Dimensionality Reduction
&
Latent Variable Modelling

Andreas Damianou

University of Sheffield, UK

Workshop on Data Science in Africa,
17th June, 2015
Dedan Kimathi University of Technology, Kenya

Working with data

- Data-science: everything revolves around a **dataset**

- Dataset: the set of data (to be) collected for our algorithms to learn from

- Example: animals dataset

$$\begin{array}{cc} \begin{array}{cc} \text{height} & \text{weight} \end{array} & \\ \begin{bmatrix} 33 & 20 \\ 40 & 28 \\ 28 & 12 \\ 70 & 36 \\ 15 & 7 \end{bmatrix} & \begin{array}{l} \rightarrow \text{animal 1} \\ \rightarrow \text{animal 2} \\ \rightarrow \text{animal 3} \\ \rightarrow \text{animal 4} \\ \rightarrow \text{animal 5} \end{array} \end{array}$$

Working with data

- Data-science: everything revolves around a **dataset**

- Dataset: the set of data (to be) collected for our algorithms to learn from

- Example: animals dataset

$$Y = \begin{array}{c} \begin{array}{cc} \text{height} & \text{weight} \end{array} \\ \begin{bmatrix} 33 & 20 \\ 40 & 28 \\ 28 & 12 \\ 70 & 36 \\ 15 & 7 \end{bmatrix} \end{array} \begin{array}{l} \\ \rightarrow \text{animal 1} \\ \rightarrow \text{animal 2} \\ \rightarrow \text{animal 3} \\ \rightarrow \text{animal 4} \\ \rightarrow \text{animal 5} \end{array}$$

Working with data

- Data-science: everything revolves around a **dataset**
- Dataset: the set of data (to be) collected for our algorithms to learn from
- Example: animals dataset

5 rows (instances)
2 columns (<u>features</u>)

$$\begin{array}{ccc} & \begin{array}{cc}\text{height} & \text{weight}\end{array} & \\ Y = & \begin{bmatrix} h_1 & w_1 \\ h_2 & w_2 \\ h_3 & w_3 \\ h_4 & w_4 \\ h_5 & w_5 \end{bmatrix} & \begin{array}{c} \rightarrow y_1 \\ \rightarrow y_2 \\ \rightarrow y_3 \\ \rightarrow y_4 \\ \rightarrow y_5 \end{array} \end{array}$$

What is “dimensionality”?

- Simply, the number of features *used* for describing each instance.
- In the previous example: height and width.
- The number of features depends on our selection or limitations during data collection.

Notation

It's convenient to use notation from linear algebra.

$$Y = \begin{bmatrix} y_{1,1} & y_{1,2} & \cdots & y_{1,d} \\ y_{2,1} & y_{2,2} & \cdots & y_{2,d} \\ \vdots & \vdots & & \vdots \\ y_{n,1} & y_{n,2} & \cdots & y_{n,d} \end{bmatrix}$$

n rows → *n* instances

d columns → *d* features (dimensions)

So, the matrix Y containsd-dimensional data

High-dimensional data

- Data having large number of features, *d*
- Examples
  - Micro-array data
  - Images

...

$$Y = \begin{bmatrix} y_{1,1} & y_{1,2} & \cdots & y_{1,d} \\ y_{2,1} & y_{2,2} & \cdots & y_{2,d} \\ \vdots & \vdots & & \vdots \\ y_{n,1} & y_{n,2} & \cdots & y_{n,d} \end{bmatrix}$$

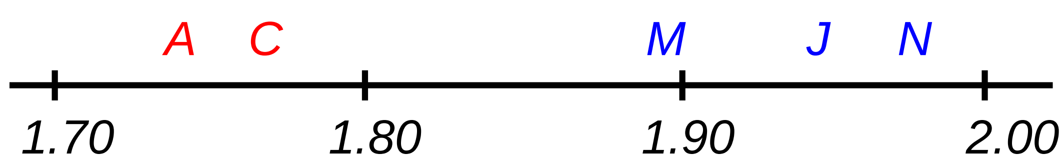
Properties of high-dimensional data

Properties of high-dimensional data

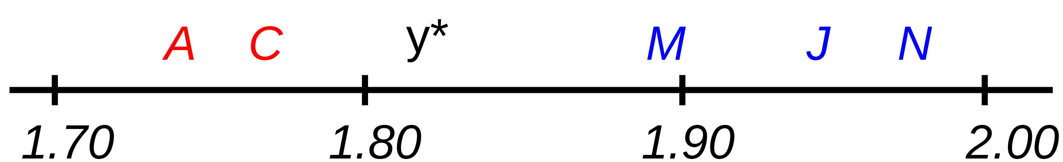
Non-British people

British people

Name	Height
Neil:	1.97
John:	1.94
Mike:	1.89
Ciira:	1.76
Andreas:	1.74

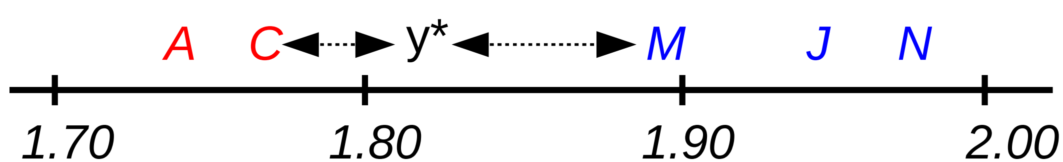


Name	Height
Neil:	1.97
John:	1.94
Mike:	1.89
Ciira:	1.76
Andreas:	1.74



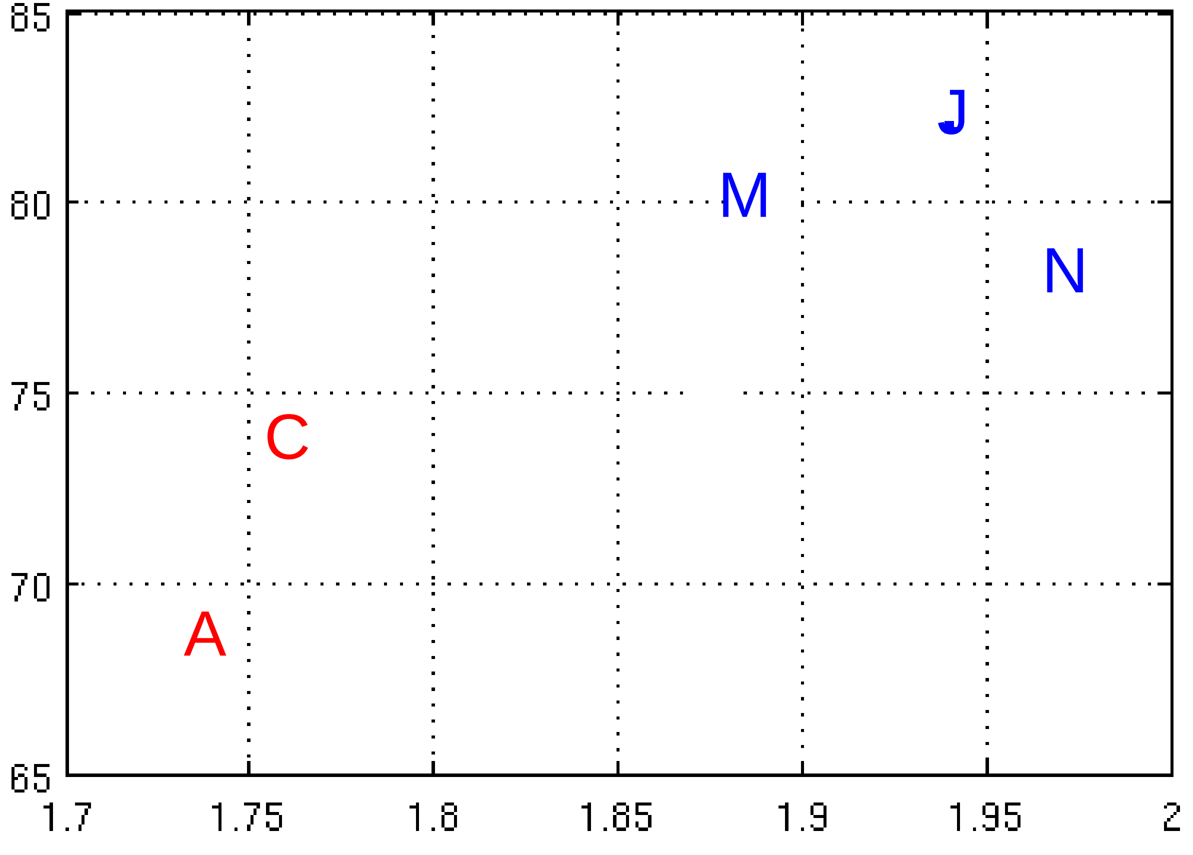
Name	Height
Neil:	1.97
John:	1.94
Mike:	1.89
Ciira:	1.76
Andreas:	1.74

Nearest neighbour classification for a new instance y*



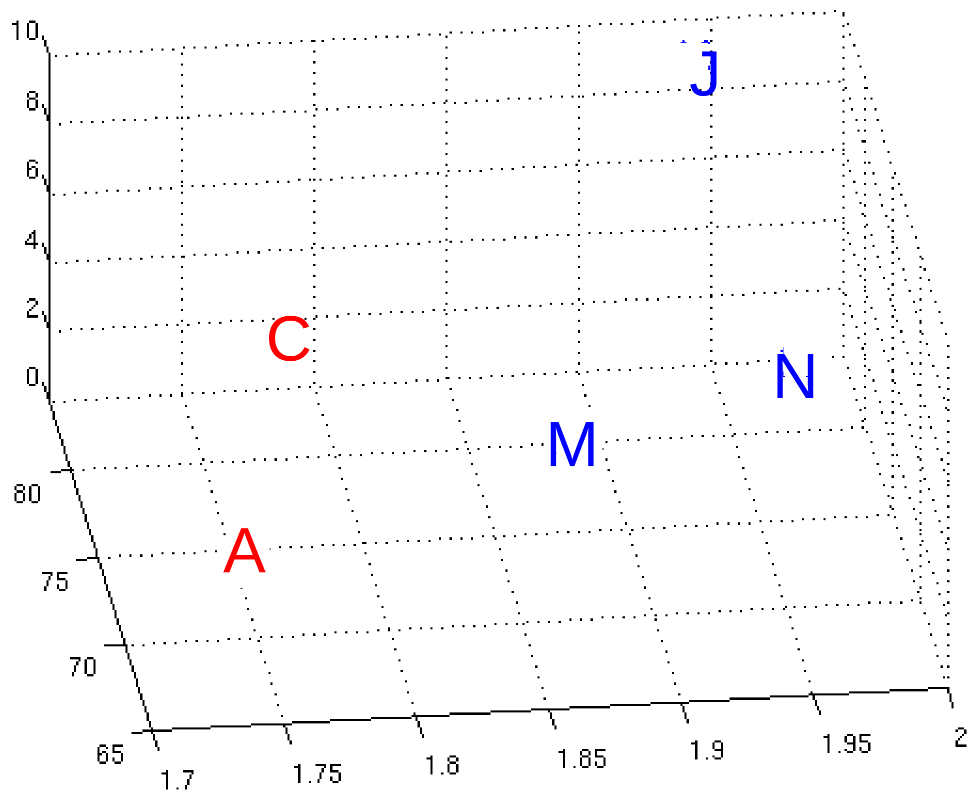
Distance from C to M: sqrt((C_h – M_h)^2) = 0.0009

Adding another feature: weight



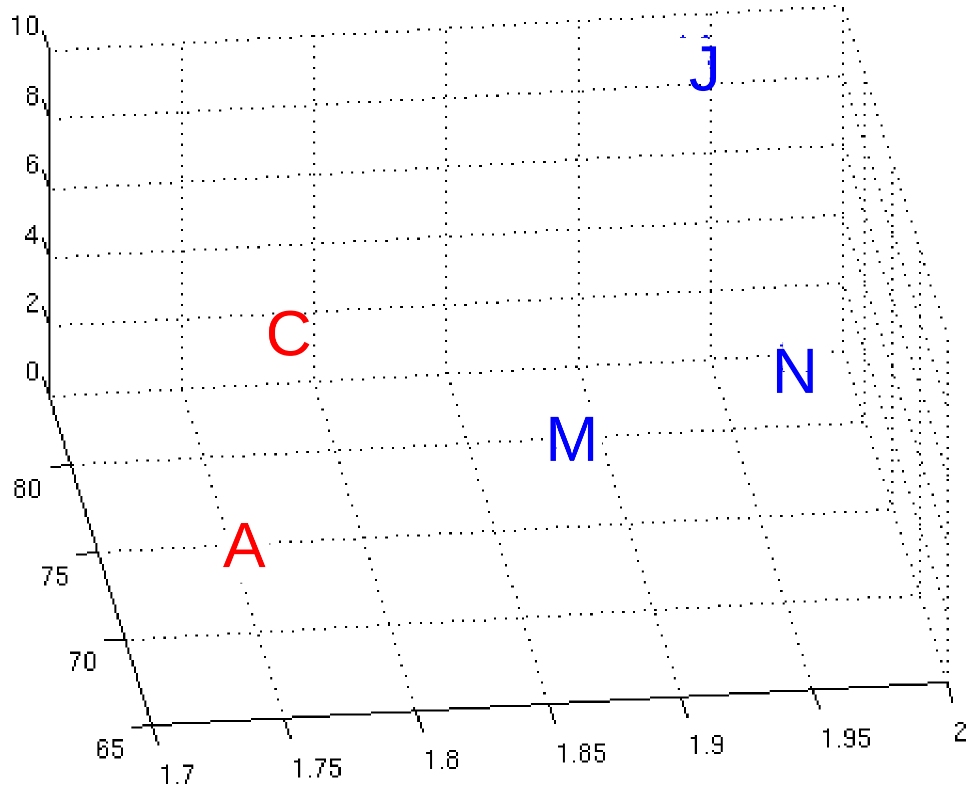
Distance from C to M: sqrt((C_h – M_h)^2 + (C_w – M_w)^2) = 16

Adding another feature: “beardness”



Distance from C to M: sqrt((C_h – M_h)^2 + (C_w – M_w)^2 + (C_b – M_b)^2) = 65

Adding another feature: “beardness”



Distance from C to M: sqrt((C_h – M_h)^2 + (C_w – M_w)^2 + (C_b – M_b)^2) = 65

Show demo!! ***(humanClassif)***

The (potential) problem

- What happens as the dimensionality grows?
  -
  -
- Why is that a problem?
  -
- Is that always a problem?
  -

The (potential) problem

- What happens as the dimensionality grows?
  - Distances grow bigger
  - Everything seems to be spread apart
- Why is that a problem?
  -
- Is that always a problem?
  -

The (potential) problem

- What happens as the dimensionality grows?
  - Distances grow bigger
  - Everything seems to be spread apart
- Why is that a problem?
  - Think about doing nearest neighbour classification. For a test instance, everything would just be super far...
- Is that always a problem?
  -

The (potential) problem

- What happens as the dimensionality grows?
  - Distances grow bigger
  - Everything seems to be spread apart
- Why is that a problem?
  - Think about doing nearest neighbour classification. For a test instance, everything would just be super far...
- Is that always a problem?
  - No, but with real, “dirty” data, it can often be

The (potential) problem

- What happens as the dimensionality grows?
  - Distances grow bigger
  - Everything seems to be spread apart
- Why is that a problem?
  - Think about doing nearest neighbour classification. For a test instance, everything would just be super far...
- Is that always a problem?
  - No, but with real, “dirty” data, it can often be
  -
- Another problem we'll see later: *noise in the data*

Why do dimensionality reduction?

Why do dimensionality reduction?

- Pre-process data for another task (e.g. classification)
- Compression (lossy)
- Visualisation
- Data understanding / clearing

Why do dimensionality reduction?

- Pre-process data for another task (e.g. classification)
- Compression (lossy)
- Visualisation
- Data understanding / clearing

Compression

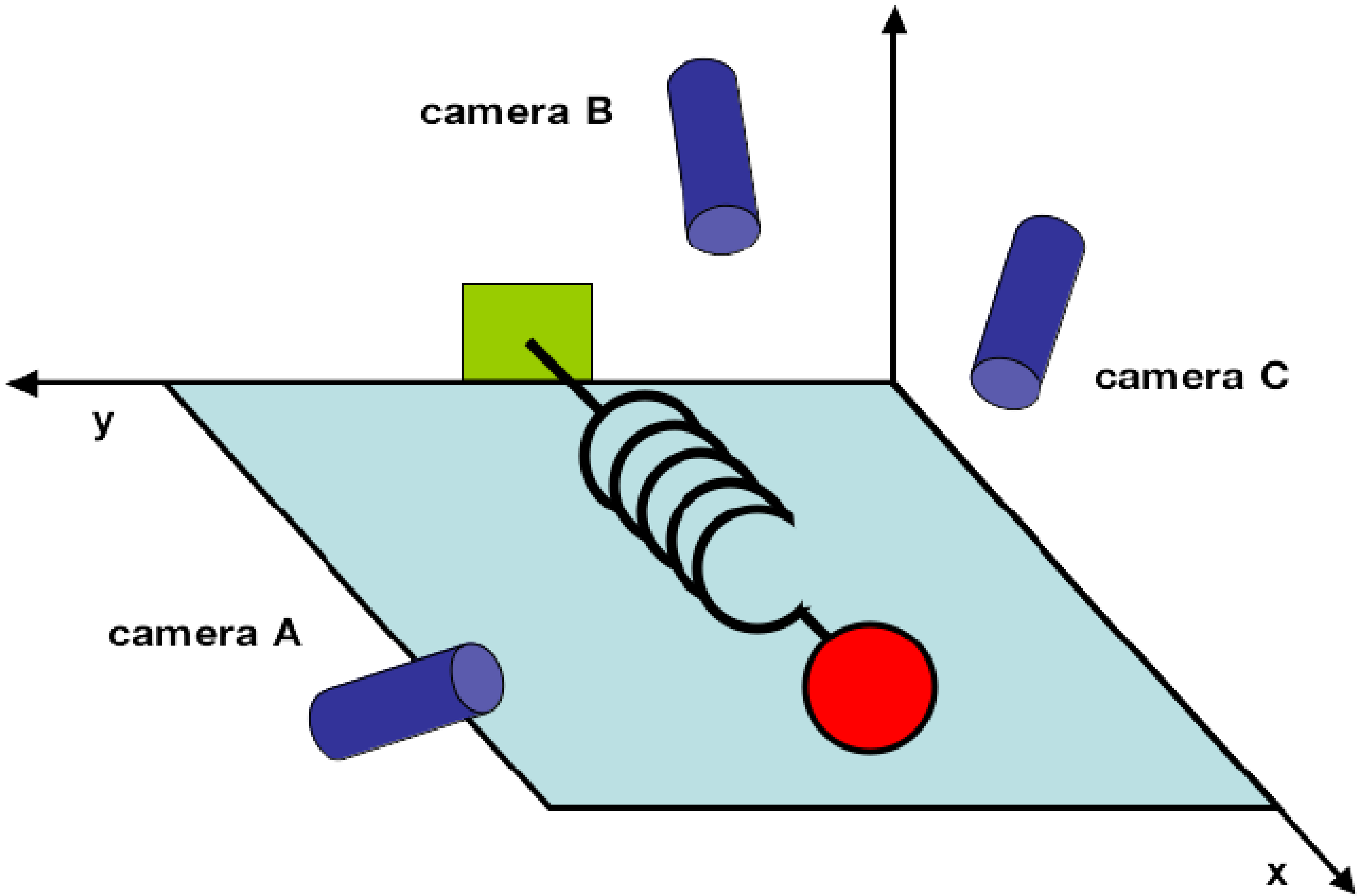
$$Y = \begin{bmatrix} h_1 & w_1 & h_1 + w_1 \\ h_2 & w_2 & h_2 + w_2 \\ h_3 & w_3 & h_3 + w_3 \\ h_4 & w_4 & h_4 + w_4 \\ h_5 & w_5 & h_5 + w_5 \end{bmatrix}$$

redundant!!

Why do dimensionality reduction?

- Pre-process data for another task (e.g. classification)
- Compression (lossy)
- Visualisation
- Data understanding / clearing

Spring-ball example



On the board!

Clustering vs Dimensionality Reduction

n x d matrix

Clustering

m x d matrix

$m < n$

Dimensionality Reduction

n x k matrix

$k < d$

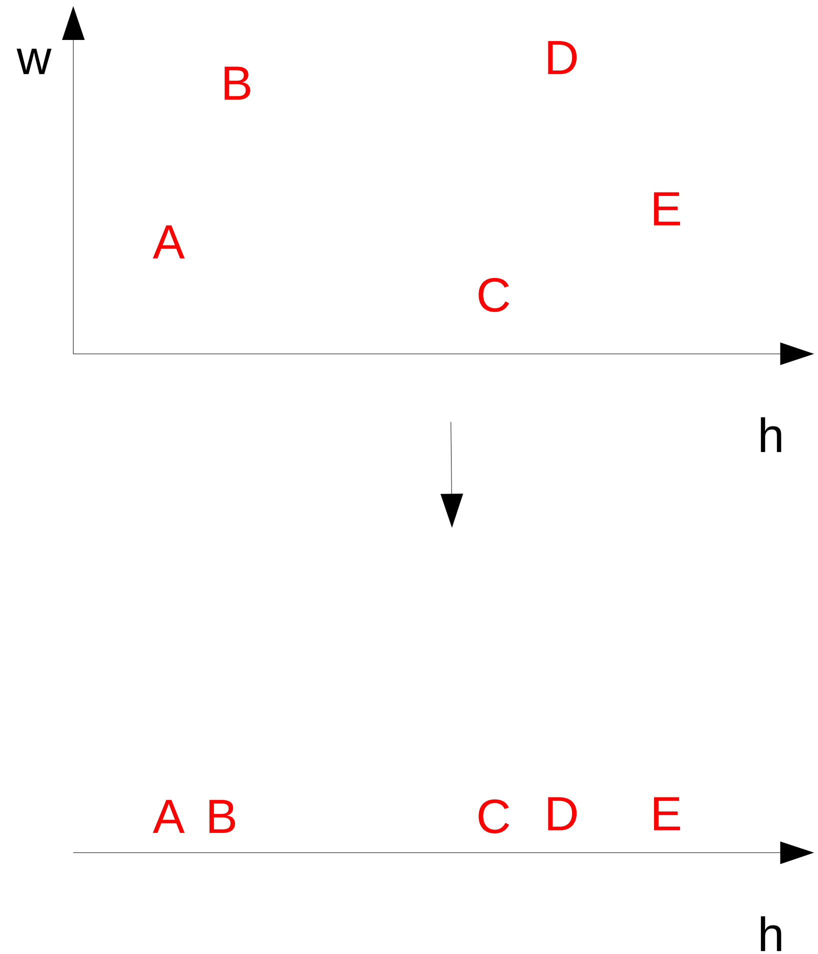
Dimensionality reduction in action

1st way: One solution is to just drop one of the features

$$Y = \begin{array}{cc} \text{height} & \text{weight} \\ \hline \end{array}$$

$$Y = \begin{array}{c} \begin{array}{cc} \text{height} & \text{weight} \end{array} \\ \begin{bmatrix} h_1 & w_1 \\ h_2 & w_2 \\ h_3 & w_3 \\ h_4 & w_4 \\ h_5 & w_5 \end{bmatrix} \end{array}$$

$$X' = \begin{array}{c} \text{height} \\ \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \end{bmatrix} \end{array}$$



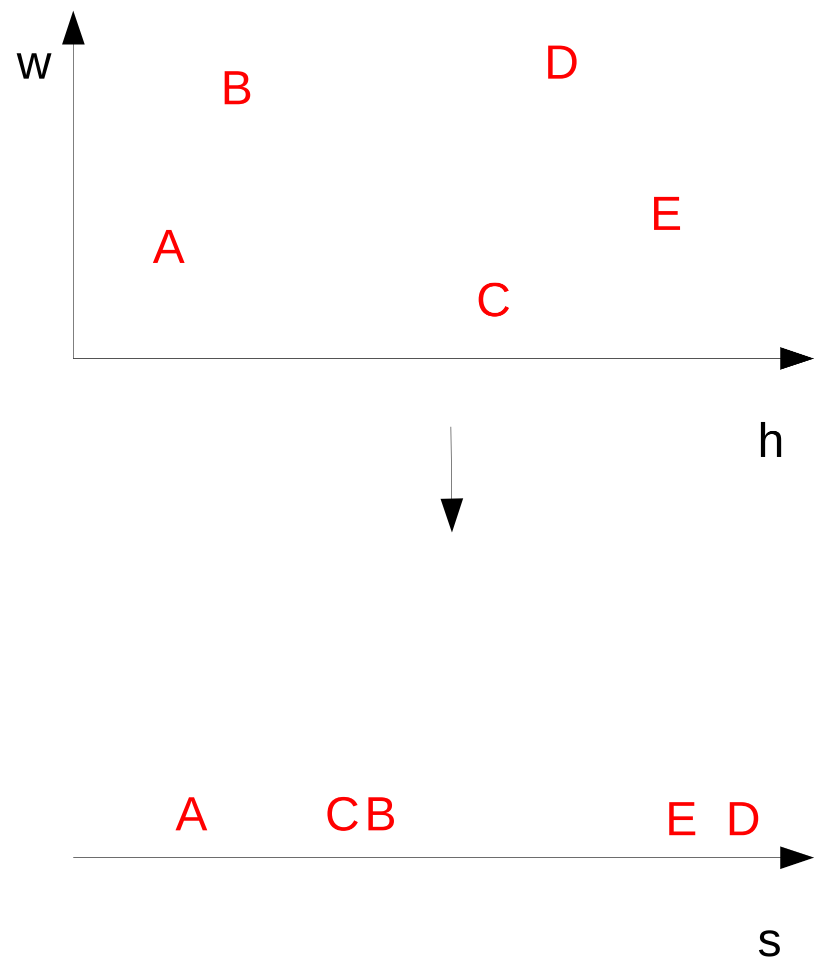
Dimensionality reduction in action

2nd way: Another solution is to transform the two features into one

$$
\begin{array}{cc}
 & \begin{array}{cc} \text{height} & \text{weight} \end{array} \\
Y = & \begin{bmatrix} h_1 & w_1 \\ h_2 & w_2 \\ h_3 & w_3 \\ h_4 & w_4 \\ h_5 & w_5 \end{bmatrix}
\end{array}
$$

$$
\begin{array}{cc}
 & \text{size} \\
X'' = & \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{bmatrix}
\end{array}
$$

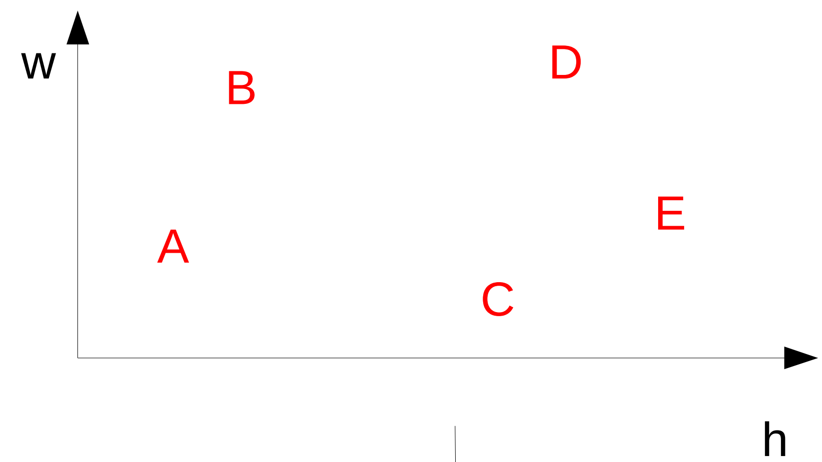
$s = f(h,w)$



Dimensionality reduction in action

Another solution is to transform the two features to one

$$Y = \begin{array}{c} \begin{array}{cc} \text{height} & \text{weight} \end{array} \\ \begin{bmatrix} h_1 & w_1 \\ h_2 & w_2 \\ h_3 & w_3 \\ h_4 & w_4 \\ h_5 & w_5 \end{bmatrix} \end{array}$$



<u>But how do we find s?</u>

$$X'' = \begin{array}{c} \text{size} \\ \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \end{bmatrix} \end{array}$$

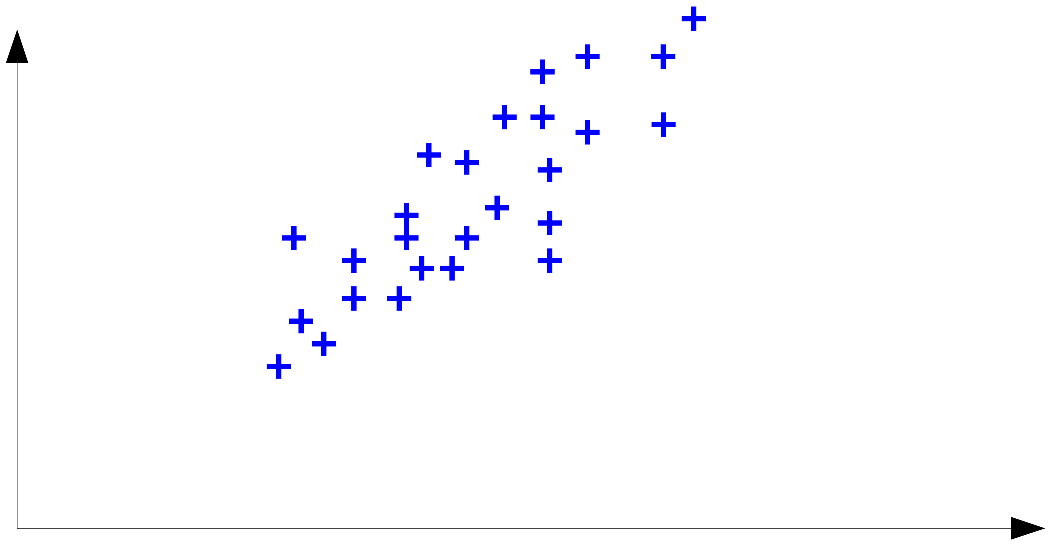
$s = f(h,w)$

Principal Component Analysis

Run demo!! (pcaPlot)

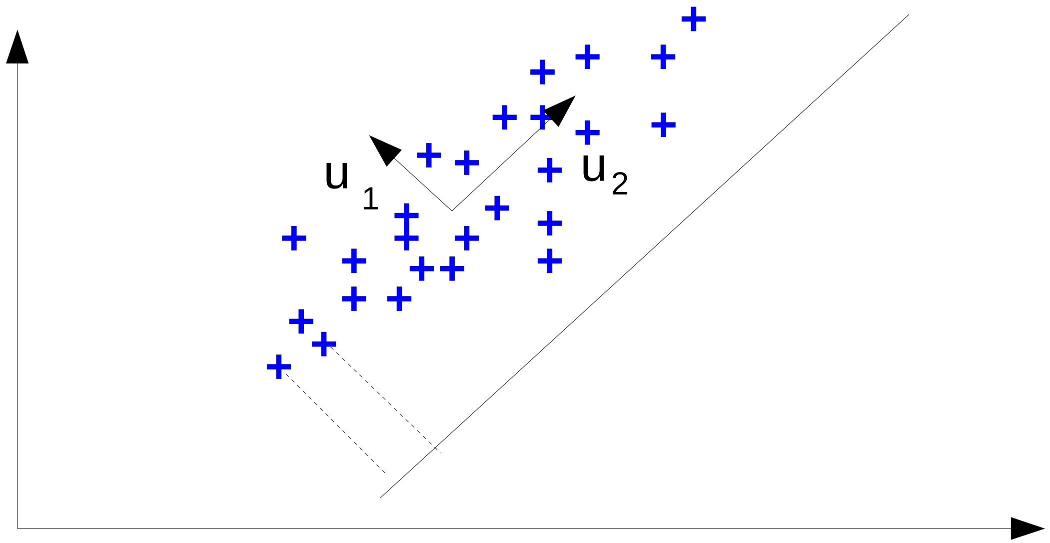
Eigendecomposition

- Gives a feeling of the properties of the matrix



Eigendecomposition

- Gives a feeling of the properties of the matrix



- u1 and u2 define the axes with maximum variances, where the data is most spread
- To reduce the dimensionality I project the data on the axis where data is the most spread
- There is no class information given

- We need to optimise in order to minimise the distance between the true point and its reconstrcution:

$$x^* = \underset{x}{\operatorname{argmin}} \; || \; y - rec(x) \; ||_2$$

y is *1 times d*
x is *1 times q*

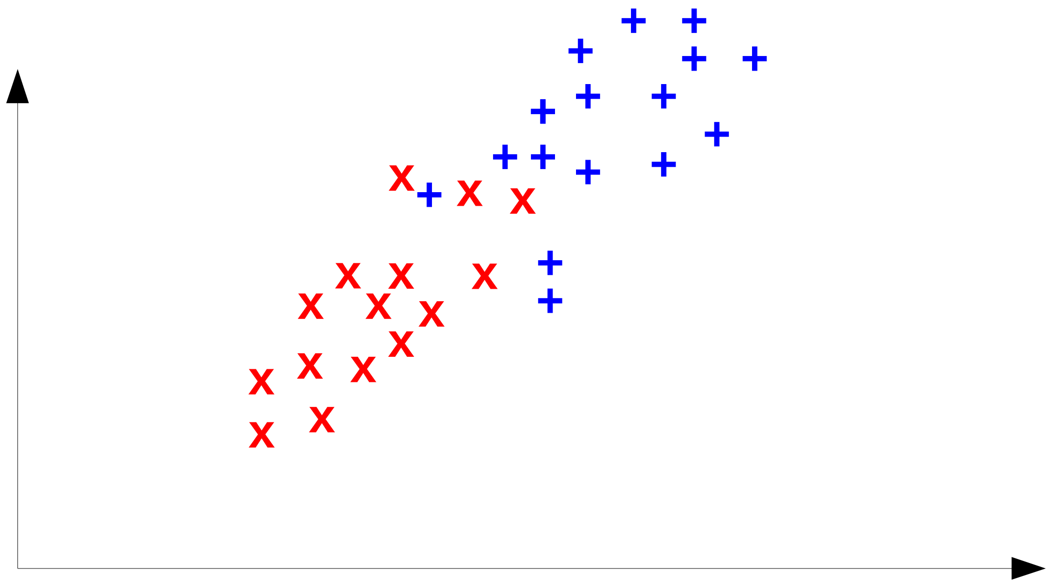
- *rec(x) = x W'*

- What is the dimensionality of *W'*?
- Answer: *d times q*

- Then: *x = y W.* Now W is q times d.

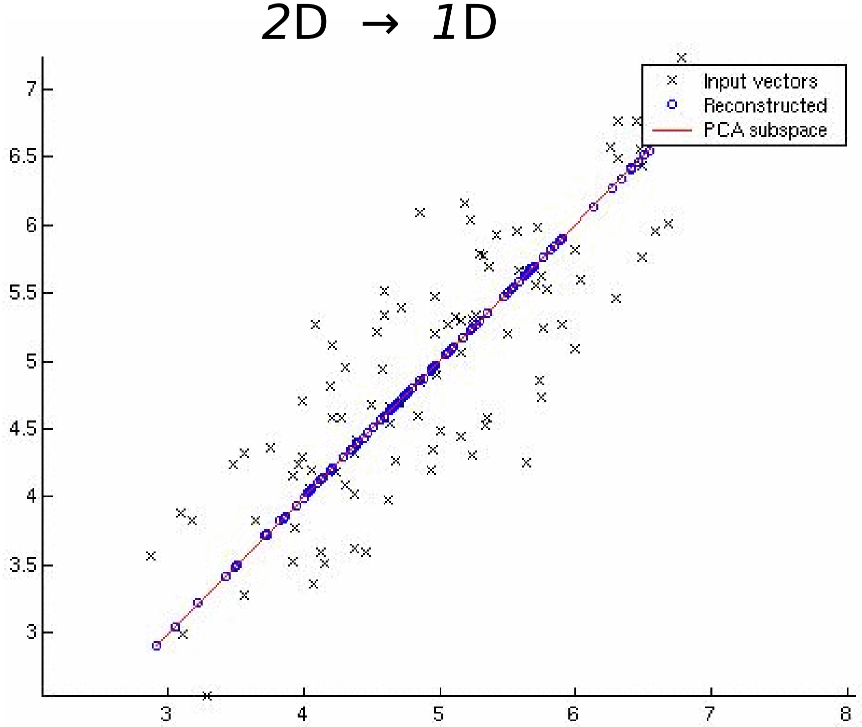
- We can determine both *x* and *W* by solving an optimisation problem (called eigenvalue problem)

- When class labels are given, PCA cannot take them into account... but we hope that the natural separation of the data (see clustering) will encode this information

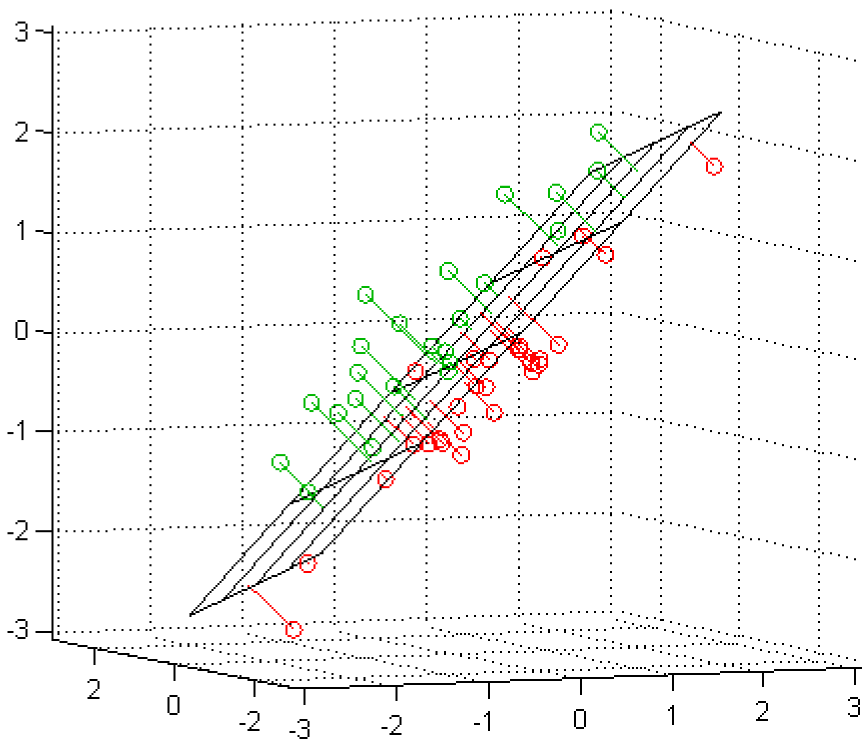


Principal Component Analysis

*2*D → *1*D

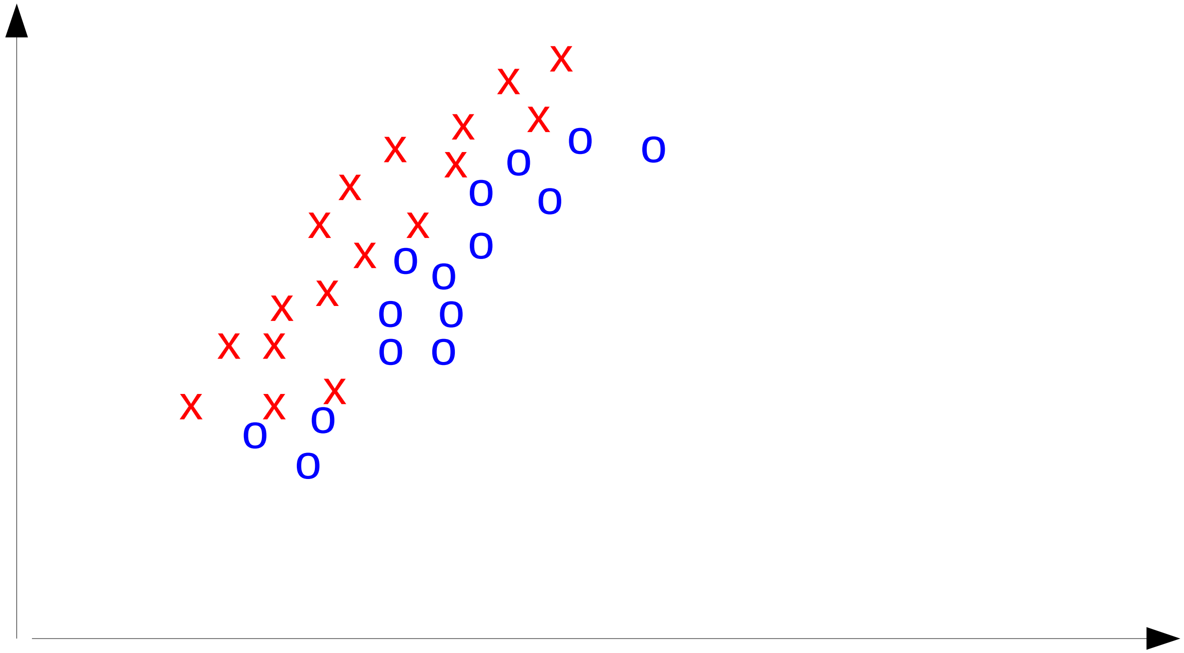


*3*D → *2*D

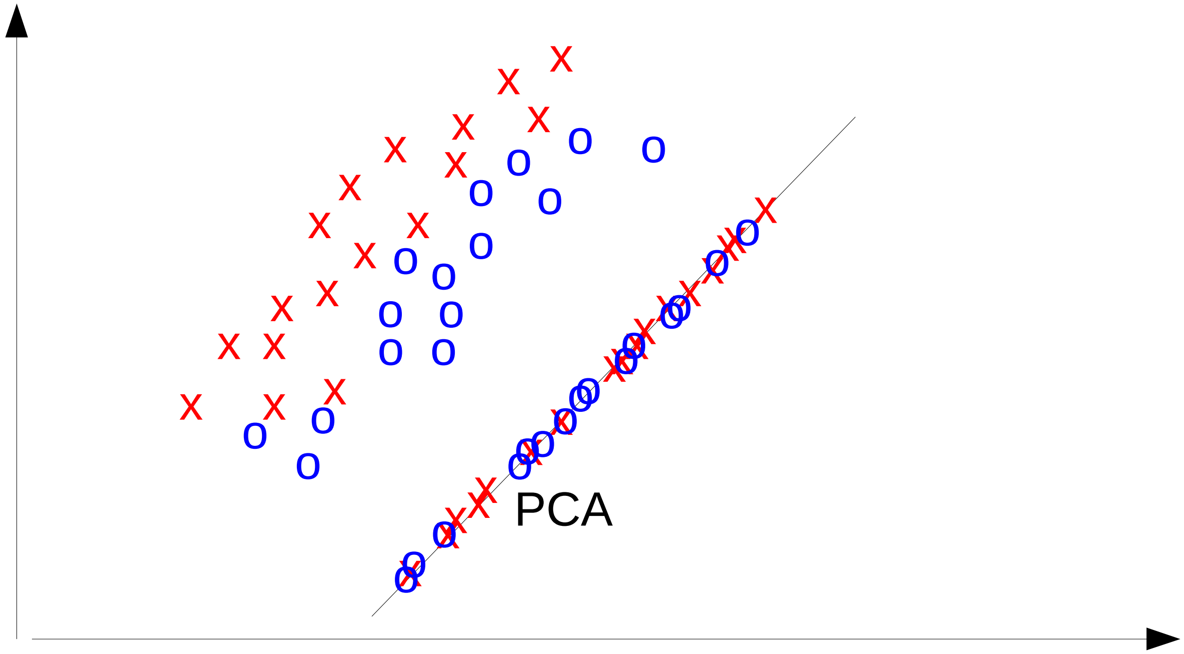


- Remember: data are noisy!
- Trade-off: reduce size / noise without losing too much information

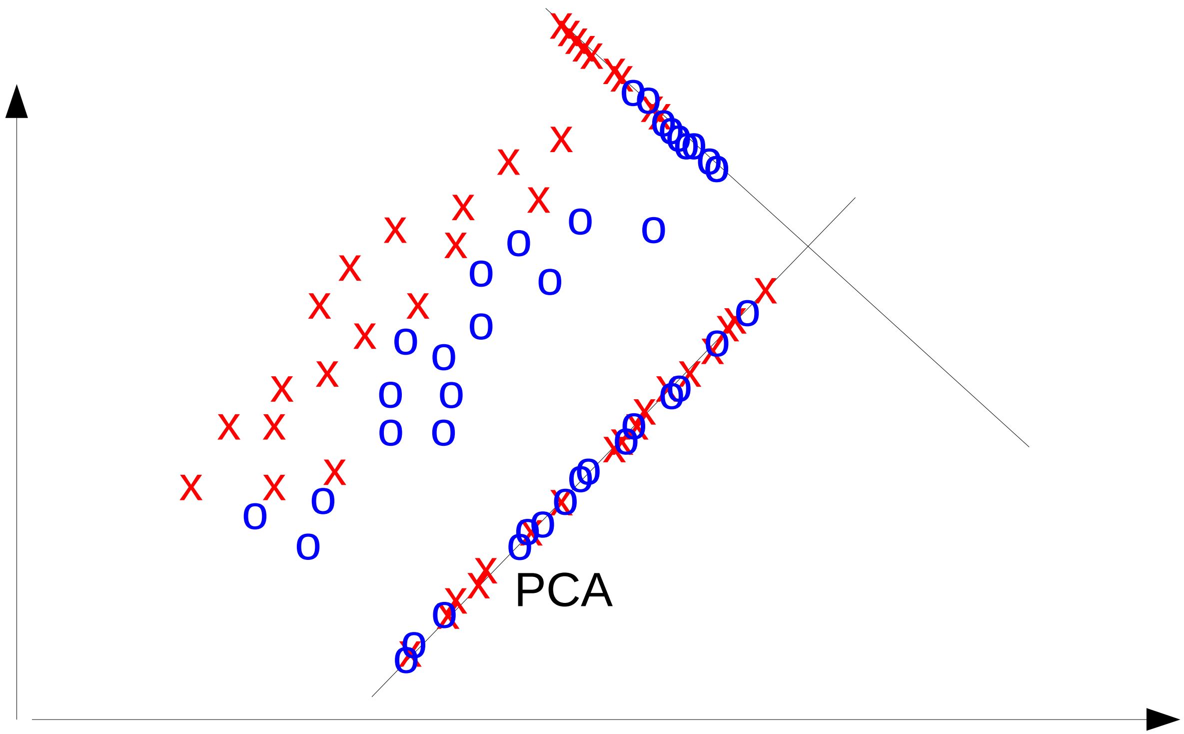
But what about this case...



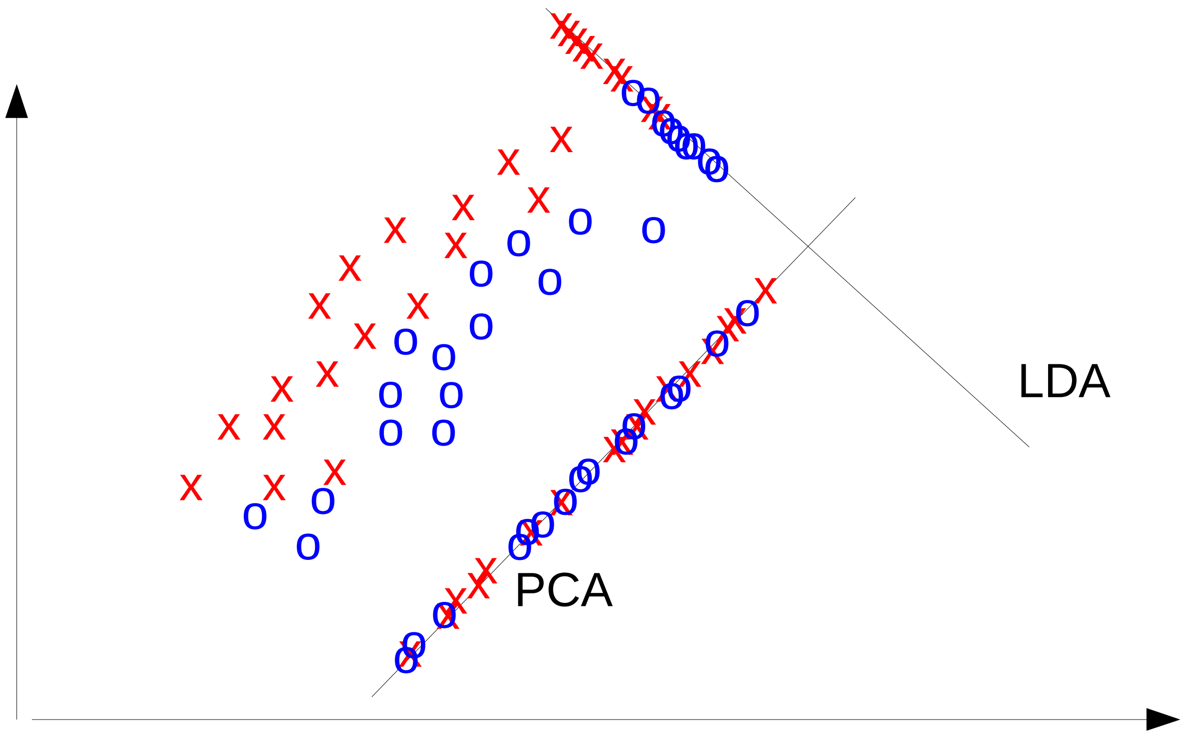
But what about this case...



But what about this case...



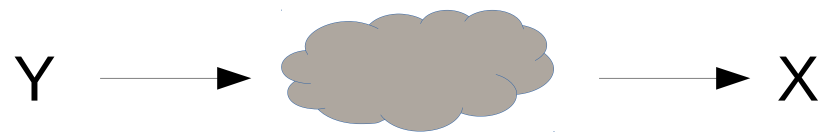
But what about this case...



In **discriminant analysis**, we want to maximise the <u>spread between classes.</u>

Latent variable models

- Non-probabilistic approach:



- Probabilistic approach:

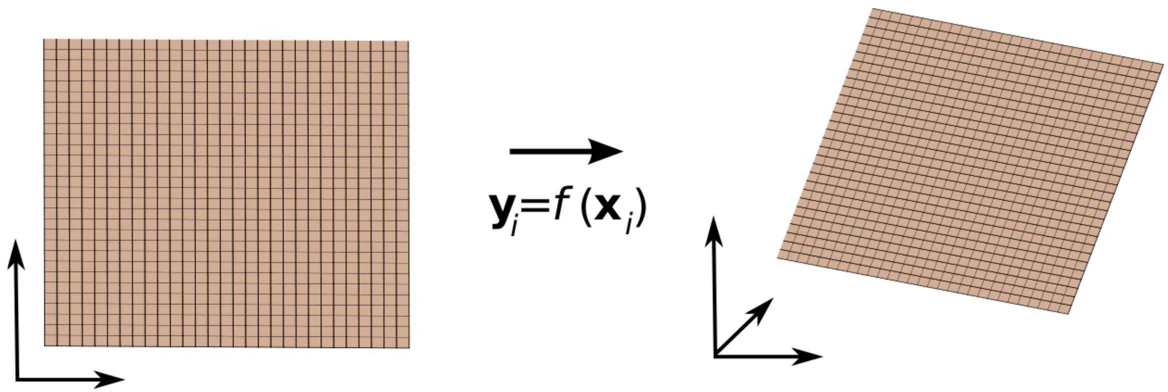
  Learn "backwards": <u>Model</u> the relationship between Y and X:

  $p(Y|X)$. This comes from: $y = wx + \varepsilon$

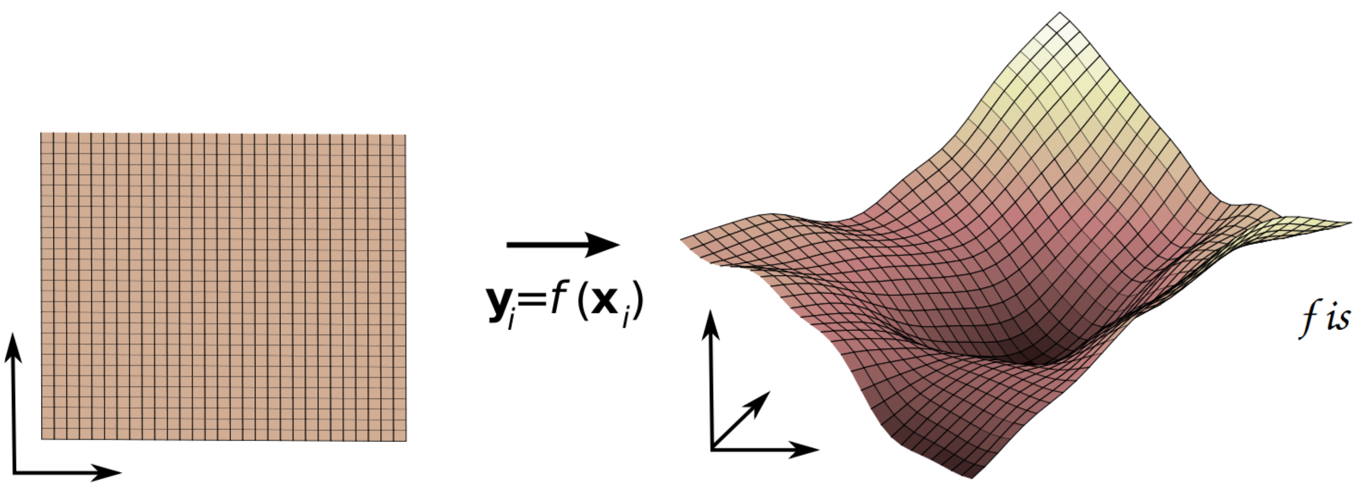
  Then, we can get the posterior distribution:

  $p(X|Y)$

Linear vs Non-linear



f is linear



f is non-linear

(Slide from Neil Lawrence.)