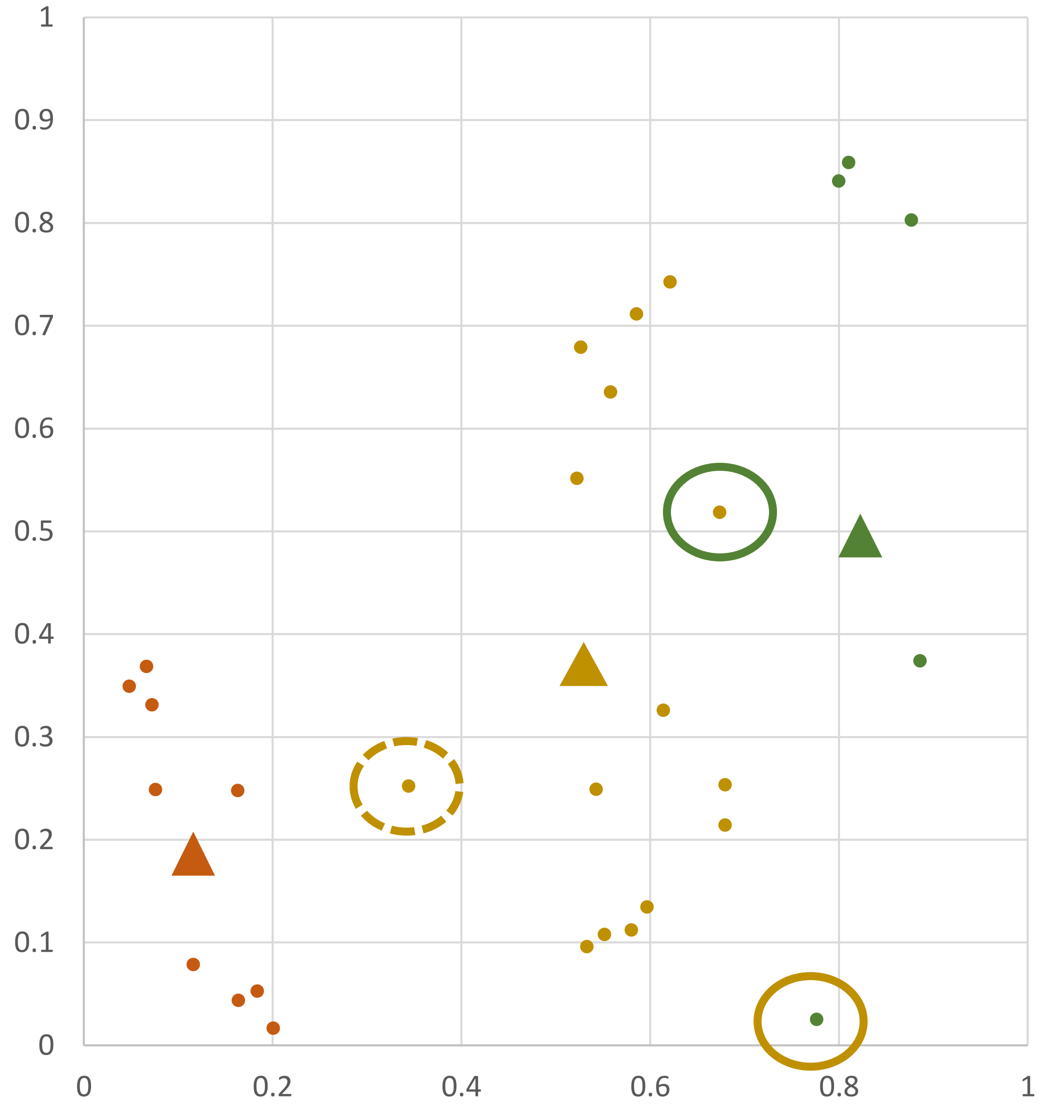
- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
  - Create k clusters by assigning each point to closest centroid
    - Calculate distance from centroids
    - Find minimum distance
    - Copy label
  - compute k new centroids by averaging points in each cluster
  - if centroids don't change:
    - stop



Changing the Initialisation

Algorithm:

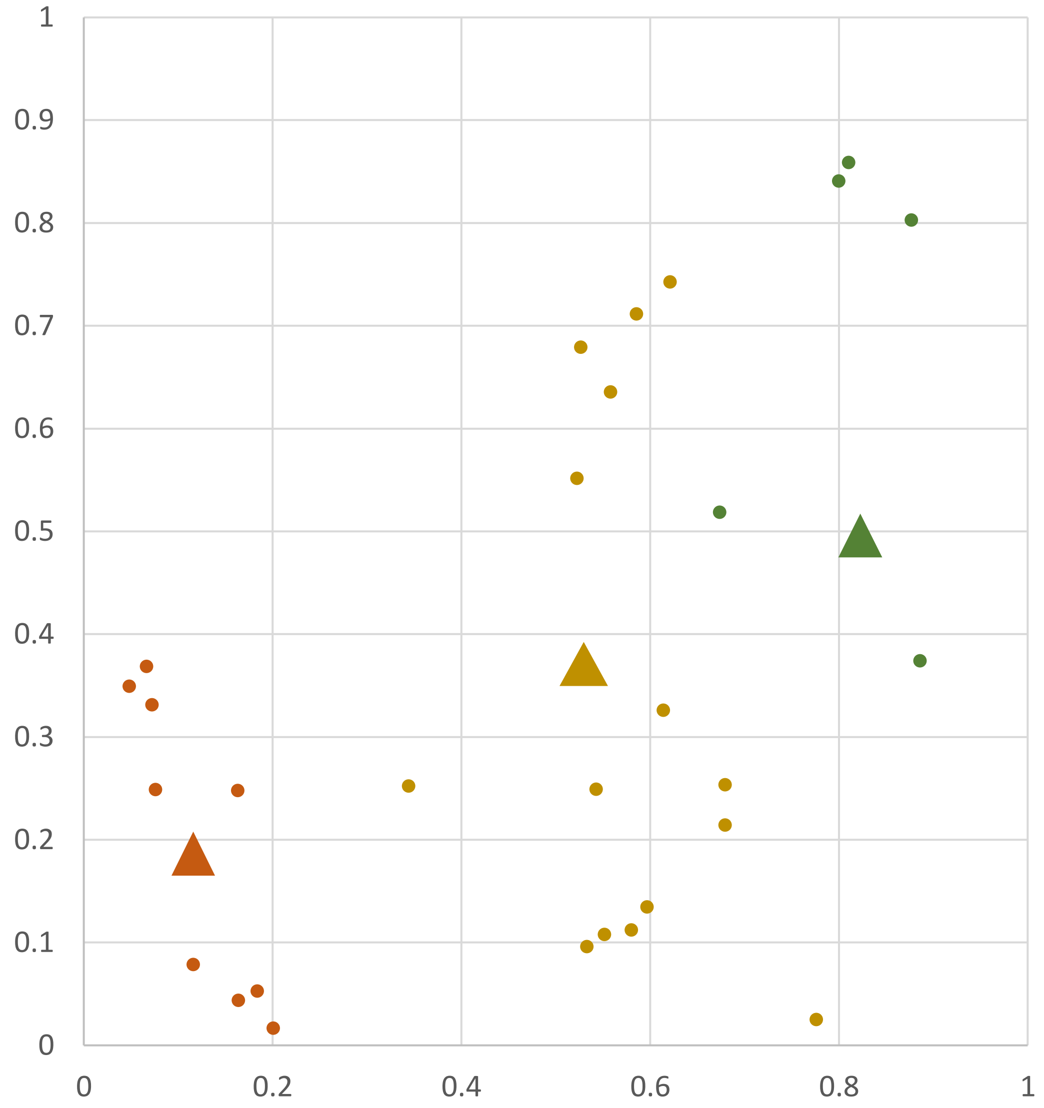
- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
  - Create k clusters by assigning each point to closest centroid
    - Calculate distance from centroids
    - Find minimum distance
    - Copy label
  - compute k new centroids by averaging points in each cluster
  - if centroids don't change:
    - stop



Changing the Initialisation

Algorithm:

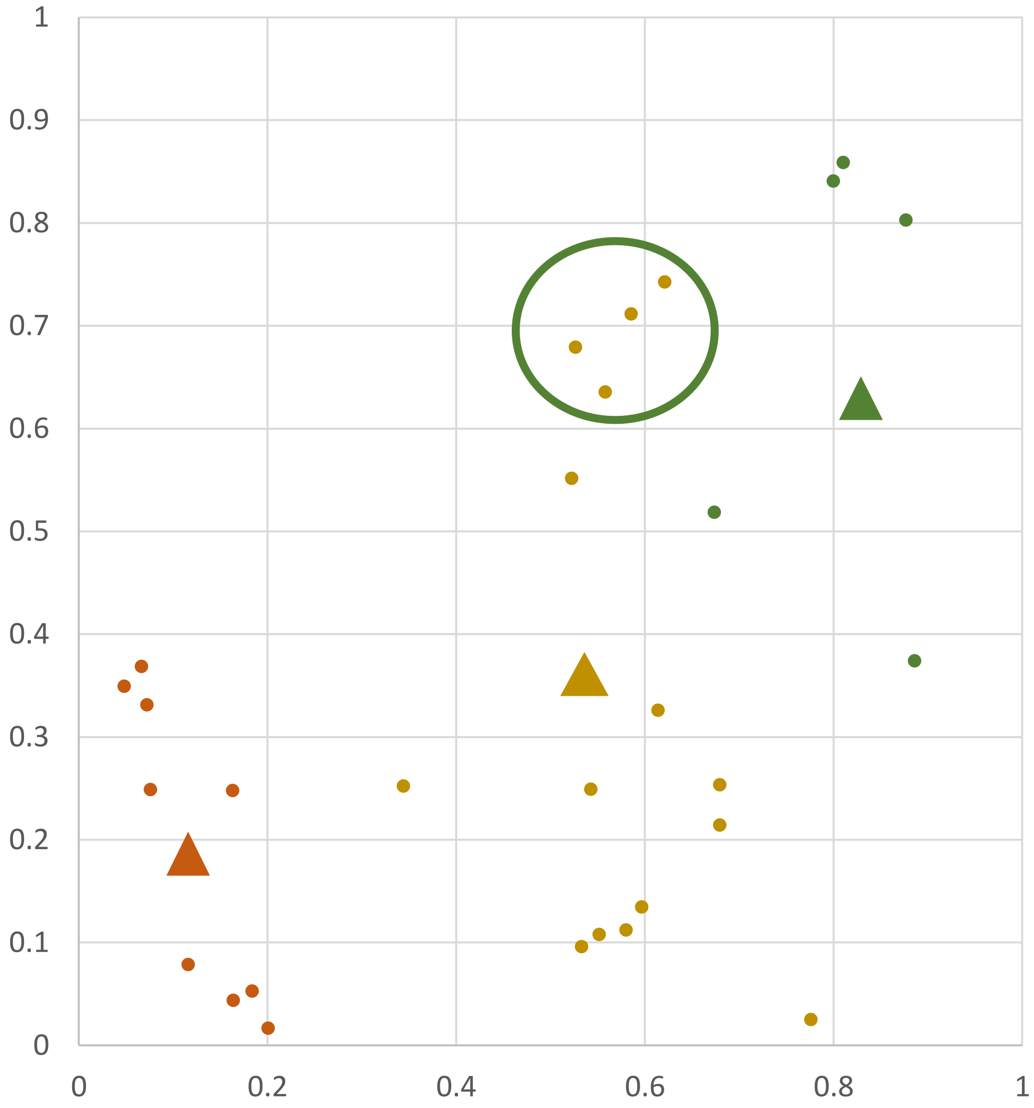
- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
  - Create k clusters by assigning each point to closest centroid
    - Calculate distance from centroids
    - Find minimum distance
    - Copy label
  - compute k new centroids by averaging points in each cluster
  - if centroids don't change:
    - stop



Changing the Initialisation

Algorithm:

- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
  - Create k clusters by assigning each point to closest centroid
    - Calculate distance from centroids
    - Find minimum distance
    - Copy label
  - compute k new centroids by averaging points in each cluster
  - if centroids don't change:
    - stop

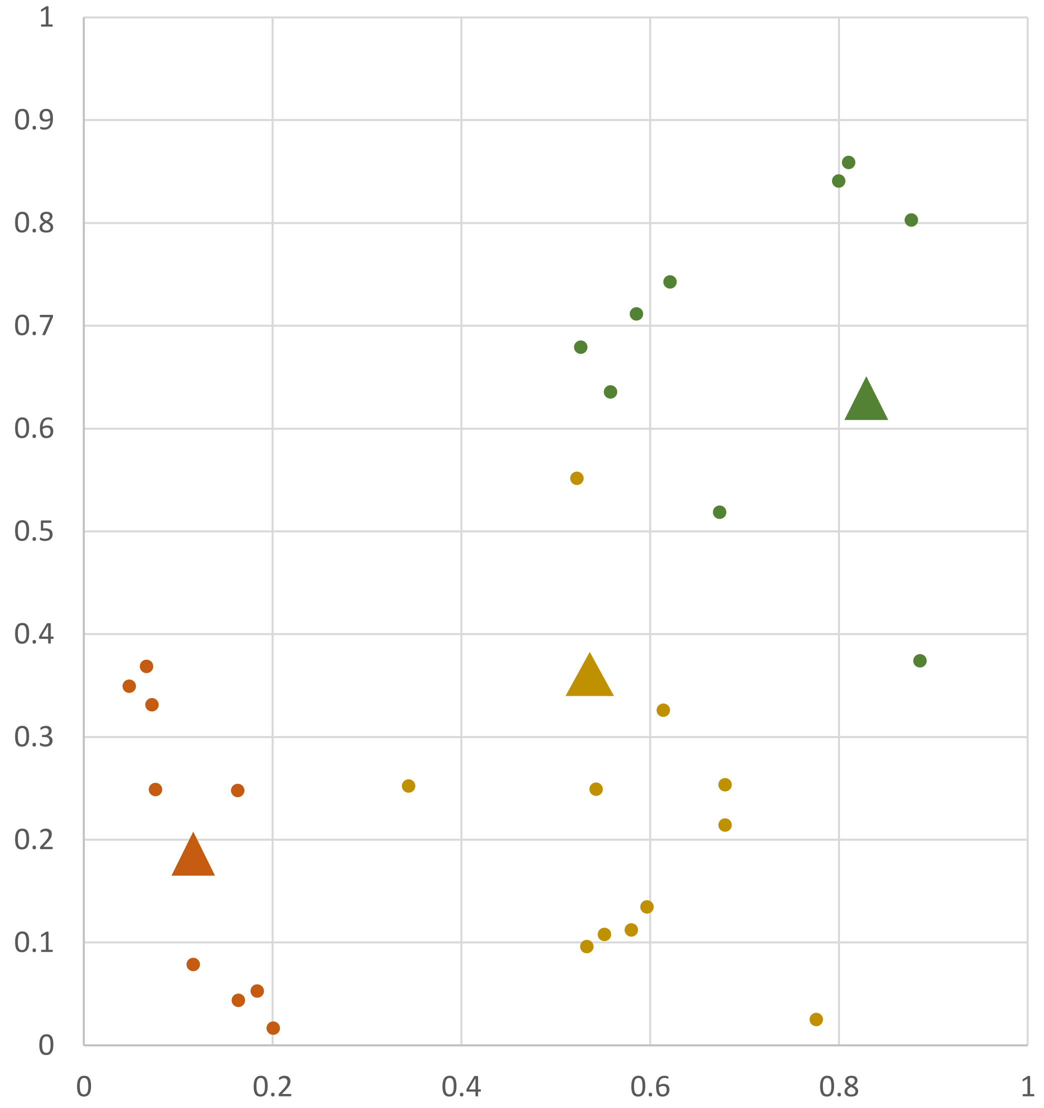


Changing the Initialisation

Lancaster University

Algorithm:

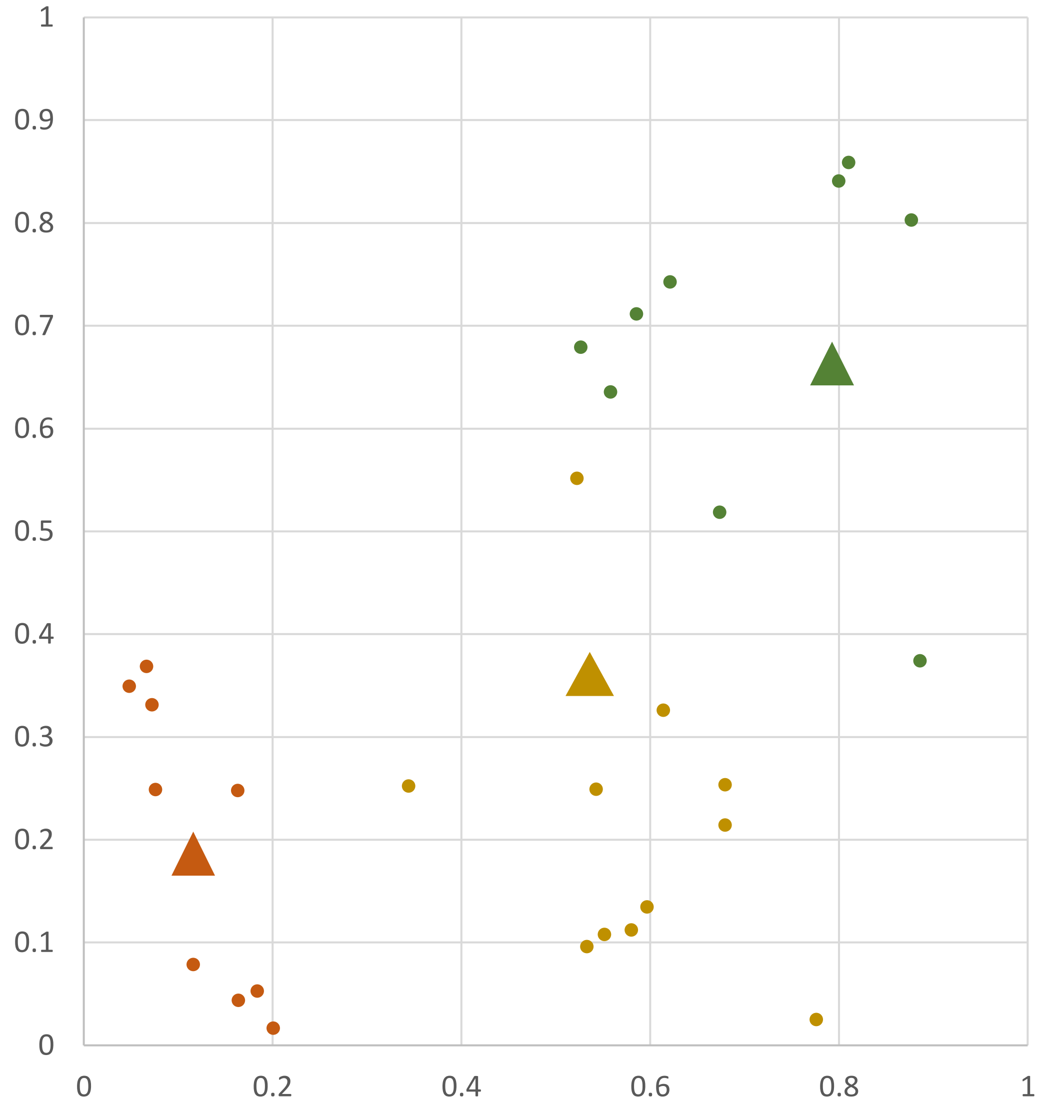
- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
    - Create k clusters by assigning each point to closest centroid
        - Calculate distance from centroids
        - Find minimum distance
        - Copy label
    - compute k new centroids by averaging points in each cluster
    - if centroids don't change:
        - stop



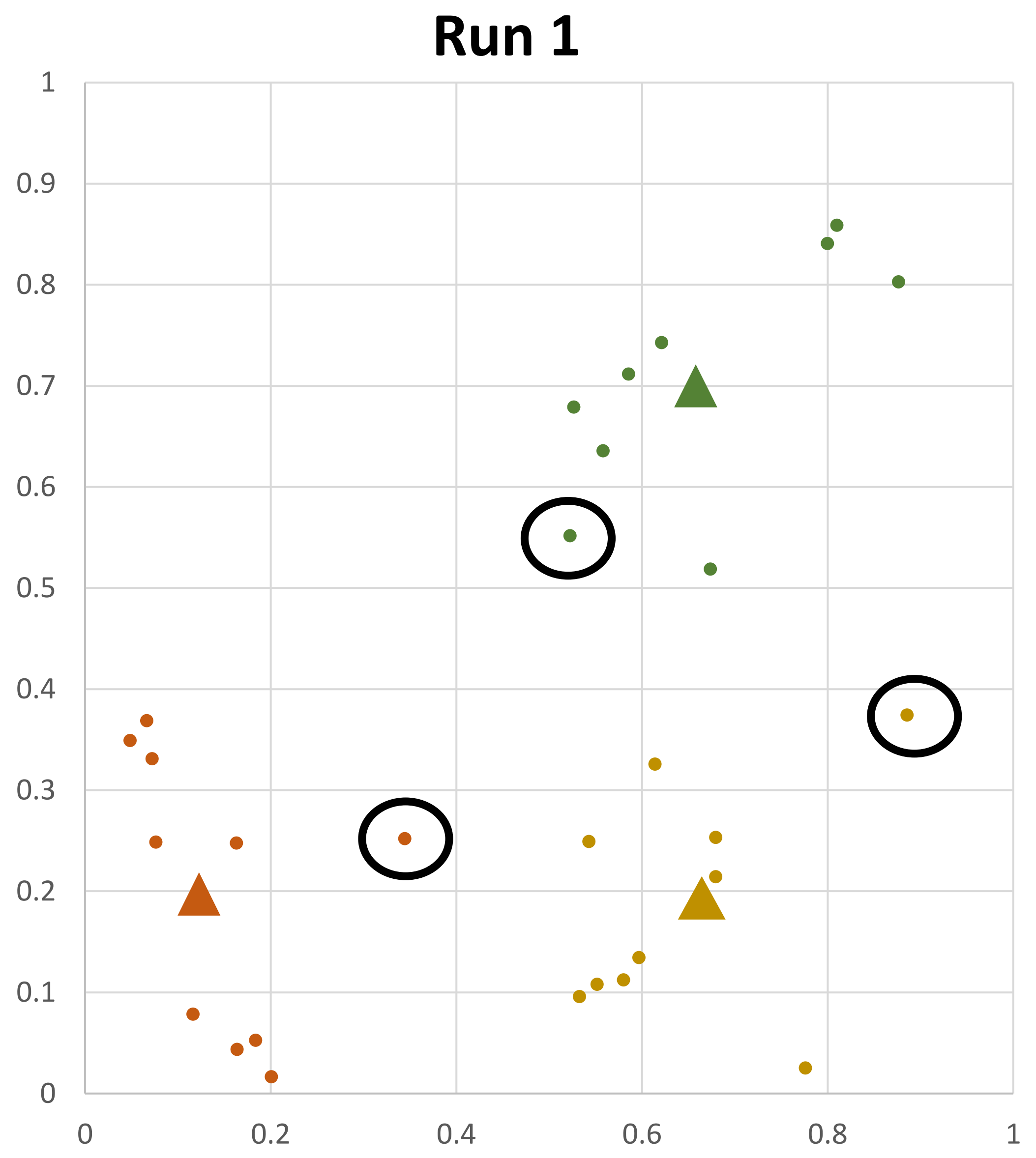
Changing the Initialisation

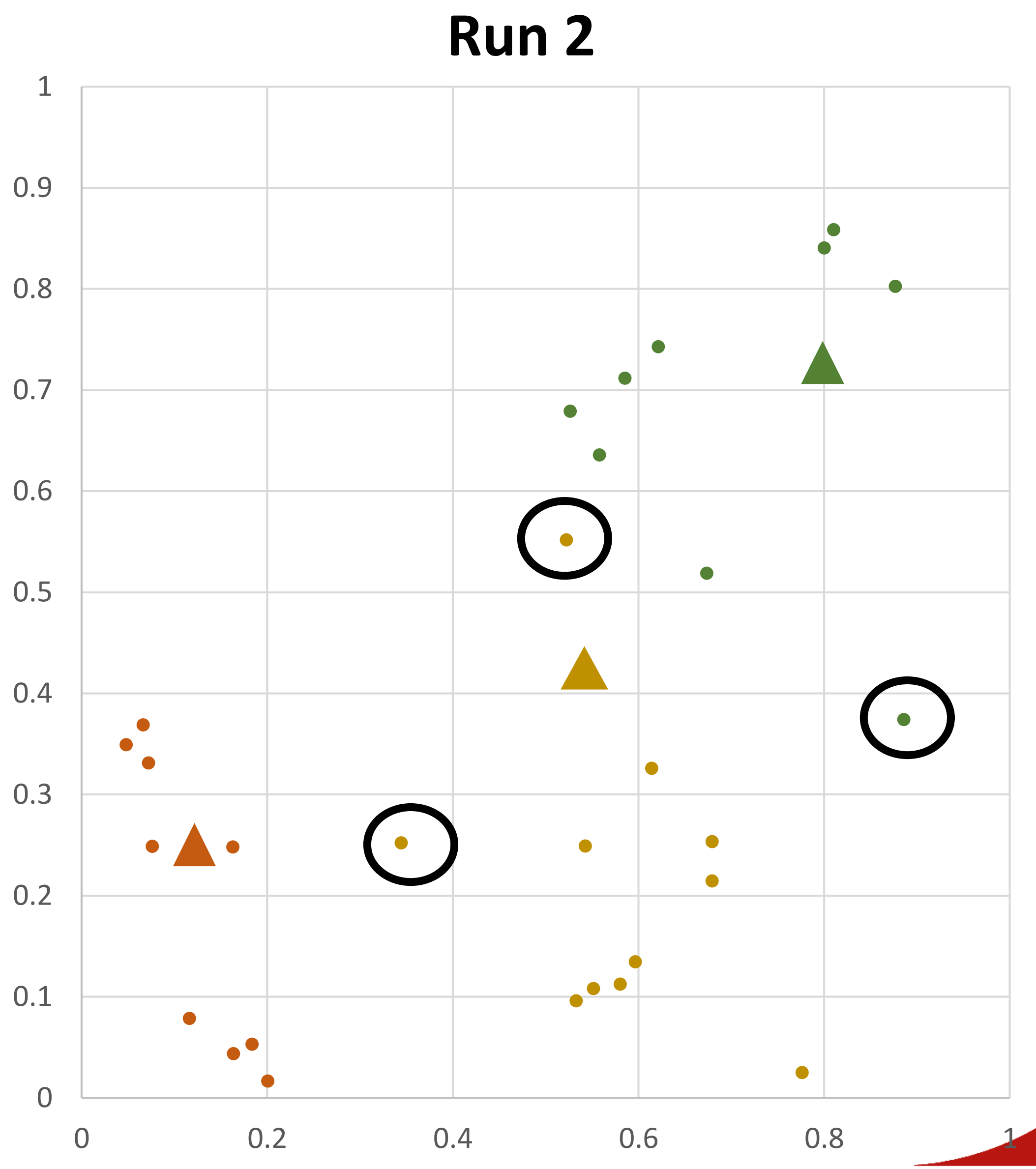
Algorithm:

- Choose k, i.e. the number of clusters
- Place k centroids
- while true:
    - Create k clusters by assigning each point to closest centroid
        - Calculate distance from centroids
        - Find minimum distance
        - Copy label
    - compute k new centroids by averaging points in each cluster
    - if centroids don't change:
        - stop



Changing the Initialisation





K-Means Summary

Efficiency

- K-Mean is efficient but has some weaknesses

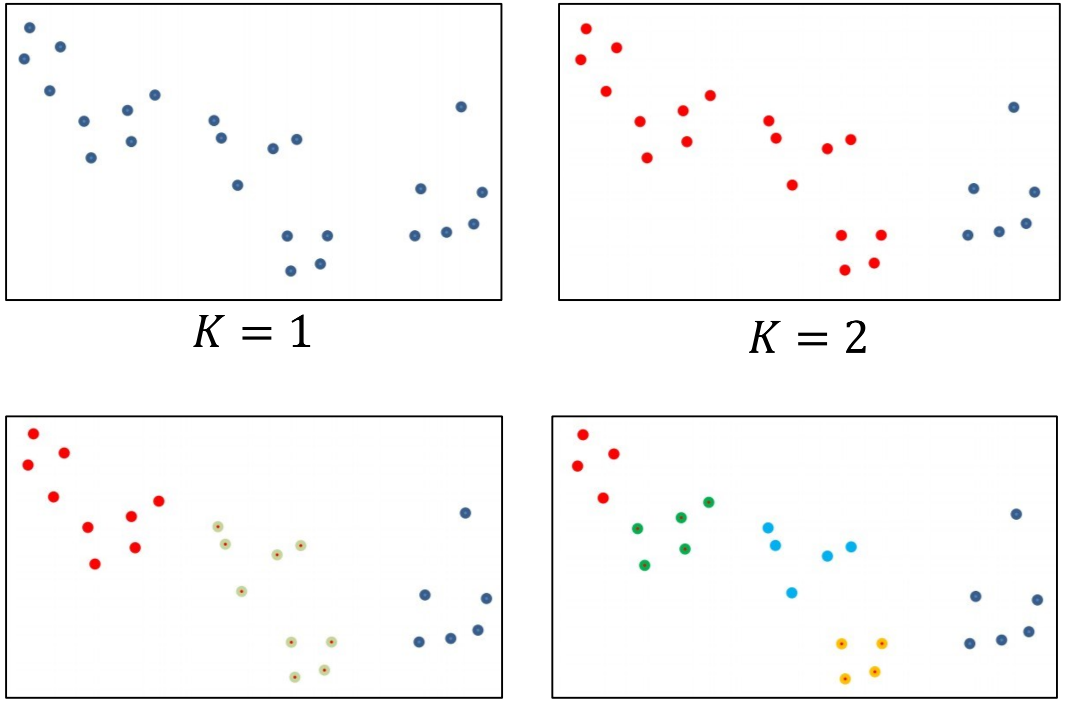
K-Means Summary

Efficiency

- K-Mean is efficient but has some weaknesses

Number of Clusters

- You don't necessarily know $\boldsymbol{k}$, i.e. number of clusters
- You can choose "wrong" k and get strange results.



$K = 1$ $K = 2$ $K = 3$ $K = 5$

How do we choose the right k?

K-Means Summary

Efficiency

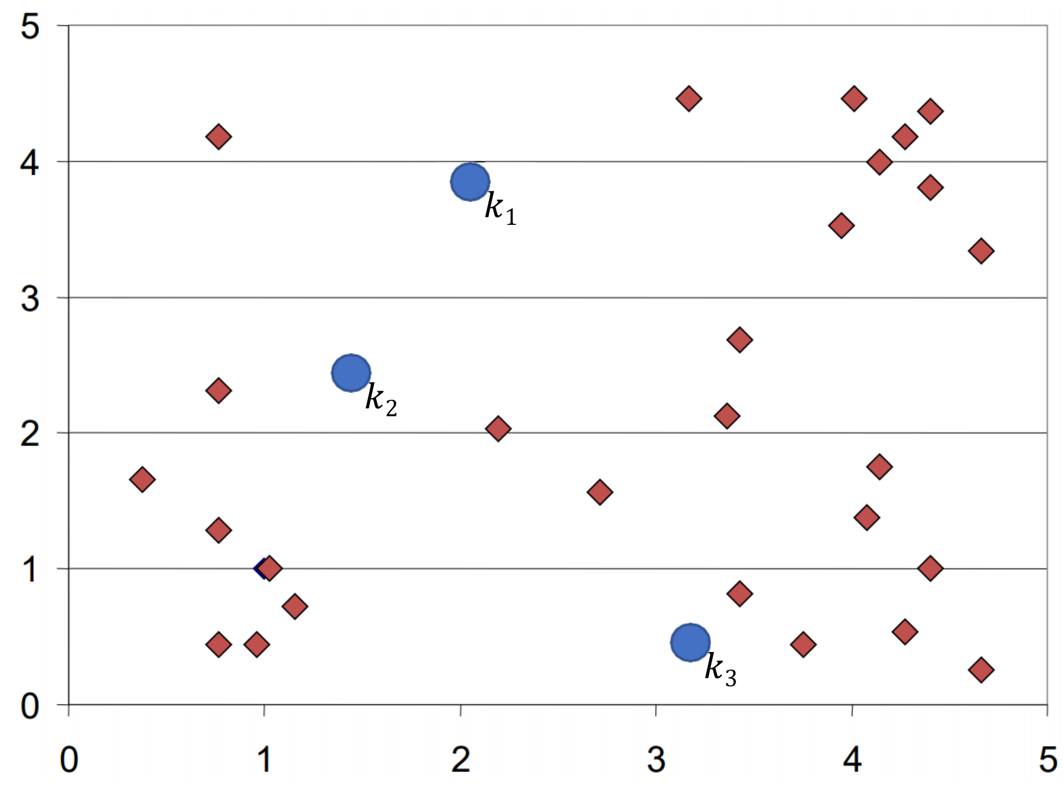
- K-Mean is efficient but has some weaknesses

Number of Clusters

- You don't necessarily know $\boldsymbol{k}$, i.e. number of clusters
- You can choose "wrong" k and get strange results.

It is non-deterministic

- Initial centroids are chosen at random



K-Means Summary

Efficiency

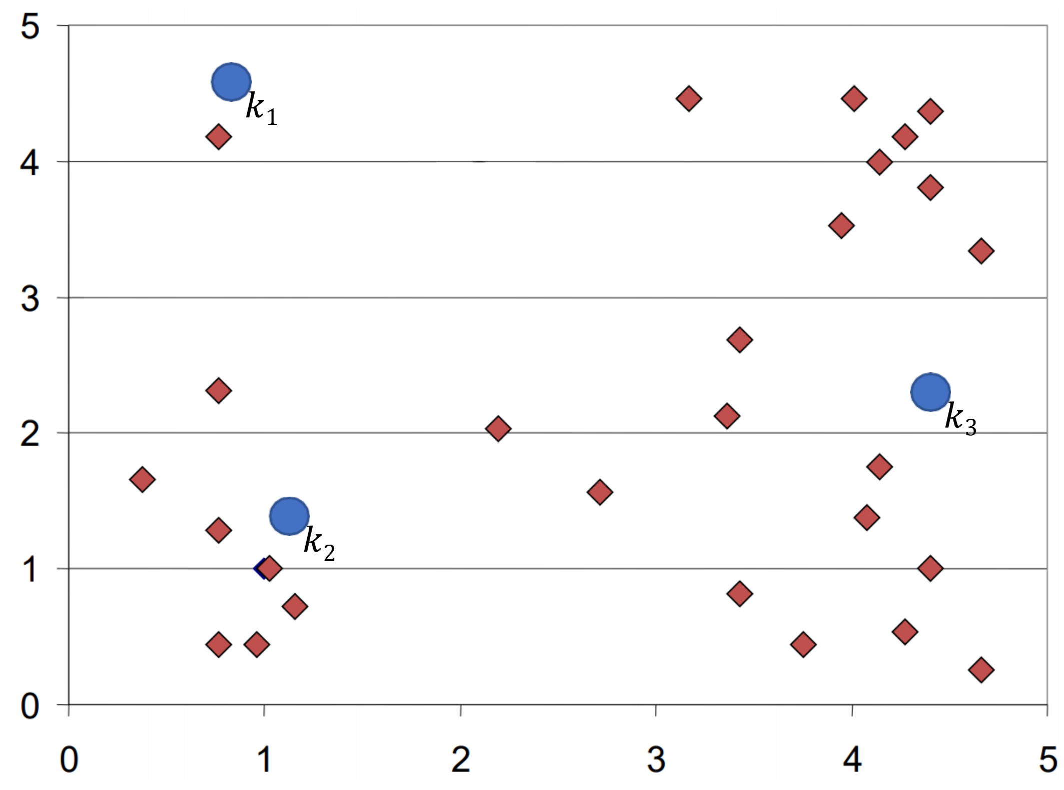
- K-Mean is efficient but has some weaknesses

Number of Clusters

- You don't necessarily know $\boldsymbol{k}$, i.e. number of clusters
- You can choose "wrong" k and get strange results.

It is non-deterministic

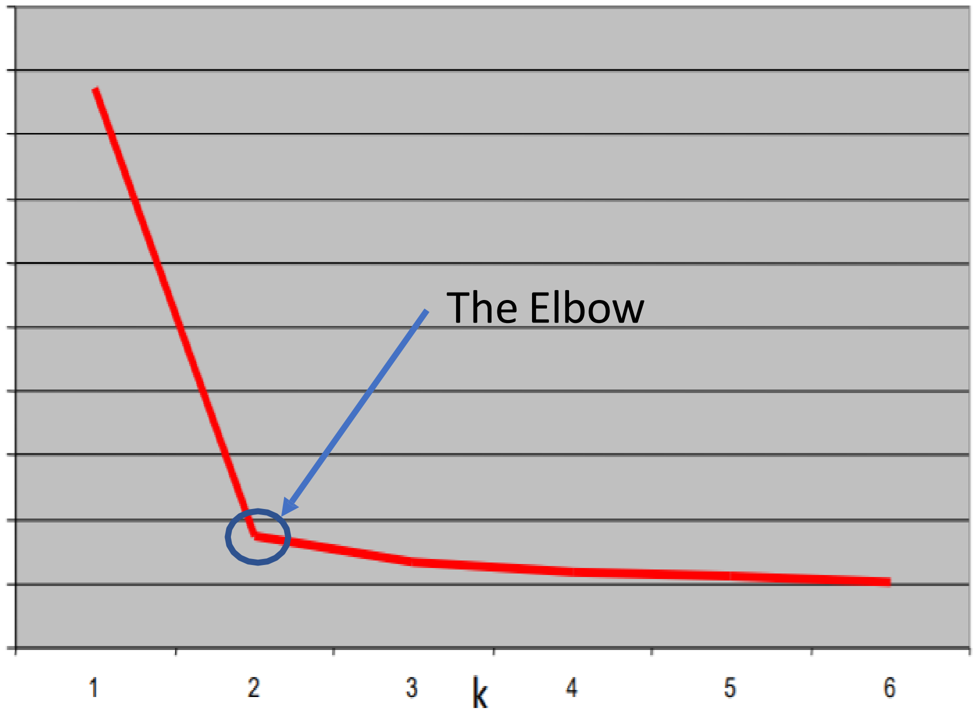
- Initial centroids are chosen at random
- Centroids can be too close



K-Means Summary

How can we choose k ?

- A prior knowledge of the data space can help
  - Three classes of flowers in the Iris dataset
  - Two types of emails: good and spam
- Use the Elbow method
  - Try different values and look for abrupt change in result
- Run hierarchical clustering on subset of data



K-Means Summary

Mitigating Initial Centroids Dependency

- Use a random number seed
- Define a minimum distance $\min(d)$ between clusters:
$$d_1, d_2, d_3 \geq \min(d)$$
- Define the minimum data points in a cluster.