Week 8 - Dimensionality Reduction II

COM2004/3004

Jon Barker

Department of Computer Science
University of Sheffield

Autumn Semester

Lecture Objectives

In this lecture we will,

- introduce the idea of a dimensionality reducing transform.
- introduce three types of transform,
  - a fixed transform – e.g. the Discrete Cosine Transform
  - a data dependent transform – e.g. principal component transform
  - a data and class dependent transform – e.g. linear discriminant analysis

Lecture Objectives

In this lecture we will,

- introduce the idea of a dimensionality reducing transform.
- introduce three types of transform,
  - a fixed transform – e.g. the Discrete Cosine Transform
  - a data dependent transform – e.g. principal component transform
  - a data and class dependent transform – e.g. linear discriminant analysis

Lecture Objectives

In this lecture we will,

- introduce the idea of a dimensionality reducing transform.
- introduce three types of transform,
  - a fixed transform – e.g. the Discrete Cosine Transform
  - a data dependent transform – e.g. principal component transform
  - a data and class dependent transform – e.g. linear discriminant analysis

Lecture Objectives

In this lecture we will,

- introduce the idea of a dimensionality reducing transform.
- introduce three types of transform,
  - a fixed transform – e.g. the Discrete Cosine Transform
  - a data dependent transform – e.g. principal component transform
  - a data and class dependent transform – e.g. linear discriminant analysis

Lecture Objectives

In this lecture we will,

- introduce the idea of a dimensionality reducing transform.
- introduce three types of transform,
  - a fixed transform – e.g. the Discrete Cosine Transform
  - a data dependent transform – e.g. principal component transform
  - a data and class dependent transform – e.g. linear discriminant analysis

How can we reduce dimensionality of $\mathbf{x}$

Consider our face data.

- Select some subset of elements, e.g. keep just a line of pixels down the center of the image
  - Will loose information.
  - How do we select which pixels to keep...?
- Use feature selection techniques like those discussed last week
  - But are any individual pixels likely to discriminate between classes?
  - People can't be identified by looking at individual pixels.
  - Need to find features that are less 'local'.
- Note the high degree of correlation between the features
  - Note, adjacent pixels tend to have similar values – they are correlated
  - A pair of correlated features hold less information that a pair of independent features
  - Intuitively, the 'effective' dimensionality of the face images in less than 17×17

How can we reduce dimensionality of $\mathbf{x}$

Consider our face data.

- Select some subset of elements, e.g. keep just a line of pixels down the center of the image
  - Will loose information.
  - How do we select which pixels to keep. . . ?
- Use feature selection techniques like those discussed last week
  - But are any individual pixels likely to discriminate between classes?
  - People can't be identified by looking at individual pixels.
  - Need to find features that are less 'local'.
- Note the high degree of correlation between the features
  - Note, adjacent pixels tend to have similar values – they are correlated
  - A pair of correlated features hold less information that a pair of independent features
  - Intuitively, the 'effective' dimensionality of the face images in less than 17×17

How can we reduce dimensionality of $\mathbf{x}$

Consider our face data.

- Select some subset of elements, e.g. keep just a line of pixels down the center of the image
  - Will loose information.
  - How do we select which pixels to keep...?
- Use feature selection techniques like those discussed last week
  - But are any individual pixels likely to discriminate between classes?
  - People can't be identified by looking at individual pixels.
  - Need to find features that are less 'local'.
- Note the high degree of correlation between the features
  - Note, adjacent pixels tend to have similar values – they are correlated
  - A pair of correlated features hold less information that a pair of independent features
  - Intuitively, the 'effective' dimensionality of the face images in less than 17×17

Dimensionality reducing transforms

- Consider some function, H that takes our feature vector x and returns a vector of lower dimensionality y
  - $y = H(x)$ where $\mathbf{x} = \{x_1, x_2, ..., x_N\}$ and $\mathbf{y} = \{y_1, y_2, ..., y_M\}$ and $M < N$.
- We will consider class of functions, H, known as linear transforms.
  - $y_1 = a_{11}x_1 + a_{12}x_2 + ... + a_{1N}x_N$;
  - $y_2 = a_{21}x_1 + a_{22}x_2 + ... + a_{2N}x_N$;
  - $y_M = a_{M1}x_1 + a_{M2}x_2 + ... + a_{MN}x_N$;
- These equations can be written more compactly as,
  - $\mathbf{y} = A\mathbf{x}$ where A is the M by N matrix of parameters a_{ij}

Dimensionality reducing transforms

- Consider some function, H that takes our feature vector x and returns a vector of lower dimensionality y
  - $y = H(x)$ where $\mathbf{x} = \{x_1, x_2, ..., x_N\}$ and $\mathbf{y} = \{y_1, y_2, ..., y_M\}$ and $M < N$.
- We will consider class of functions, H, known as linear transforms.
  - $y_1 = a_{11}x_1 + a_{12}x_2 + ... + a_{1N}x_N;$
  - $y_2 = a_{21}x_1 + a_{22}x_2 + ... + a_{2N}x_N;$
  - $y_M = a_{M1}x_1 + a_{M2}x_2 + ... + a_{MN}x_N;$
- These equations can be written more compactly as,
  - $\mathbf{y} = A\mathbf{x}$ where A is the M by N matrix of parameters a_{ij}

Dimensionality reducing transforms

- Consider some function, H that takes our feature vector x and returns a vector of lower dimensionality y
  - $y = H(x)$ where $\mathbf{x} = \{x_1, x_2, ..., x_N\}$ and $\mathbf{y} = \{y_1, y_2, ..., y_M\}$ and $M < N$.
- We will consider class of functions, H, known as linear transforms.
  - $y_1 = a_{11}x_1 + a_{12}x_2 + ... + a_{1N}x_N$;
  - $y_2 = a_{21}x_1 + a_{22}x_2 + ... + a_{2N}x_N$;
  - $y_M = a_{M1}x_1 + a_{M2}x_2 + ... + a_{MN}x_N$;
- These equations can be written more compactly as,
  - $\mathbf{y} = A\mathbf{x}$ where A is the M by N matrix of parameters a_{ij}

Dimensionality reducing transforms

- Note, feature selection can be seen as a linear transform.
- Special case where for y_i one a_{ij} is 1 and all others are 0.
- For example, consider we are reducing our letter images down to a 3-d feature vector by choosing pixels 20, 145 and 179, then,
  - $y_1 = 0x_1 + 0x_2 + ... + 1x_{20} ... + 0x_{900}$;
  - $y_2 = 0x_1 + 0x_2 + ... + 1x_{145} ... + 0x_{900}$;
  - $y_3 = 0x_1 + 0x_2 + ... + 1x_{179} ... + 0x_{900}$;
- Or $\mathbf{y} = A\mathbf{x}$ with A having 3 rows and 900 column, all 0's except for 1's at $\{1, 20\}$ $\{2, 245\}$ and $\{3, 179\}$
- **Question**: Can we design *better* dimensionality reducing transforms by allowing the matrix A to have an arbitrary form?

Dimensionality reducing transforms

- Note, feature selection can be seen as a linear transform.
- Special case where for y_i one a_{ij} is 1 and all others are 0.
- For example, consider we are reducing our letter images down to a 3-d feature vector by choosing pixels 20, 145 and 179, then,
  - $y_1 = 0x_1 + 0x_2 + ... + 1x_{20} ... + 0x_{900}$;
  - $y_2 = 0x_1 + 0x_2 + ... + 1x_{145} ... + 0x_{900}$;
  - $y_3 = 0x_1 + 0x_2 + ... + 1x_{179} ... + 0x_{900}$;
- Or $\mathbf{y} = A\mathbf{x}$ with A having 3 rows and 900 column, all 0's except for 1's at $\{1, 20\}$ $\{2, 245\}$ and $\{3, 179\}$
- **Question**: Can we design *better* dimensionality reducing transforms by allowing the matrix A to have an arbitrary form?

Dimensionality reducing transforms

- Note, feature selection can be seen as a linear transform.
- Special case where for y_i one a_{ij} is 1 and all others are 0.
- For example, consider we are reducing our letter images down to a 3-d feature vector by choosing pixels 20, 145 and 179, then,
  - $y_1 = 0x_1 + 0x_2 + ... + 1x_{20} ... + 0x_{900};$
  - $y_2 = 0x_1 + 0x_2 + ... + 1x_{145} ... + 0x_{900};$
  - $y_3 = 0x_1 + 0x_2 + ... + 1x_{179} ... + 0x_{900};$
- Or $\mathbf{y} = A\mathbf{x}$ with A having 3 rows and 900 column, all 0's except for 1's at $\{1, 20\}$ $\{2, 245\}$ and $\{3, 179\}$
- **Question**: Can we design *better* dimensionality reducing transforms by allowing the matrix A to have an arbitrary form?

Dimensionality reducing transforms

- Note, feature selection can be seen as a linear transform.
- Special case where for y_i one a_{ij} is 1 and all others are 0.
- For example, consider we are reducing our letter images down to a 3-d feature vector by choosing pixels 20, 145 and 179, then,
  - $y_1 = 0x_1 + 0x_2 + \ldots + 1x_{20} \ldots + 0x_{900}$;
  - $y_2 = 0x_1 + 0x_2 + \ldots + 1x_{145} \ldots + 0x_{900}$;
  - $y_3 = 0x_1 + 0x_2 + \ldots + 1x_{179} \ldots + 0x_{900}$;
- Or $\mathbf{y} = A\mathbf{x}$ with A having 3 rows and 900 column, all 0's except for 1's at $\{1, 20\}$ $\{2, 245\}$ and $\{3, 179\}$
- **Question**: Can we design *better* dimensionality reducing transforms by allowing the matrix A to have an arbitrary form?

Dimensionality reducing transforms

- Note, feature selection can be seen as a linear transform.
- Special case where for y_i one a_{ij} is 1 and all others are 0.
- For example, consider we are reducing our letter images down to a 3-d feature vector by choosing pixels 20, 145 and 179, then,
  - $y_1 = 0x_1 + 0x_2 + \ldots + 1x_{20} \ldots + 0x_{900}$;
  - $y_2 = 0x_1 + 0x_2 + \ldots + 1x_{145} \ldots + 0x_{900}$;
  - $y_3 = 0x_1 + 0x_2 + \ldots + 1x_{179} \ldots + 0x_{900}$;
- Or $\mathbf{y} = A\mathbf{x}$ with A having 3 rows and 900 column, all 0's except for 1's at $\{1, 20\}$ $\{2, 245\}$ and $\{3, 179\}$
- **Question**: Can we design *better* dimensionality reducing transforms by allowing the matrix A to have an arbitrary form?

What would make a good y?

Some questions that we might consider:

- Is the dimensionality of y much lower than that of the input vector x?
- Does y 'capture the information' that is in x?
- Are the features of y uncorrelated?
- Does y separate out the classes?

What would make a good y?

Some questions that we might consider:

- Is the dimensionality of y much lower than that of the input vector x?
- Does y 'capture the information' that is in x?
- Are the features of y uncorrelated?
- Does y separate out the classes?

What would make a good y?

Some questions that we might consider:

- Is the dimensionality of y much lower than that of the input vector x?
- Does y ‘capture the information’ that is in x?
- Are the features of y uncorrelated?
- Does y separate out the classes?

What would make a good y?

Some questions that we might consider:

- Is the dimensionality of y much lower than that of the input vector x?
- Does y 'capture the information' that is in x?
- Are the features of y uncorrelated?
- Does y separate out the classes?

Three different approaches

- Discrete Cosine Transform
  - A fixed transform which does not depend on the data.
- Principal Component Analysis
  - A data dependent approach but which does not consider class labels.
- Linear Discriminant Analysis
  - A transform which considers both the data and the class labels.

Three different approaches

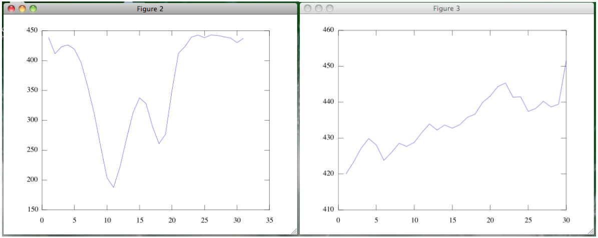
- Discrete Cosine Transform
  - A fixed transform which does not depend on the data.
- Principal Component Analysis
  - A data dependent approach but which does not consider class labels.
- Linear Discriminant Analysis
  - A transform which considers both the data and the class labels.

Three different approaches

- Discrete Cosine Transform
  - A fixed transform which does not depend on the data.
- Principal Component Analysis
  - A data dependent approach but which does not consider class labels.
- Linear Discriminant Analysis
  - A transform which considers both the data and the class labels.

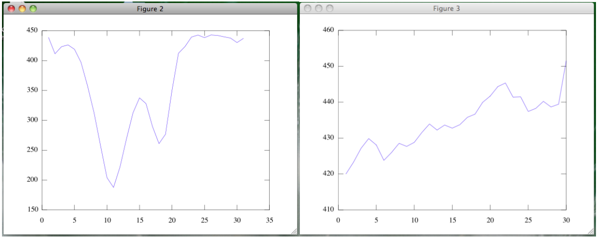
Discrete Cosine Transform

- Consider raw feature vectors made of sequence data
  - temporal sequence - e.g., sound samples, share price history
  - spatial sequence - e.g., pixels in an image
- There are two general observations,
  - adjacent samples in the sequence may be highly correlated
  - rapid fluctuations in the sequence are often uninteresting noise effect
- e.g., consider two rows of pixel data taken from letter 'A'



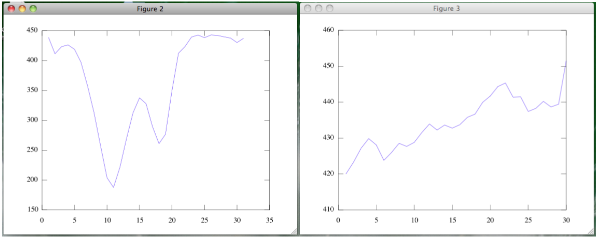
Discrete Cosine Transform

- Consider raw feature vectors made of sequence data
  - temporal sequence - e.g., sound samples, share price history
  - spatial sequence - e.g., pixels in an image
- There are two general observations,
  - adjacent samples in the sequence may be highly correlated
  - rapid fluctuations in the sequence are often uninteresting noise effect
- e.g., consider two rows of pixel data taken from letter 'A'

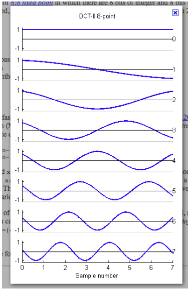


Discrete Cosine Transform

- Consider raw feature vectors made of sequence data
  - temporal sequence - e.g., sound samples, share price history
  - spatial sequence - e.g., pixels in an image
- There are two general observations,
  - adjacent samples in the sequence may be highly correlated
  - rapid fluctuations in the sequence are often uninteresting noise effect
- e.g., consider two rows of pixel data taken from letter 'A'

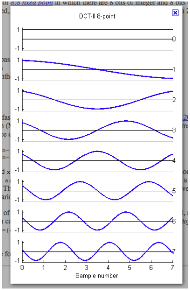


Discrete Cosine Transform

- The Discrete Cosine Transform is a linear transform $\mathbf{y} = A\mathbf{x}$, with
  - $a_{ij} = \cos\left(\frac{\pi}{N}(j + \frac{1}{2})i\right)$
- Can consider the transform as breaking the sequence $\mathbf{x}$ into a weighted sum of a set of basis functions.
  - 

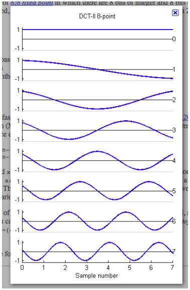
- y_1 how much DC component, y_2 how much tilt, y_3 how much dip in middle etc
- Rapid variations represented by parameters at end of the vector y
- The elements of y tend to be fairly independent.

Discrete Cosine Transform

- The Discrete Cosine Transform is a linear transform $\mathbf{y} = A\mathbf{x}$, with
  - $a_{ij} = \cos\left(\frac{\pi}{N}(j + \frac{1}{2})i\right)$
- Can consider the transform as breaking the sequence $\mathbf{x}$ into a weighted sum of a set of basis functions.
  - 

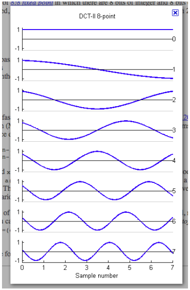
- y_1 how much DC component, y_2 how much tilt, y_3 how much dip in middle etc
- Rapid variations represented by parameters at end of the vector y
- The elements of y tend to be fairly independent.

Discrete Cosine Transform

- The Discrete Cosine Transform is a linear transform $\mathbf{y} = A\mathbf{x}$, with
  - $a_{ij} = \cos\left(\frac{\pi}{N}(j + \frac{1}{2})i\right)$
- Can consider the transform as breaking the sequence $\mathbf{x}$ into a weighted sum of a set of basis functions.
  - 

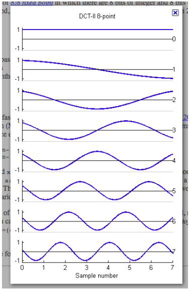
- y_1 how much DC component, y_2 how much tilt, y_3 how much dip in middle etc
- Rapid variations represented by parameters at end of the vector y
- The elements of y tend to be fairly independent.

Discrete Cosine Transform

- The Discrete Cosine Transform is a linear transform $\mathbf{y} = A\mathbf{x}$, with
  - $a_{ij} = \cos\left(\frac{\pi}{N}(j + \frac{1}{2})i\right)$
- Can consider the transform as breaking the sequence $\mathbf{x}$ into a weighted sum of a set of basis functions.
  - 

- y_1 how much DC component, y_2 how much tilt, y_3 how much dip in middle etc
- Rapid variations represented by parameters at end of the vector y
- The elements of y tend to be fairly independent.

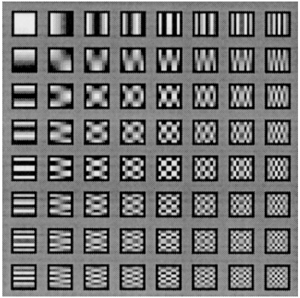
Discrete Cosine Transform

- The Discrete Cosine Transform is a linear transform $\mathbf{y} = A\mathbf{x}$, with
  - $a_{ij} = \cos\left(\frac{\pi}{N}(j + \frac{1}{2})i\right)$
- Can consider the transform as breaking the sequence $\mathbf{x}$ into a weighted sum of a set of basis functions.
  - 

- y_1 how much DC component, y_2 how much tilt, y_3 how much dip in middle etc
- Rapid variations represented by parameters at end of the vector y
- The elements of y tend to be fairly independent.

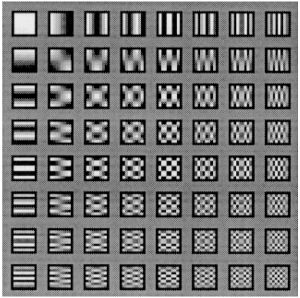
Discrete Cosine Transform

- There is a 2-D form of the DCT that can be used to transform 2-D images
- Equivalent to describing image as a weighted sum of 'basis images' of the form below.

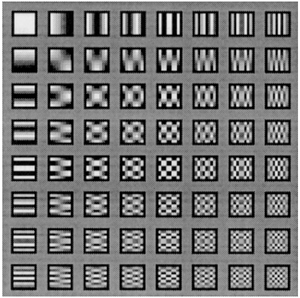


- This is how JPEG image compression works.

Discrete Cosine Transform

- There is a 2-D form of the DCT that can be used to transform 2-D images
- Equivalent to describing image as a weighted sum of 'basis images' of the form below.
  - 
- This is how JPEG image compression works.

Discrete Cosine Transform

- There is a 2-D form of the DCT that can be used to transform 2-D images
- Equivalent to describing image as a weighted sum of ‘basis images’ of the form below.
- 
- This is how JPEG image compression works.

Principal Component Analysis (PCA)

- DCT is a fixed data-independent transform.
- Generally does a good just of decorrelating features and removing irrelevant fine detail.
- Can we do better by tailoring a transform to our particular training data?
- Principal Component Analysis is one approach.
- PCA aims to reduce the dimensionality of the data while preserving its spread.

Principal Component Analysis (PCA)

- DCT is a fixed data-independent transform.
- Generally does a good just of decorrelating features and removing irrelevant fine detail.
- Can we do better by tailoring a transform to our particular training data?
- Principal Component Analysis is one approach.
- PCA aims to reduce the dimensionality of the data while preserving its spread.

Principal Component Analysis (PCA)

- DCT is a fixed data-independent transform.
- Generally does a good just of decorrelating features and removing irrelevant fine detail.
- Can we do better by tailoring a transform to our particular training data?
- Principal Component Analysis is one approach.
- PCA aims to reduce the dimensionality of the data while preserving its spread.

Principal Component Analysis (PCA)

- DCT is a fixed data-independent transform.
- Generally does a good just of decorrelating features and removing irrelevant fine detail.
- Can we do better by tailoring a transform to our particular training data?
- Principal Component Analysis is one approach.
- PCA aims to reduce the dimensionality of the data while preserving its spread.

Principal Component Analysis (PCA)

- DCT is a fixed data-independent transform.
- Generally does a good just of decorrelating features and removing irrelevant fine detail.
- Can we do better by tailoring a transform to our particular training data?
- Principal Component Analysis is one approach.
- PCA aims to reduce the dimensionality of the data while preserving its spread.

PCA – the basic idea

- Consider the act of photographing a 3D object.
- You are reducing a set of 3D points in the real world to a set of 2D points in the image.
- Dimensionality has been reduced and some information has been lost.
- What angle would you choose to photograph the object from in order to preserve as much information as possible?
- Example: photographing a teapot

PCA – the basic idea

- Consider the act of photographing a 3D object.
- You are reducing a set of 3D points in the real world to a set of 2D points in the image.
- Dimensionality has been reduced and some information has been lost.
- What angle would you choose to photograph the object from in order to preserve as much information as possible?
- Example: photographing a teapot

PCA – the basic idea

- Consider the act of photographing a 3D object.
- You are reducing a set of 3D points in the real world to a set of 2D points in the image.
- Dimensionality has been reduced and some information has been lost.
- What angle would you choose to photograph the object from in order to preserve as much information as possible?
- Example: photographing a teapot

PCA – the basic idea

- Consider the act of photographing a 3D object.
- You are reducing a set of 3D points in the real world to a set of 2D points in the image.
- Dimensionality has been reduced and some information has been lost.
- What angle would you choose to photograph the object from in order to preserve as much information as possible?
- Example: photographing a teapot

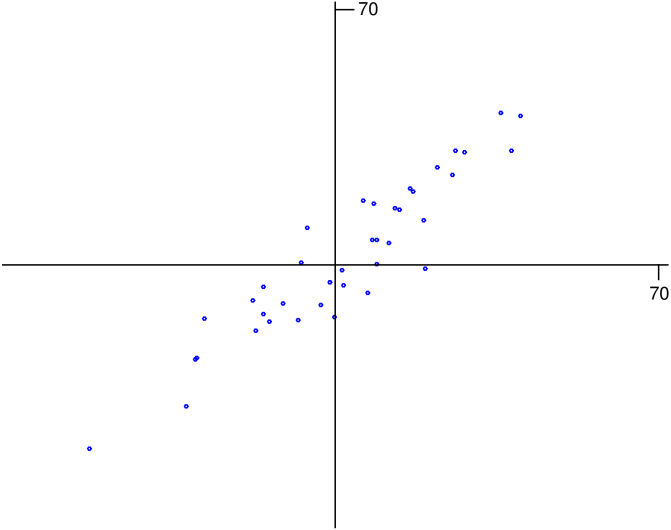
PCA – the basic idea

- Consider the act of photographing a 3D object.
- You are reducing a set of 3D points in the real world to a set of 2D points in the image.
- Dimensionality has been reduced and some information has been lost.
- What angle would you choose to photograph the object from in order to preserve as much information as possible?
- Example: photographing a teapot

Teapot example

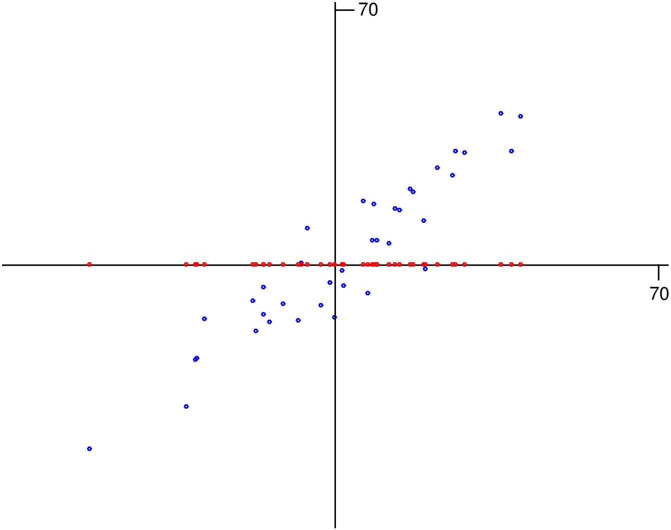
Principal Component Analysis (PCA)

- Consider the following highly correlated data.!!



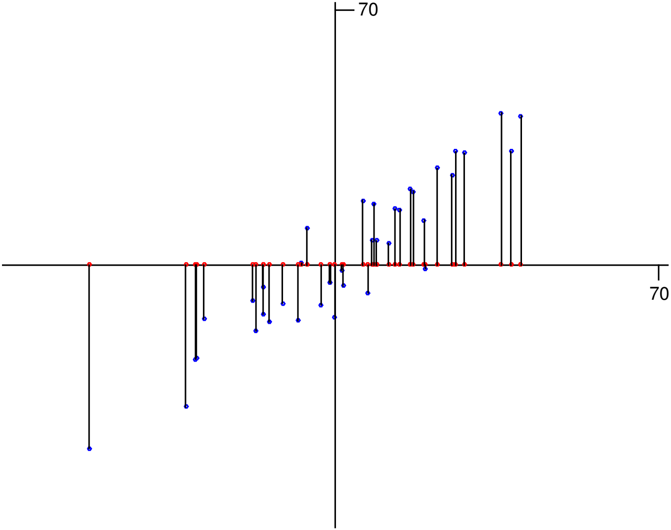
Principal Component Analysis (PCA)

- Consider the following highly correlated data.!!



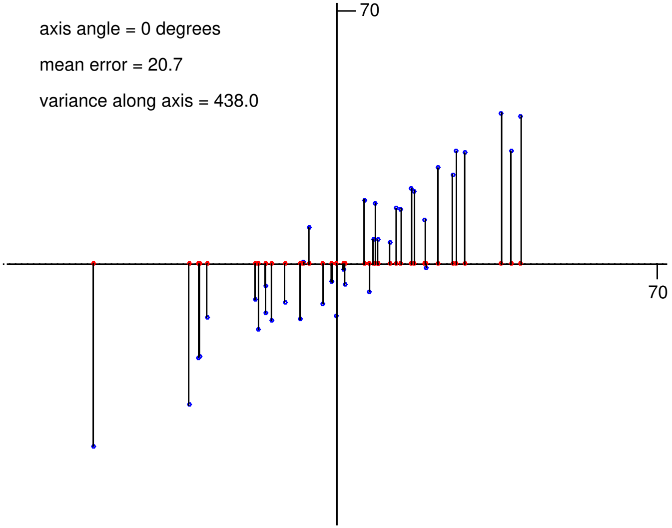
Principal Component Analysis (PCA)

- Consider the following highly correlated data.!!



Principal Component Analysis (PCA)

- Consider the following highly correlated data.!!



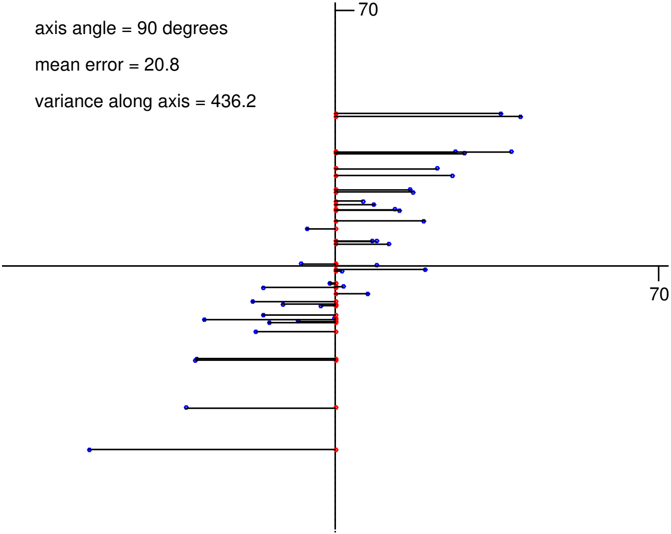
Principal Component Analysis (PCA)

- Consider the following highly correlated data.!!

axis angle = 90 degrees

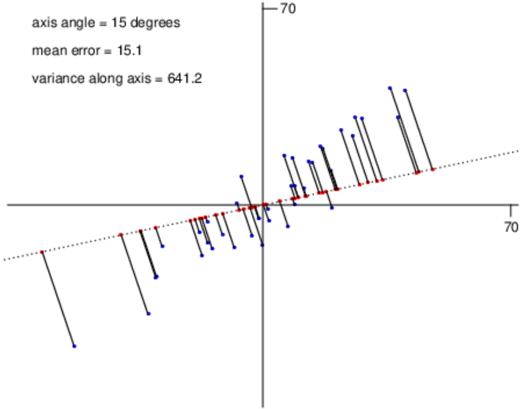
mean error = 20.8

variance along axis = 436.2



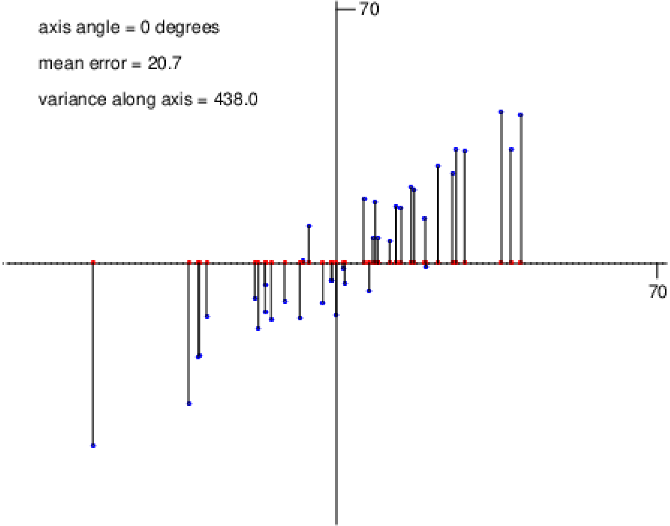
Projecting points onto a new axis

- Points can be 'projected' onto any axis.
- The projection of a point is the point on the axis which lies closest to it.
- Note how the line from a point to its projection meets the axis at right angles.



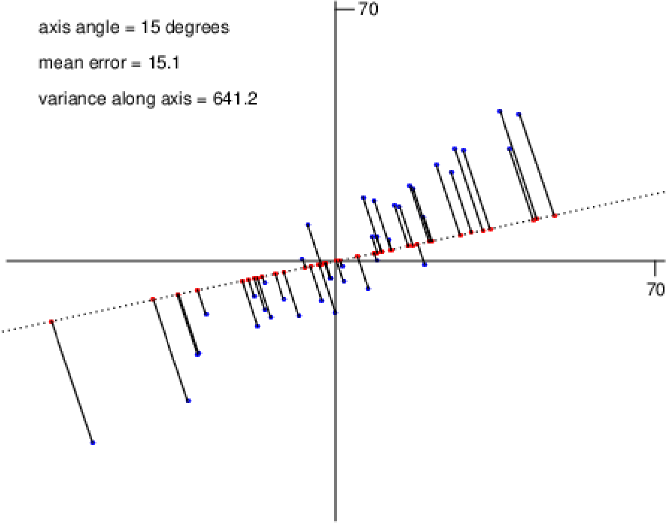
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



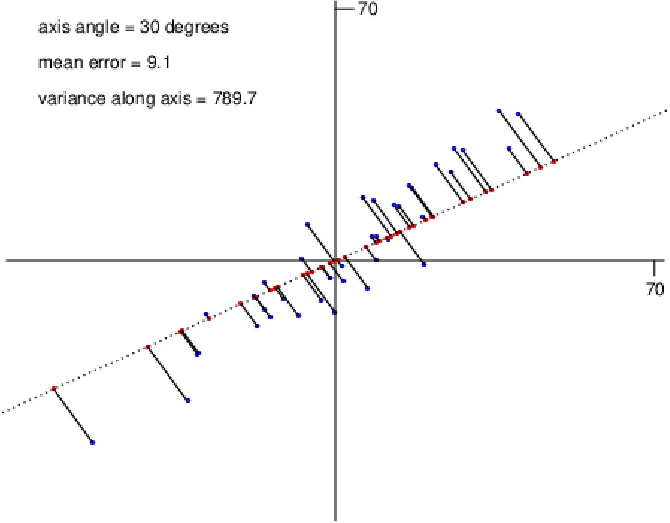
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



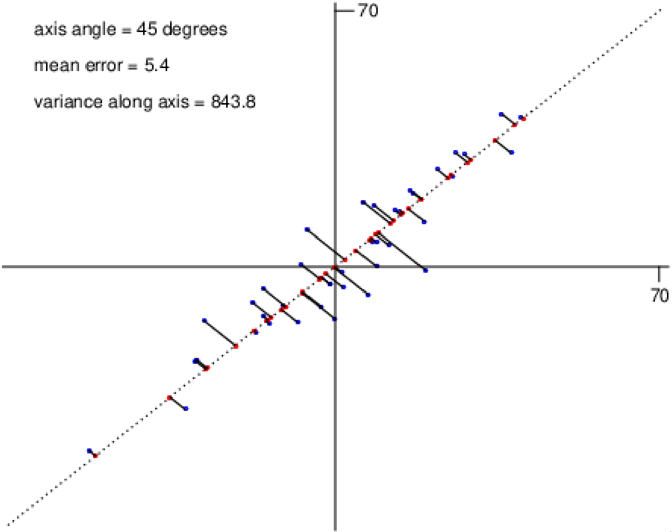
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



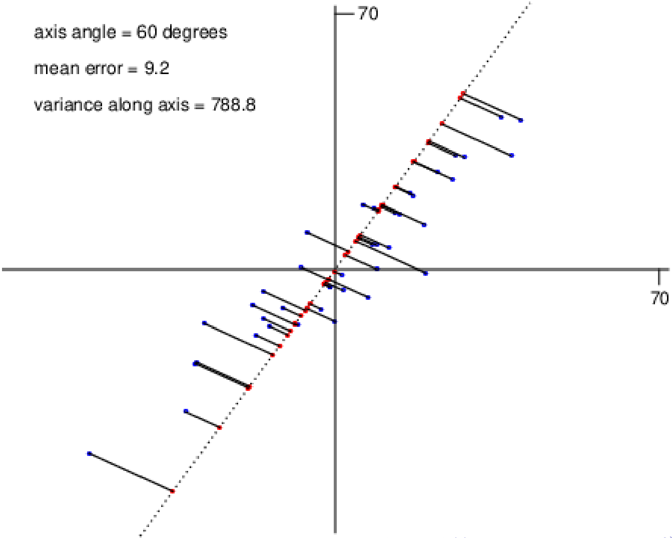
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



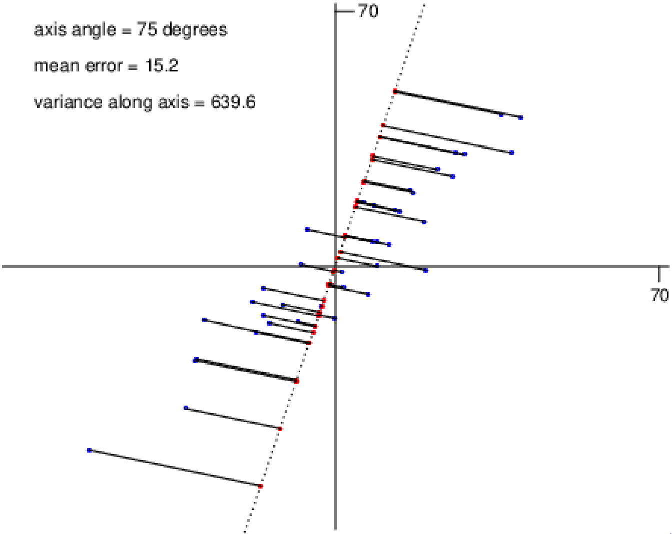
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



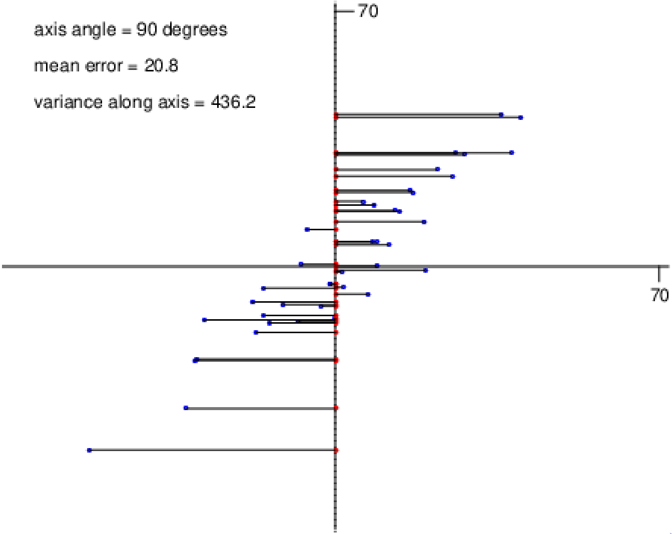
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



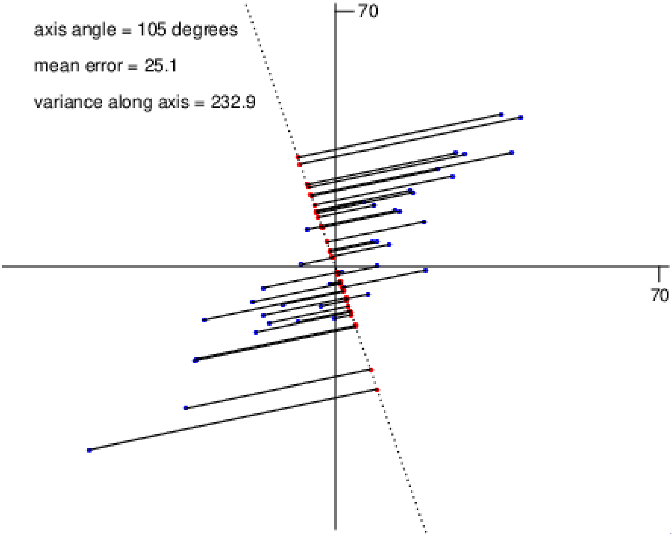
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



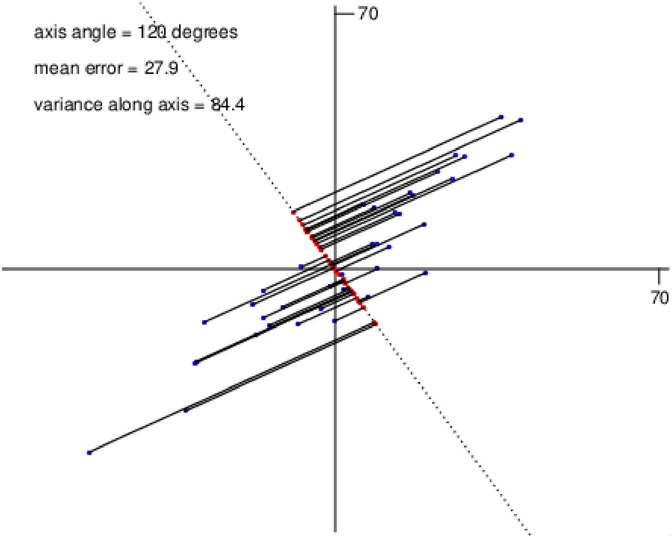
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



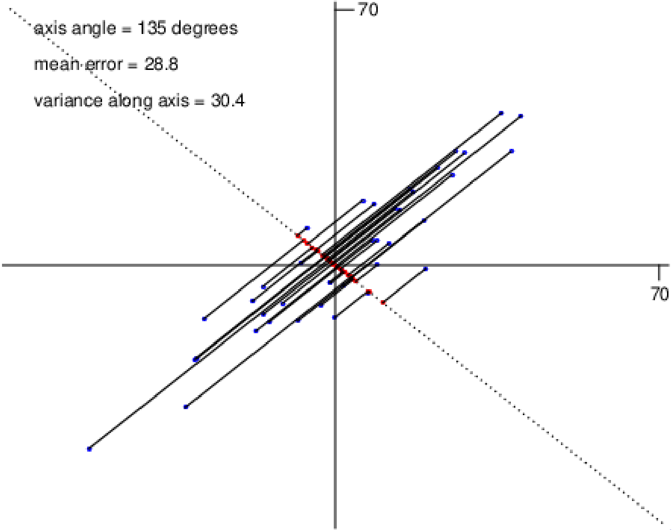
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



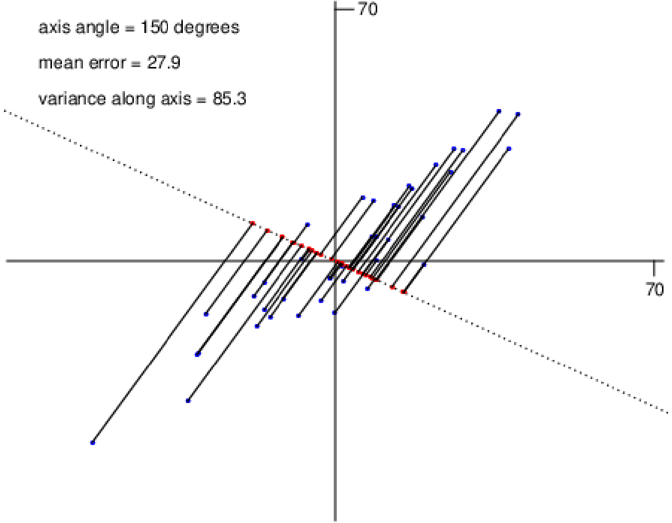
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



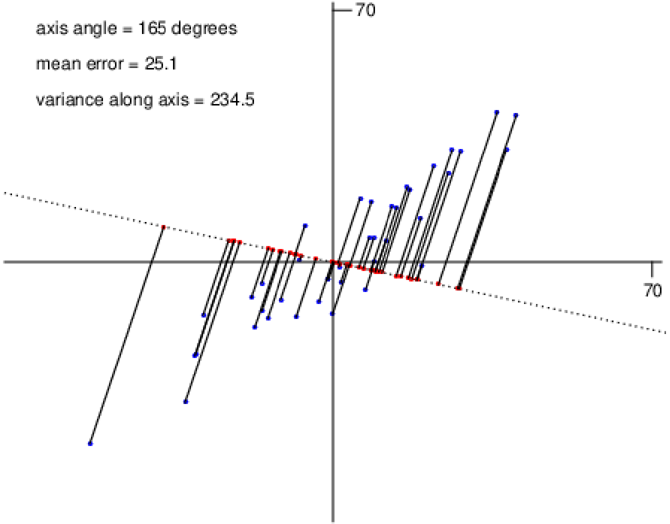
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



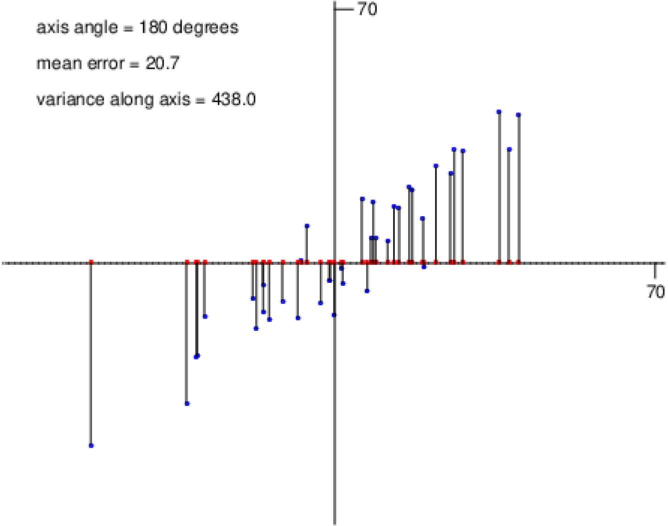
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



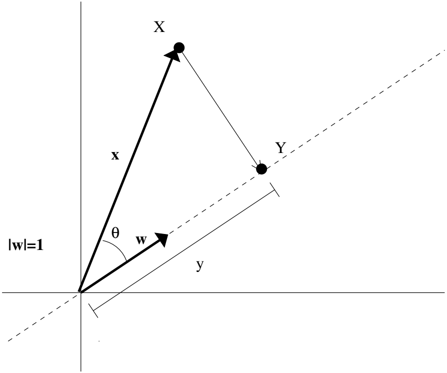
Finding the principal axis

- The principal axis is the one that best represents the data, i.e. projected points lie close to original points.



Position of projected point

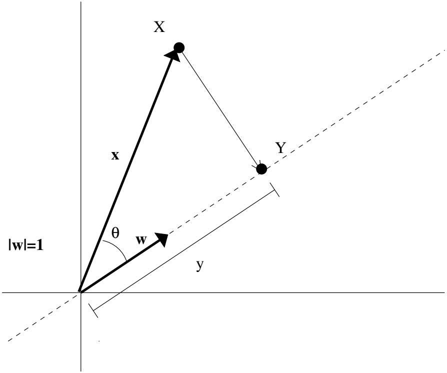
- Given a point X and an axis direction $\mathbf{w}$ we wish to know the distance, y, of the point's projection along the axis.



- From the figure we can see that $y = |\mathbf{x}| \cos\theta$
- Remember, $\mathbf{x}.\mathbf{w} = |\mathbf{x}||\mathbf{w}| \cos\theta$ where θ is the angle between the vectors.
- $\mathbf{w}$ has unit length, so $\mathbf{x}.\mathbf{w} = |\mathbf{x}| \cos\theta = y$
- i.e., $y = \mathbf{x}.\mathbf{w}$, distance y is the dot product of the data point $\mathbf{x}$ and the axis direction vector $\mathbf{w}$

Position of projected point

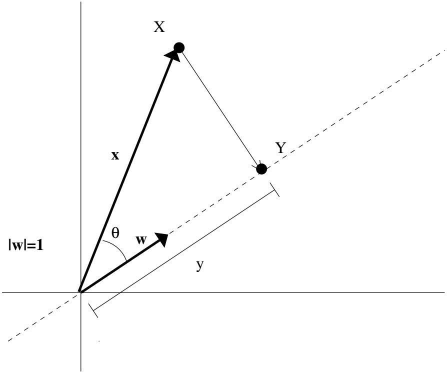
- Given a point X and an axis direction $\mathbf{w}$ we wish to know the distance, y, of the point's projection along the axis.



- From the figure we can see that $y = |\mathbf{x}| \cos\theta$
- Remember, $\mathbf{x}.\mathbf{w} = |\mathbf{x}||\mathbf{w}| \cos\theta$ where θ is the angle between the vectors.
- $\mathbf{w}$ has unit length, so $\mathbf{x}.\mathbf{w} = |\mathbf{x}| \cos\theta = y$
- i.e., $y = \mathbf{x}.\mathbf{w}$, distance y is the dot product of the data point $\mathbf{x}$ and the axis direction vector $\mathbf{w}$

Position of projected point

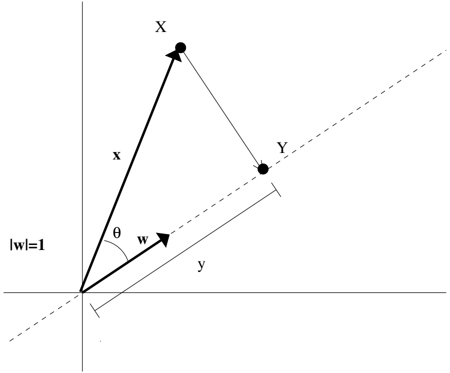
- Given a point X and an axis direction $\mathbf{w}$ we wish to know the distance, y, of the point's projection along the axis.



- From the figure we can see that $y = |\mathbf{x}| \cos\theta$
- Remember, $\mathbf{x}.\mathbf{w} = |\mathbf{x}||\mathbf{w}| \cos\theta$ where θ is the angle between the vectors.
- $\mathbf{w}$ has unit length, so $\mathbf{x}.\mathbf{w} = |\mathbf{x}| \cos\theta = y$
- i.e., $y = \mathbf{x}.\mathbf{w}$, distance y is the dot product of the data point $\mathbf{x}$ and the axis direction vector $\mathbf{w}$

Position of projected point

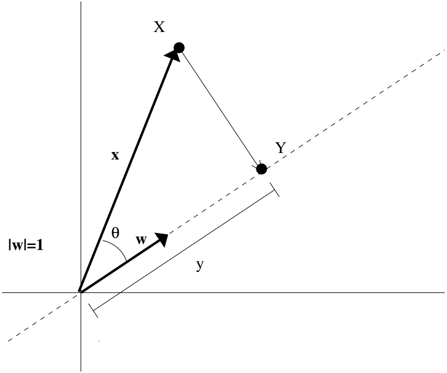
- Given a point X and an axis direction $\mathbf{w}$ we wish to know the distance, y, of the point's projection along the axis.



- From the figure we can see that $y = |\mathbf{x}| \cos\theta$
- Remember, $\mathbf{x}.\mathbf{w} = |\mathbf{x}||\mathbf{w}| \cos\theta$ where θ is the angle between the vectors.
- $\mathbf{w}$ has unit length, so $\mathbf{x}.\mathbf{w} = |\mathbf{x}| \cos\theta = y$
- i.e., $y = \mathbf{x}.\mathbf{w}$, distance y is the dot product of the data point $\mathbf{x}$ and the axis direction vector $\mathbf{w}$

Position of projected point

- Given a point X and an axis direction $\mathbf{w}$ we wish to know the distance, y, of the point's projection along the axis.



- From the figure we can see that $y = |\mathbf{x}| \cos\theta$
- Remember, $\mathbf{x}.\mathbf{w} = |\mathbf{x}||\mathbf{w}| \cos\theta$ where θ is the angle between the vectors.
- $\mathbf{w}$ has unit length, so $\mathbf{x}.\mathbf{w} = |\mathbf{x}| \cos\theta = y$
- i.e., $y = \mathbf{x}.\mathbf{w}$, distance y is the dot product of the data point $\mathbf{x}$ and the axis direction vector $\mathbf{w}$

Principal Component Analysis

- Consider n d-dimensional samples, $\mathbf{x}_1, \ldots, \mathbf{x}_n$,
- Consider, y_i, the coordinate of the projection of these points onto a new axis $\mathbf{w}$

$$y_i = \mathbf{w}^t \mathbf{x}_i$$

- The first principal axis, $\mathbf{w}_1$ is defined as the linear combination $y_i = \mathbf{w}_1^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_1^t \mathbf{w}_1 = 1$.
- i.e. the axis $\mathbf{w}_1$ along which the projected points are most spread out.

Principal Component Analysis

- Consider n d-dimensional samples, $\mathbf{x}_1, \ldots, \mathbf{x}_n$,
- Consider, y_i, the coordinate of the projection of these points onto a new axis $\mathbf{w}$

$$y_i = \mathbf{w}^t \mathbf{x}_i$$

- The first principal axis, $\mathbf{w}_1$ is defined as the linear combination $y_i = \mathbf{w}_1^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_1^t \mathbf{w}_1 = 1$.
- i.e. the axis $\mathbf{w}_1$ along which the projected points are most spread out.

Principal Component Analysis

- Consider n d-dimensional samples, $\mathbf{x}_1, \ldots, \mathbf{x}_n$,
- Consider, y_i, the coordinate of the projection of these points onto a new axis $\mathbf{w}$

$$y_i = \mathbf{w}^t \mathbf{x}_i$$

- The first principal axis, $\mathbf{w}_1$ is defined as the linear combination $y_i = \mathbf{w}_1^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_1^t \mathbf{w}_1 = 1$.
- i.e. the axis $\mathbf{w}_1$ along which the projected points are most spread out.

Principal Component Analysis

- Consider n d-dimensional samples, $\mathbf{x}_1, \ldots, \mathbf{x}_n$,
- Consider, y_i, the coordinate of the projection of these points onto a new axis $\mathbf{w}$

$$y_i = \mathbf{w}^t \mathbf{x}_i$$

- The first principal axis, $\mathbf{w}_1$ is defined as the linear combination $y_i = \mathbf{w}_1^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_1^t \mathbf{w}_1 = 1$.
- i.e. the axis $\mathbf{w}_1$ along which the projected points are most spread out.

Principal Component Analysis

- The first principal axis, $\mathbf{w}_1$ is defined as the linear combination $y_i = \mathbf{w}_1^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}^t \mathbf{w} = 1$.
- The second principal axis, $\mathbf{w}_2$ is the linear combination $y_i = \mathbf{w}_2^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_2^t \mathbf{w}_2 = 1$ **and** such that $\mathbf{w}_2^t \mathbf{w}_1 = 0$ (orthogonal)
- The third principal axis, $\mathbf{w}_3$ is the linear combination $y_i = \mathbf{w}_3^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_3^t \mathbf{w}_3 = 1$ **and** such that $\mathbf{w}_3^t \mathbf{w}_1 = 0$ **and** $\mathbf{w}_3^t \mathbf{w}_2 = 0$
- etc
- Once we have found all the first M axes we can project an x onto the new set of axes by simply computing $\mathbf{y} = A\mathbf{x}$ where the matrix $\mathbf{A}$ is $\{\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_M^T\}$
- But how do we find the axes $\mathbf{w}$?

Principal Component Analysis

- The first principal axis, $\mathbf{w}_1$ is defined as the linear combination $y_i = \mathbf{w}_1^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}^t \mathbf{w} = 1$.
- The second principal axis, $\mathbf{w}_2$ is the linear combination $y_i = \mathbf{w}_2^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_2^t \mathbf{w}_2 = 1$ **and** such that $\mathbf{w}_2^t \mathbf{w}_1 = 0$ (orthogonal)
- The third principal axis, $\mathbf{w}_3$ is the linear combination $y_i = \mathbf{w}_3^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_3^t \mathbf{w}_3 = 1$ **and** such that $\mathbf{w}_3^t \mathbf{w}_1 = 0$ **and** $\mathbf{w}_3^t \mathbf{w}_2 = 0$
- etc
- Once we have found all the first M axes we can project an x onto the new set of axes by simply computing $\mathbf{y} = A\mathbf{x}$ where the matrix $\mathbf{A}$ is $\{\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_M^T\}$
- But how do we find the axes $\mathbf{w}$?

Principal Component Analysis

- The first principal axis, $\mathbf{w}_1$ is defined as the linear combination $y_i = \mathbf{w}_1^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}^t \mathbf{w} = 1$.
- The second principal axis, $\mathbf{w}_2$ is the linear combination $y_i = \mathbf{w}_2^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_2^t \mathbf{w}_2 = 1$ **and** such that $\mathbf{w}_2^t \mathbf{w}_1 = 0$ (orthogonal)
- The third principal axis, $\mathbf{w}_3$ is the linear combination $y_i = \mathbf{w}_3^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_3^t \mathbf{w}_3 = 1$ **and** such that $\mathbf{w}_3^t \mathbf{w}_1 = 0$ **and** $\mathbf{w}_3^t \mathbf{w}_2 = 0$
- etc
- Once we have found all the first M axes we can project an x onto the new set of axes by simply computing $\mathbf{y} = A\mathbf{x}$ where the matrix $\mathbf{A}$ is $\{\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_M^T\}$
- But how do we find the axes $\mathbf{w}$?

Principal Component Analysis

- The first principal axis, $\mathbf{w}_1$ is defined as the linear combination $y_i = \mathbf{w}_1^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}^t \mathbf{w} = 1$.
- The second principal axis, $\mathbf{w}_2$ is the linear combination $y_i = \mathbf{w}_2^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_2^t \mathbf{w}_2 = 1$ **and** such that $\mathbf{w}_2^t \mathbf{w}_1 = 0$ (orthogonal)
- The third principal axis, $\mathbf{w}_3$ is the linear combination $y_i = \mathbf{w}_3^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_3^t \mathbf{w}_3 = 1$ **and** such that $\mathbf{w}_3^t \mathbf{w}_1 = 0$ **and** $\mathbf{w}_3^t \mathbf{w}_2 = 0$
- etc
- Once we have found all the first M axes we can project an x onto the new set of axes by simply computing $\mathbf{y} = A\mathbf{x}$ where the matrix $\mathbf{A}$ is $\{\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_M^T\}$
- But how do we find the axes $\mathbf{w}$?

Principal Component Analysis

- The first principal axis, $\mathbf{w}_1$ is defined as the linear combination $y_i = \mathbf{w}_1^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}^t \mathbf{w} = 1$.
- The second principal axis, $\mathbf{w}_2$ is the linear combination $y_i = \mathbf{w}_2^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_2^t \mathbf{w}_2 = 1$ **and** such that $\mathbf{w}_2^t \mathbf{w}_1 = 0$ (orthogonal)
- The third principal axis, $\mathbf{w}_3$ is the linear combination $y_i = \mathbf{w}_3^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_3^t \mathbf{w}_3 = 1$ **and** such that $\mathbf{w}_3^t \mathbf{w}_1 = 0$ **and** $\mathbf{w}_3^t \mathbf{w}_2 = 0$
- etc
- Once we have found all the first M axes we can project an x onto the new set of axes by simply computing $\mathbf{y} = A\mathbf{x}$ where the matrix $\mathbf{A}$ is $\{\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_M^T\}$
- But how do we find the axes $\mathbf{w}$?

Principal Component Analysis

- The first principal axis, $\mathbf{w}_1$ is defined as the linear combination $y_i = \mathbf{w}_1^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}^t \mathbf{w} = 1$.
- The second principal axis, $\mathbf{w}_2$ is the linear combination $y_i = \mathbf{w}_2^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_2^t \mathbf{w}_2 = 1$ **and** such that $\mathbf{w}_2^t \mathbf{w}_1 = 0$ (orthogonal)
- The third principal axis, $\mathbf{w}_3$ is the linear combination $y_i = \mathbf{w}_3^t \mathbf{x}_i$ for which the y_i have the largest possible variance given $\mathbf{w}_3^t \mathbf{w}_3 = 1$ **and** such that $\mathbf{w}_3^t \mathbf{w}_1 = 0$ **and** $\mathbf{w}_3^t \mathbf{w}_2 = 0$
- etc
- Once we have found all the first M axes we can project an x onto the new set of axes by simply computing $\mathbf{y} = A\mathbf{x}$ where the matrix $\mathbf{A}$ is $\{\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_M^T\}$
- But how do we find the axes $\mathbf{w}$?

Principal Component Analysis

Let $S(y)$ be the variance of the values y.

By definition

$$S(y) = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \tilde{y})^2$$

Substituting $y_i = \mathbf{w}^t\mathbf{x}_i$ and performing a little algebra,

$$\begin{aligned} S(y) &= \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \tilde{y})^2 \\ &= \frac{1}{n-1}\sum_{i=1}^{n}\mathbf{w}^t(\mathbf{x}_i - \tilde{\mathbf{x}})(\mathbf{x}_i - \tilde{\mathbf{x}})^t\mathbf{w} \\ &= \mathbf{w}^t[\frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{x}_i - \tilde{\mathbf{x}})(\mathbf{x}_i - \tilde{\mathbf{x}})^t]\mathbf{w} \end{aligned}$$

Principal Component Analysis

So, we have,

$$S_y = \mathbf{w}^t \left[\frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{x}_i - \tilde{\mathbf{x}})(\mathbf{x}_i - \tilde{\mathbf{x}})^t \right] \mathbf{w}$$

but the bit inside [] is just the definition of the covariance matrix $\mathbf{S}_x$ computed from the original data points x.

So we can compute, $\mathbf{S}_x$ from our data, and then find the $\mathbf{w}$ that maximises, S_y

$$S_y = \mathbf{w}^t \mathbf{S}_x \mathbf{w}$$

The $\mathbf{w}$ is constrained to be of unit length, i.e.,

$$\mathbf{w}^t \mathbf{w} = 1$$

Principal Component Analysis

Now it turns out that the $\mathbf{w}$ that maximises $S_y = \mathbf{w}S_x\mathbf{w}$ must also satisfy,

$$S_x\mathbf{w} = \lambda\mathbf{w}$$

i.e. $\mathbf{w}$ is an eigenvector of the covariance matrix S_x.

But S_x will have more than one eigenvector. Which one do we pick?

Note, premultiplying both sides of the equation above by $\mathbf{w}^t$,

$$\mathbf{w}^t S_x\mathbf{w} = \lambda\mathbf{w}^t\mathbf{w} = \lambda$$

We want to maximise $\mathbf{w}^t S_x\mathbf{w}$, so we choose the eigenvector $\mathbf{w}$ corresponding to the eigenvalue, λ, which has the largest value.

Principal Component Analysis

For the second axis, p_1, we again want to maximise $S_y^t = \mathbf{p}_2 S_x \mathbf{p}_2$ but now subject:to the two constraints:

- $\mathbf{p}_2^t \mathbf{p}_2 = 1$ and
- $\mathbf{p}_2^t \mathbf{p}_1 = 0$ (i.e. 2nd axis is orthogonal to 1st).

With these constraints it can be shown that $\mathbf{p}_2$ is in fact the eigenvector of S_x associated with the 2nd largest eigenvalue. We continue the process, so that each new axis maximised S_y^t while being constrained to be orthogonal to all the others found so far. It turns out, the new axes are simply the eigenvectors of S_x, ordered by their respective eigenvalues.

Principal Components as 'Basis' Vectors

- The principle components, $\mathbf{w}_1$, $\mathbf{w}_2$ ... $\mathbf{w}_M$ can also be seen as a set of 'basis' vectors.
- $\mathbf{y} = \mathbf{A}\mathbf{x}$ can be written as $\mathbf{A}^T\mathbf{y} = \mathbf{x}$
  - this is because $\mathbf{A}$ is orthogonal, i.e. $\mathbf{A}\mathbf{A}^T = I$
- Remember, $\mathbf{A} = \{\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_M^T\}$ so $\mathbf{A}^T\mathbf{y} = \mathbf{x}$ is just saying,
  - $\mathbf{x} = y_1 * \mathbf{w}_1 + y_2 * \mathbf{w}_2 + ... + y_N * \mathbf{w}_N$
  - i.e., $\mathbf{x}$ is made up of a weighted sum of the principle component where the y vector is storing the weights.
- If we truncate the sum and use just the first few dimensions of $\mathbf{y}$ then,
  - $\mathbf{x} \approx y_1 * \mathbf{w}_1 + y_2 * \mathbf{w}_2 + ... + y_M * \mathbf{w}_M$ with $M < N$

Principal Components as 'Basis' Vectors

- The principle components, $\mathbf{w}_1$, $\mathbf{w}_2$ ... $\mathbf{w}_M$ can also be seen as a set of 'basis' vectors.
- $\mathbf{y} = \mathbf{A}\mathbf{x}$ can be written as $\mathbf{A}^T\mathbf{y} = \mathbf{x}$
  - this is because $\mathbf{A}$ is orthogonal, i.e. $\mathbf{A}\mathbf{A}^T = I$
- Remember, $\mathbf{A} = \{\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_M^T\}$ so $\mathbf{A}^T\mathbf{y} = \mathbf{x}$ is just saying,
  - $\mathbf{x} = y_1 * \mathbf{w}_1 + y_2 * \mathbf{w}_2 + ... + y_N * \mathbf{w}_N$
  - i.e., $\mathbf{x}$ is made up of a weighted sum of the principle component where the y vector is storing the weights.
- If we truncate the sum and use just the first few dimensions of $\mathbf{y}$ then,
  - $\mathbf{x} \approx y_1 * \mathbf{w}_1 + y_2 * \mathbf{w}_2 + ... + y_M * \mathbf{w}_M$ with $M < N$

Principal Components as 'Basis' Vectors

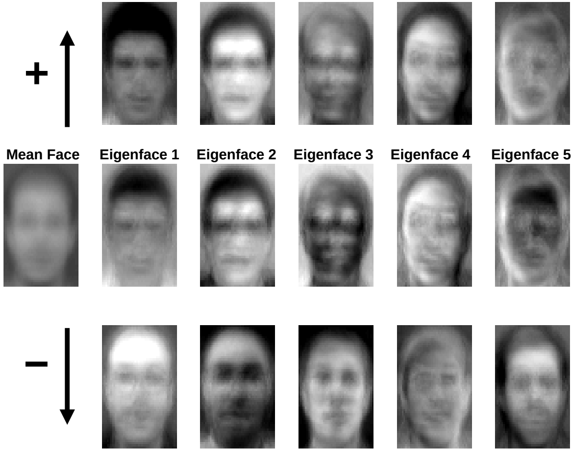
- The principle components, $\mathbf{w}_1$, $\mathbf{w}_2$ ... $\mathbf{w}_M$ can also be seen as a set of 'basis' vectors.
- $\mathbf{y} = \mathbf{A}\mathbf{x}$ can be written as $\mathbf{A}^T\mathbf{y} = \mathbf{x}$
  - this is because $\mathbf{A}$ is orthogonal, i.e. $\mathbf{A}\mathbf{A}^T = I$
- Remember, $\mathbf{A} = \{\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_M^T\}$ so $\mathbf{A}^T\mathbf{y} = \mathbf{x}$ is just saying,
  - $\mathbf{x} = y_1 * \mathbf{w}_1 + y_2 * \mathbf{w}_2 + ... + y_N * \mathbf{w}_N$
  - i.e., $\mathbf{x}$ is made up of a weighted sum of the principle component where the y vector is storing the weights.
- If we truncate the sum and use just the first few dimensions of $\mathbf{y}$ then,
  - $\mathbf{x} \approx y_1 * \mathbf{w}_1 + y_2 * \mathbf{w}_2 + ... + y_M * \mathbf{w}_M$ with $M < N$

Principal Components as ‘Basis’ Vectors

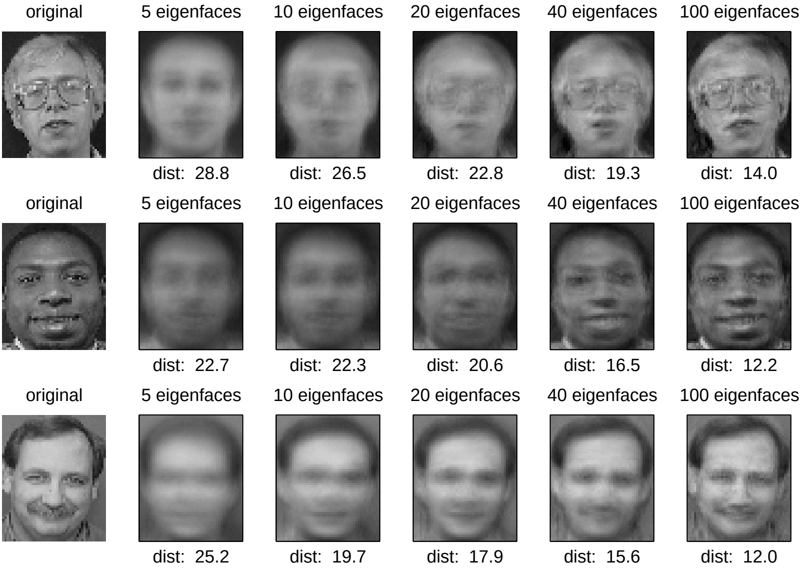
- The principle components, $\mathbf{w}_1$, $\mathbf{w}_2$ ... $\mathbf{w}_M$ can also be seen as a set of ‘basis’ vectors.
- $\mathbf{y} = \mathbf{A}\mathbf{x}$ can be written as $\mathbf{A}^T\mathbf{y} = \mathbf{x}$
  - this is because $\mathbf{A}$ is orthogonal, i.e. $\mathbf{A}\mathbf{A}^T = I$
- Remember, $\mathbf{A} = \{\mathbf{w}_1^T, \mathbf{w}_2^T, ..., \mathbf{w}_M^T\}$ so $\mathbf{A}^T\mathbf{y} = \mathbf{x}$ is just saying,
  - $\mathbf{x} = y_1 * \mathbf{w}_1 + y_2 * \mathbf{w}_2 + ... + y_N * \mathbf{w}_N$
  - i.e., $\mathbf{x}$ is made up of a weighted sum of the principle component where the y vector is storing the weights.
- If we truncate the sum and use just the first few dimensions of $\mathbf{y}$ then,
  - $\mathbf{x} \approx y_1 * \mathbf{w}_1 + y_2 * \mathbf{w}_2 + ... + y_M * \mathbf{w}_M$ with $M < N$

Faces – variability represented by first 5 principal components

After applying PCA to image data we can reshape the principle component vectors into matrices and display them as images.

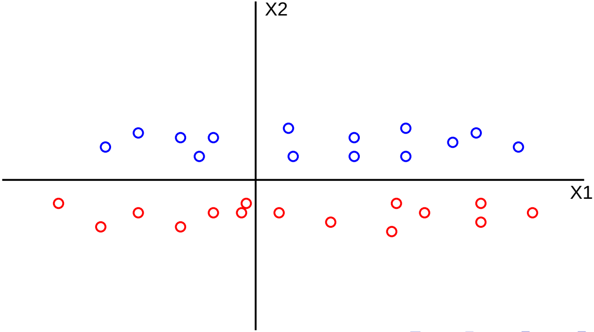


Faces – using increasing number of PCA dimensions



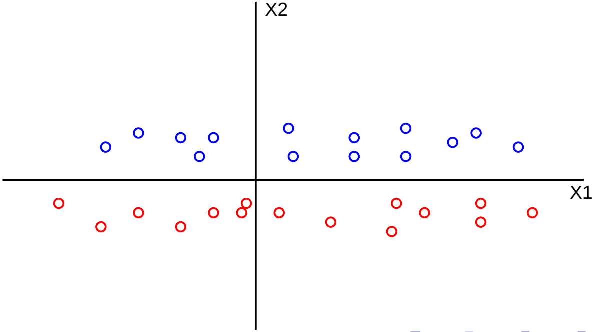
Linear Discriminant Analysis

- PCA ensures that the data
  - has independent features – easy to model statistically
  - is well spread out – so classes are less likely to overlap
- But ‘spreadoutness’ has been maximised by looking at the data as a whole, i.e.,
  - the algorithm doesn’t make use of the class labels,
  - and spreading out the data doesn’t necessarily separate the classes,
  - e.g., in example below, X1 spreads the data but X2 separates the classes,



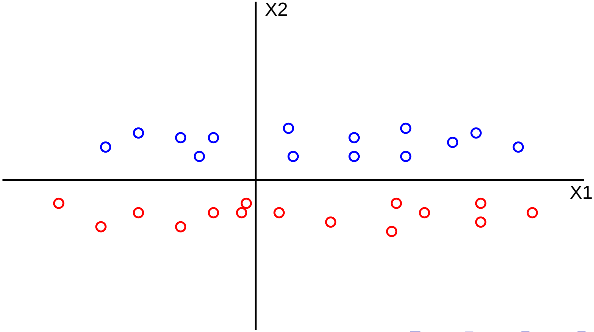
Linear Discriminant Analysis

- PCA ensures that the data
  - has independent features – easy to model statistically
  - is well spread out – so classes are less likely to overlap
- But ‘spreadoutness’ has been maximised by looking at the data as a whole, i.e.,
  - the algorithm doesn’t make use of the class labels,
  - and spreading out the data doesn’t necessarily separate the classes,
  - e.g., in example below, X1 spreads the data but X2 separates the classes,
  -



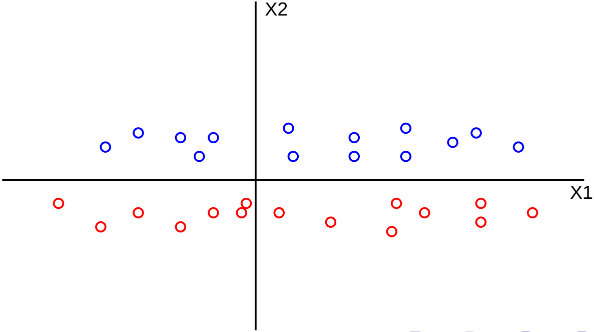
Linear Discriminant Analysis

- PCA ensures that the data
  - has independent features – easy to model statistically
  - is well spread out – so classes are less likely to overlap
- But ‘spreadoutness’ has been maximised by looking at the data as a whole, i.e.,
  - the algorithm doesn’t make use of the class labels,
  - and spreading out the data doesn’t necessarily separate the classes,
  - e.g., in example below, X1 spreads the data but X2 separates the classes,



Linear Discriminant Analysis

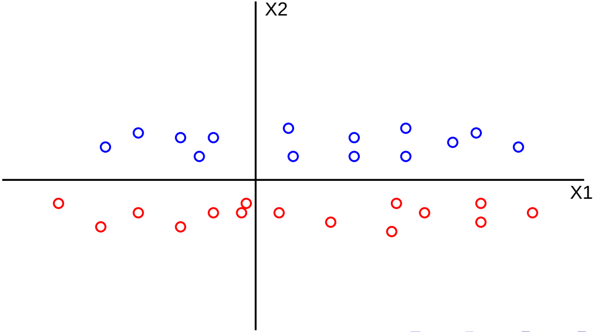
- PCA ensures that the data
  - has independent features – easy to model statistically
  - is well spread out – so classes are less likely to overlap
- But 'spreadoutness' has been maximised by looking at the data as a whole, i.e.,
  - the algorithm doesn't make use of the class labels,
  - and spreading out the data doesn't necessarily separate the classes,
  - e.g., in example below, X1 spreads the data but X2 separates the classes,



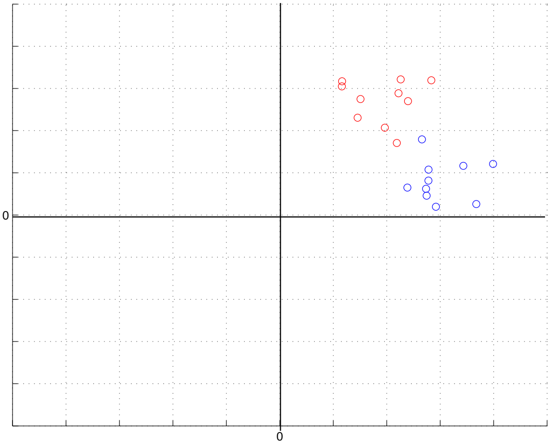
-

Linear Discriminant Analysis

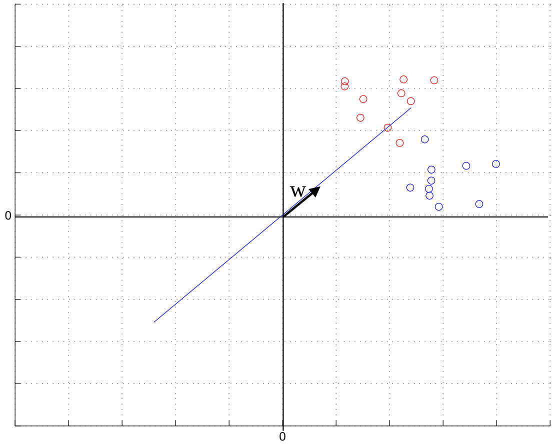
- PCA ensures that the data
  - has independent features – easy to model statistically
  - is well spread out – so classes are less likely to overlap
- But ‘spreadoutness’ has been maximised by looking at the data as a whole, i.e.,
  - the algorithm doesn’t make use of the class labels,
  - and spreading out the data doesn’t necessarily separate the classes,
  - e.g., in example below, X1 spreads the data but X2 separates the classes,



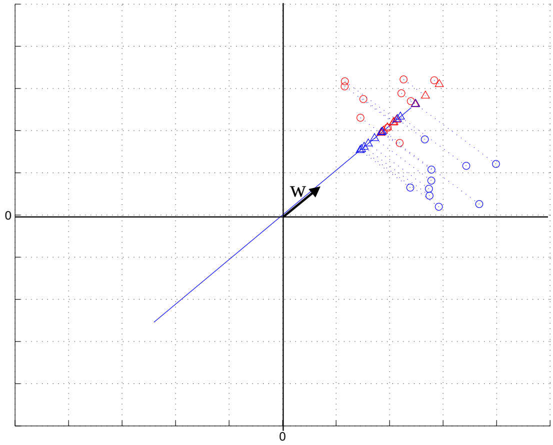
Linear Discriminant Analysis



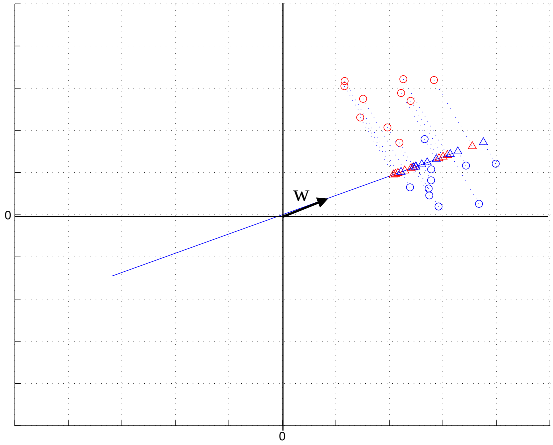
Linear Discriminant Analysis



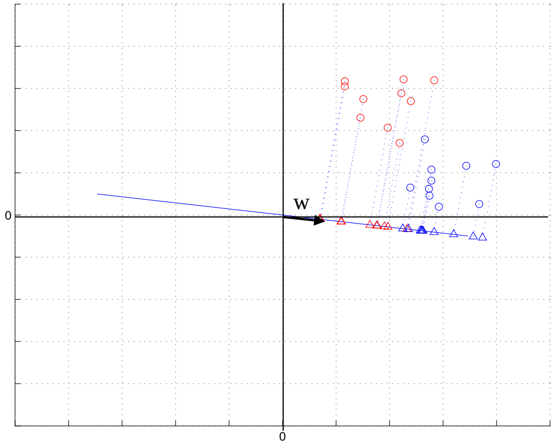
Linear Discriminant Analysis



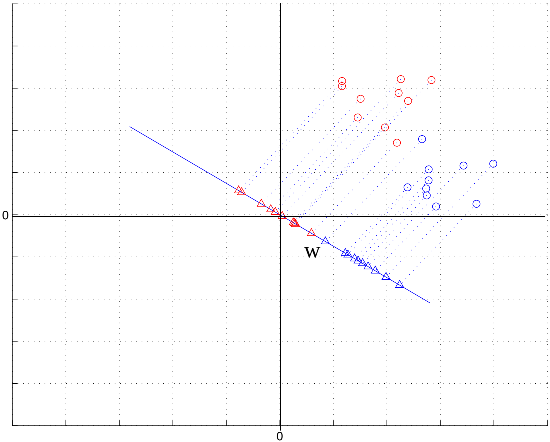
Linear Discriminant Analysis



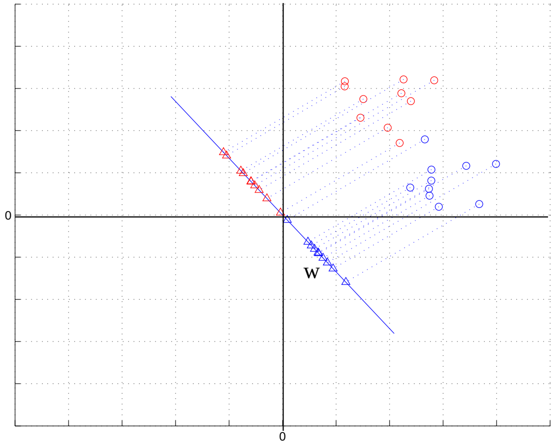
Linear Discriminant Analysis



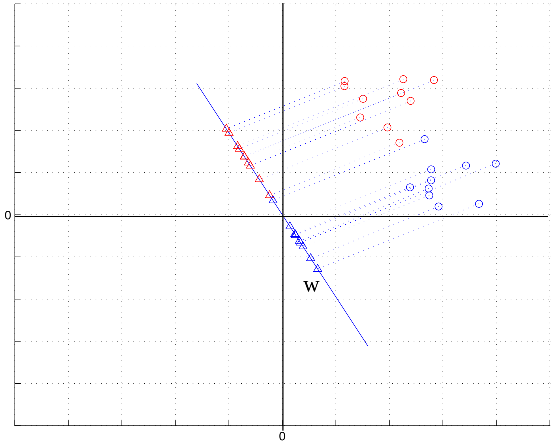
Linear Discriminant Analysis



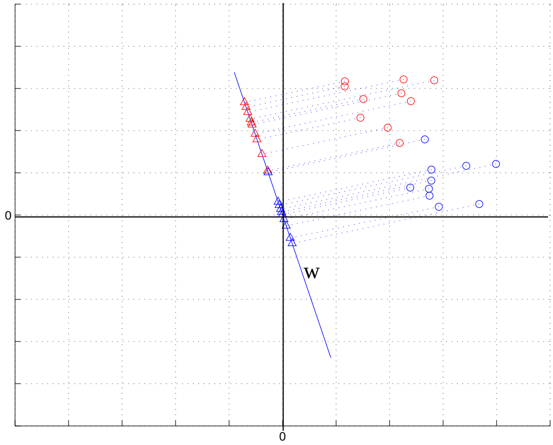
Linear Discriminant Analysis



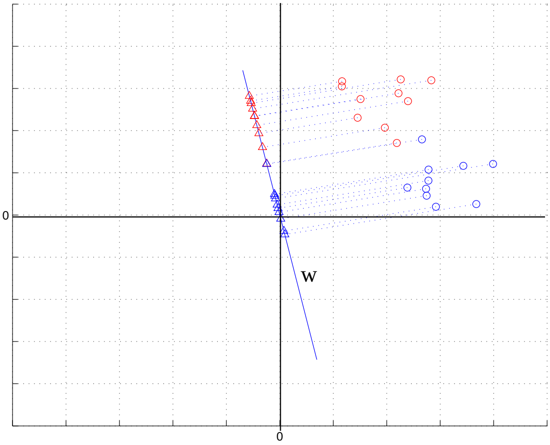
Linear Discriminant Analysis



Linear Discriminant Analysis



Linear Discriminant Analysis



Summary

- Dimensionality reduction is a generalisation of the feature selection idea
- A popular approach is to perform a linear transform of the data.
- DCT - a fixed transform that works well for sequence data.
- PCA - a data-driven transform that aims to reduce dimensionality while retaining the spread of the data.
- LDA - a data and label driven transform that reduces dimensionality while retaining the separability of the classes.
- PCA and LDA also result in decorrelated features that mean simple statistical models can be used for classification.

Summary

- Dimensionality reduction is a generalisation of the feature selection idea
- A popular approach is to perform a linear transform of the data.
- DCT - a fixed transform that works well for sequence data.
- PCA - a data-driven transform that aims to reduce dimensionality while retaining the spread of the data.
- LDA - a data and label driven transform that reduces dimensionality while retaining the separability of the classes.
- PCA and LDA also result in decorrelated features that mean simple statistical models can be used for classification.

Summary

- Dimensionality reduction is a generalisation of the feature selection idea
- A popular approach is to perform a linear transform of the data.
- DCT - a fixed transform that works well for sequence data.
- PCA - a data-driven transform that aims to reduce dimensionality while retaining the spread of the data.
- LDA - a data and label driven transform that reduces dimensionality while retaining the separability of the classes.
- PCA and LDA also result in decorrelated features that mean simple statistical models can be used for classification.

Summary

- Dimensionality reduction is a generalisation of the feature selection idea
- A popular approach is to perform a linear transform of the data.
- DCT - a fixed transform that works well for sequence data.
- PCA - a data-driven transform that aims to reduce dimensionality while retaining the spread of the data.
- LDA - a data and label driven transform that reduces dimensionality while retaining the separability of the classes.
- PCA and LDA also result in decorrelated features that mean simple statistical models can be used for classification.

Summary

- Dimensionality reduction is a generalisation of the feature selection idea
- A popular approach is to perform a linear transform of the data.
- DCT - a fixed transform that works well for sequence data.
- PCA - a data-driven transform that aims to reduce dimensionality while retaining the spread of the data.
- LDA - a data and label driven transform that reduces dimensionality while retaining the separability of the classes.
- PCA and LDA also result in decorrelated features that mean simple statistical models can be used for classification.

Summary

- Dimensionality reduction is a generalisation of the feature selection idea
- A popular approach is to perform a linear transform of the data.
- DCT - a fixed transform that works well for sequence data.
- PCA - a data-driven transform that aims to reduce dimensionality while retaining the spread of the data.
- LDA - a data and label driven transform that reduces dimensionality while retaining the separability of the classes.
- PCA and LDA also result in decorrelated features that mean simple statistical models can be used for classification.