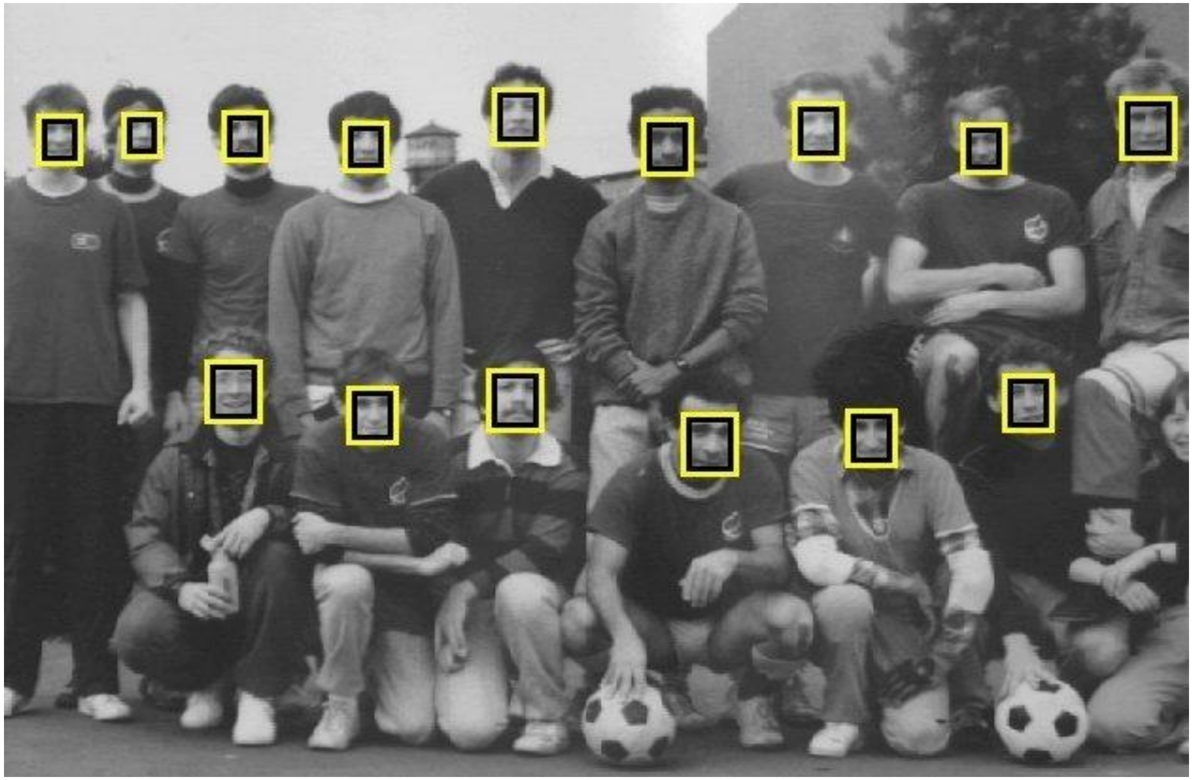
Machine Learning

Lecture 8

Data Processing and Representation
Principal Component Analysis (PCA)

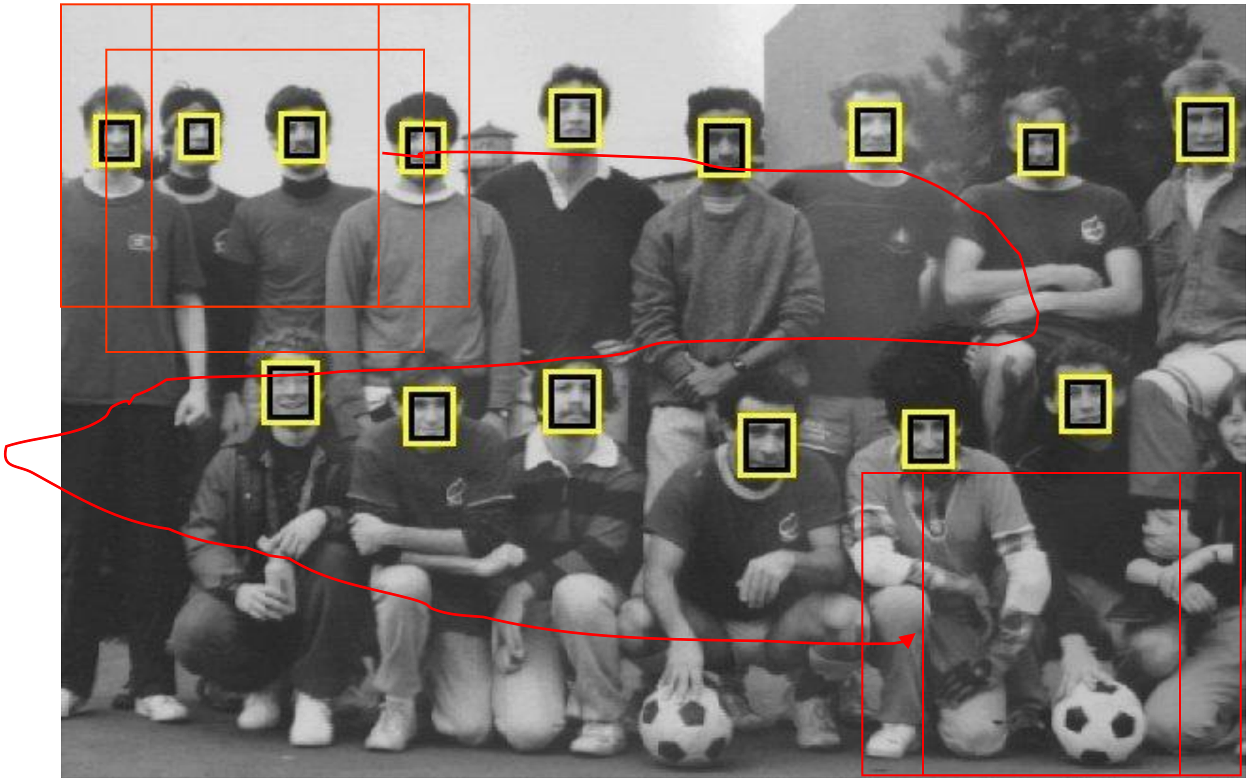
Problems

- Object Detection



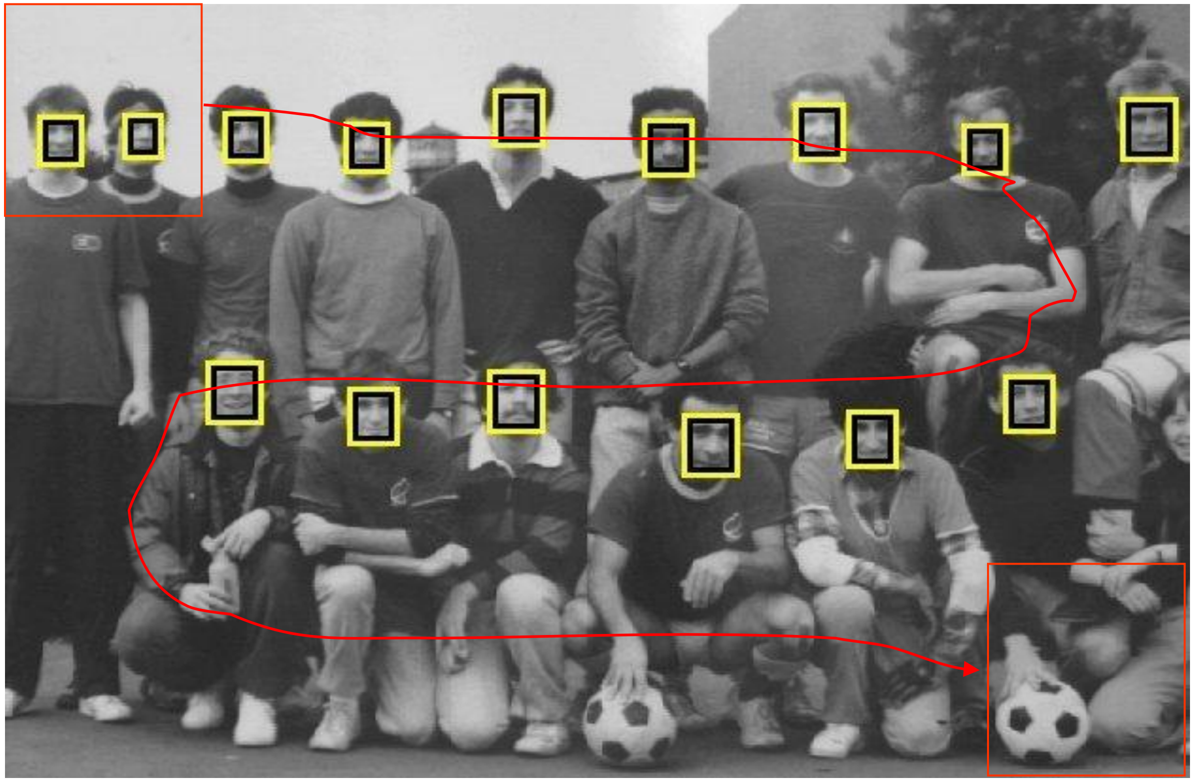
Problems

- **Object Detection:** Many detection windows



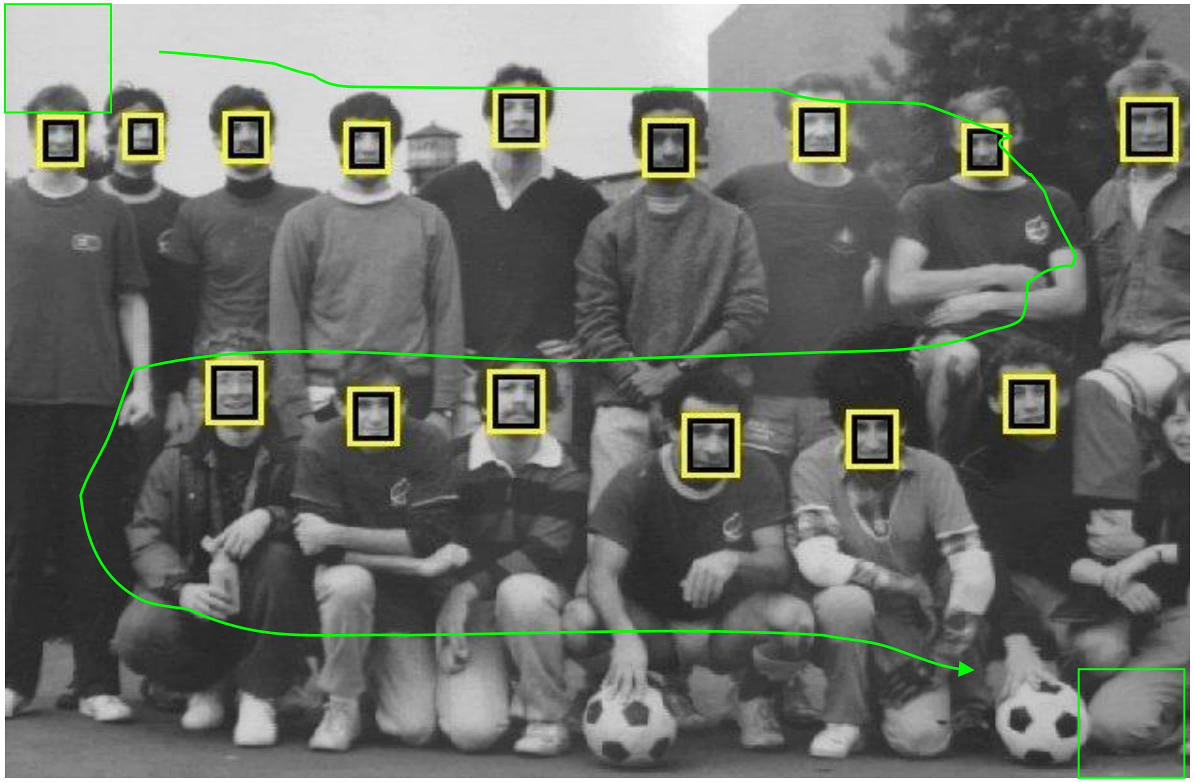
Problems

- **Object Detection:** Many detection windows



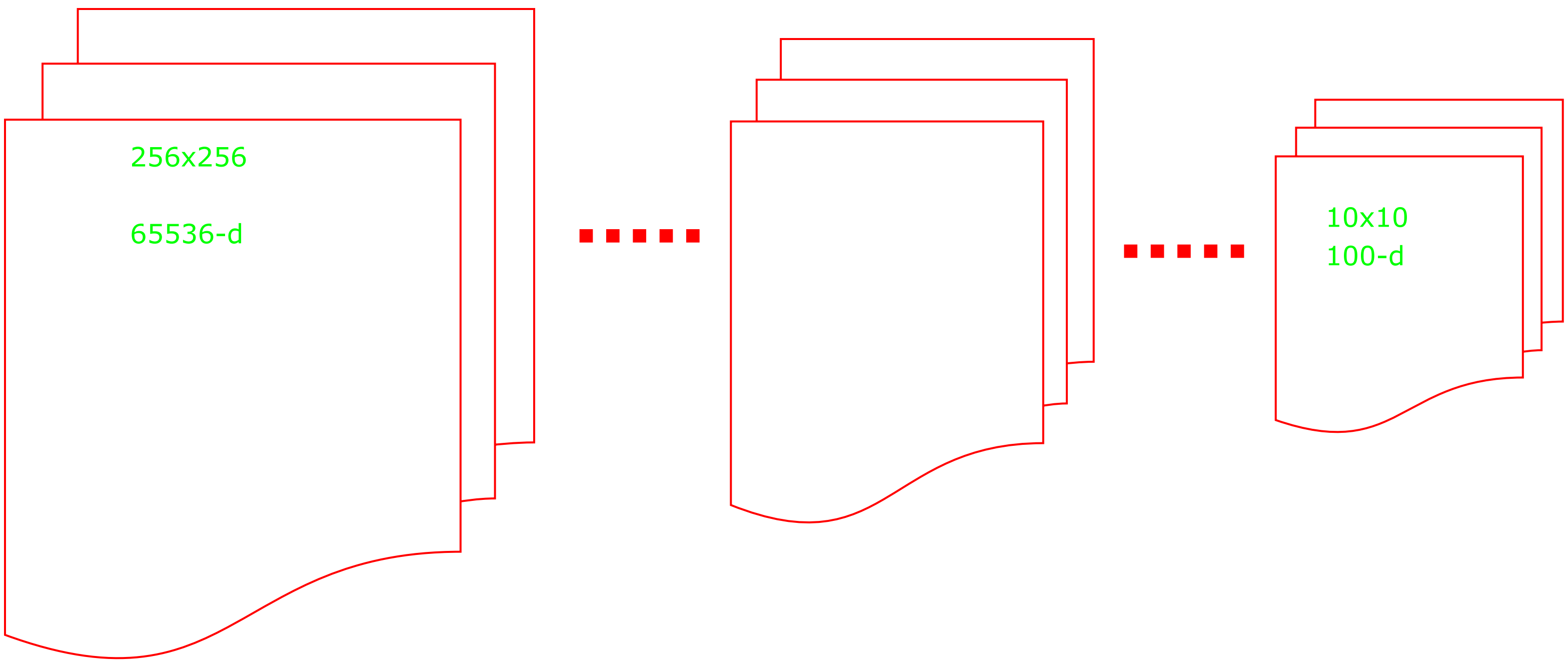
Problems

- Object Detection: Many detection windows



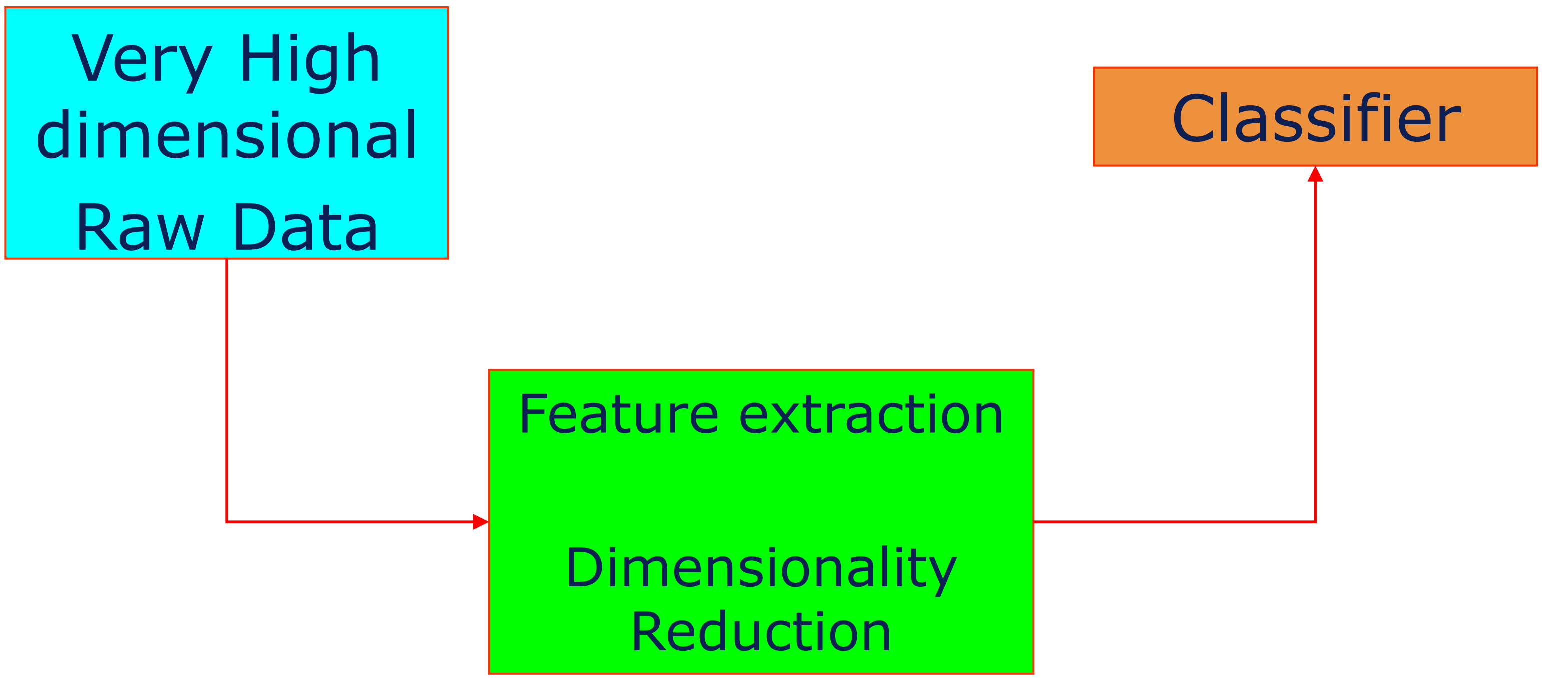
Problems

- Object Detection: Each window is very high dimension data



Processing Methods

- General framework



Feature extraction/Dimensionality reduction

- It is impossible to processing raw image data (pixels) directly
  - Too many of them (or data dimensionality too high)
  - Curse of dimensionality problem
- Process the raw pixel to produce a smaller set of numbers which will capture most information contained in the original data – this is often called a feature vector

Feature extraction/Dimensionality reduction

- Basic Principle
  - From a raw data (vector) X of N-dimension to a new vector Y of n-dimensional (n < < N) via a transformation matrix A such that Y will capture most information in X

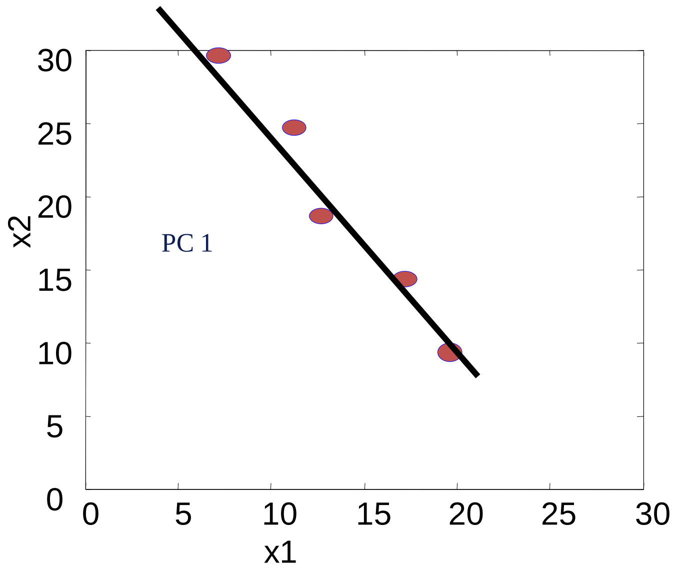
$$Y = AX = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & & a_{1N} \\ & \cdots & & \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & \cdots & & a_{nN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_N \end{bmatrix}$$

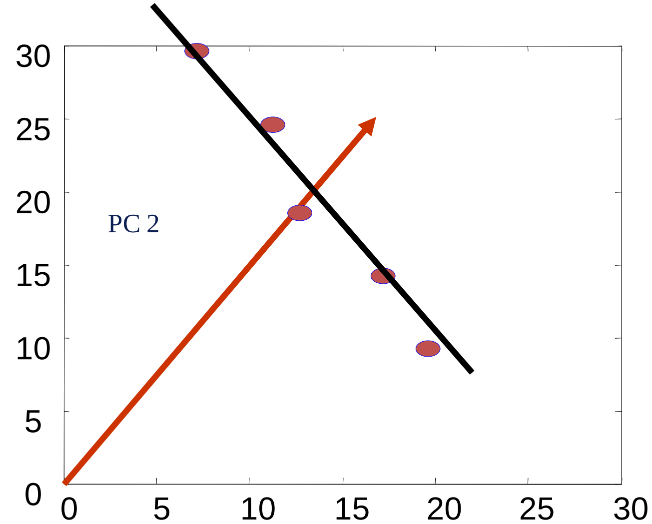
PCA

- Principal Component Analysis (PCA) is one of the most often used dimensionality reduction technique.

Principal Components

- All principal components (PCs) start at the origin of the ordinate axes.
- First PC is direction of maximum variance from origin
- Subsequent PCs are orthogonal to 1st PC and describe maximum residual variance





PCA Goal

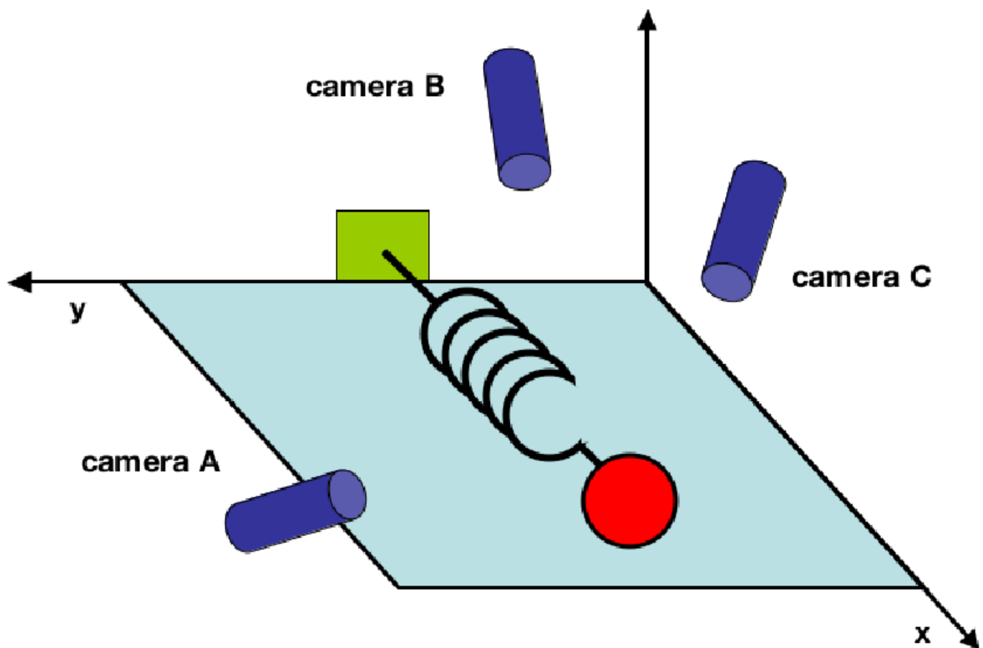
We wish to explain/summarize the underlying variance-covariance structure of a large set of variables through a few linear combinations of these variables.

Applications

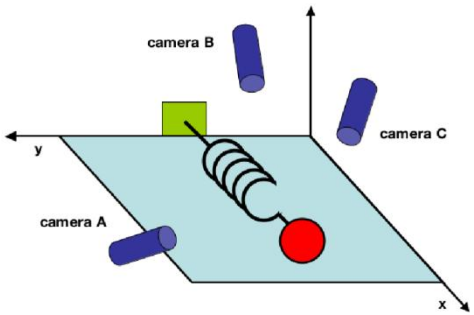
– Data Visualization
– Data Reduction
– Data Classification
– Trend Analysis
– Factor Analysis
– Noise Reduction

An example

- A toy example: The movement of an ideal spring, the underlying dynamics can be expressed as a function of a single variable x.



An example



- But, pretend that we are ignorant of that and
- Using 3 cameras, each records 2d projection of the ball's position. We record the data for 2 minutes at 200Hz
- We have 12,000, 6-d data
- How can we work out the dynamic is only along the x-axis
- Thus determining that only the dynamics along x are important and the rest are redundant.

An example

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{16} \\ & & \\ & & \\ & \cdots & \\ & & \\ a_{61} & \cdots & a_{66} \end{bmatrix} \begin{bmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \end{bmatrix}$$

An example

1st Eigenvector of the Covariance matrix

2nd Eigenvector of the Covariance matrix

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{16} \\ & & \\ & \cdots & \\ & & \\ & & \\ a_{61} & \cdots & a_{66} \end{bmatrix} \begin{bmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \end{bmatrix}$$

6th Eigenvector of the Covariance matrix

An example

1st Eigenvector of the Covariance matrix

2nd Eigenvector of the Covariance matrix

1st Principal Component

2nd Principal Component

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{16} \\ & & \\ & \cdots & \\ & & \\ & & \\ a_{61} & \cdots & a_{66} \end{bmatrix} \begin{bmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \end{bmatrix}$$

6th Eigenvector of the Covariance matrix

PCA

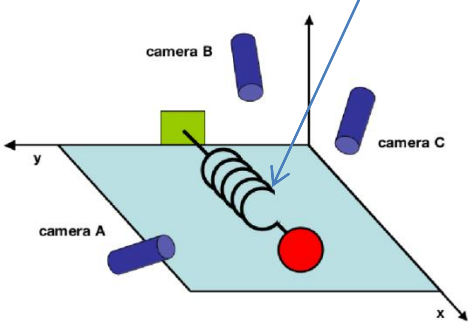
1st Eigenvector of the Covariance matrix

2nd Eigenvector of the Covariance matrix

6th Eigenvector of the Covariance matrix

Dynamic of the spring

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{16} \\ & & \\ & \cdots & \\ & & \\ a_{61} & \cdots & a_{66} \end{bmatrix} \begin{bmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \end{bmatrix}$$



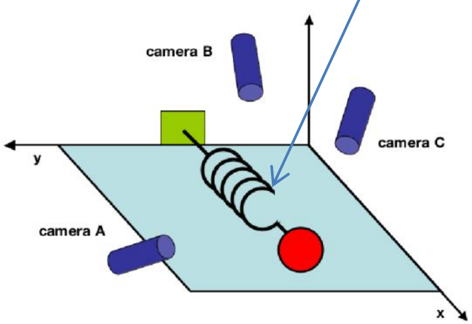
PCA

1st Eigenvector of the Covariance matrix

2nd Eigenvector of the Covariance matrix

Dynamic of the spring

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{16} \\ & & \\ & \cdots & \\ & & \\ a_{61} & \cdots & a_{66} \end{bmatrix} \begin{bmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \end{bmatrix}$$



They contain no useful information and can be discarded!

6th Eigenvector of the Covariance matrix

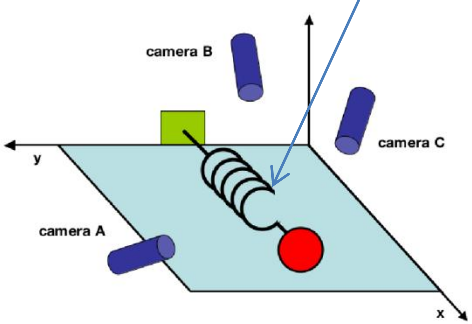
PCA

We only need ONE number

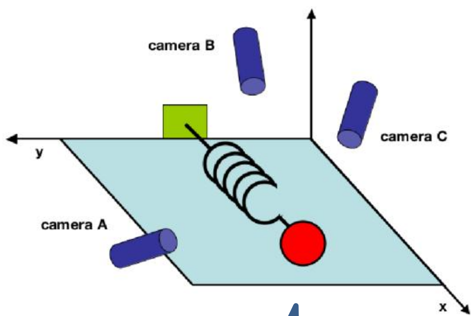
Dynamic of the spring

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{16} \\ & & \\ & \cdots & \\ & & \\ & & \\ a_{61} & \cdots & a_{66} \end{bmatrix} \begin{bmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \end{bmatrix}$$

Instead of
SIX
Numbers!



PCA



$$\begin{bmatrix} x_A \\ y_A \\ x_B \\ y_B \\ x_C \\ y_C \end{bmatrix} = \begin{bmatrix} a_{11} \\ \\ \\ \\ \\ a_{16} \end{bmatrix} \begin{bmatrix} v_1 \end{bmatrix}$$

Linear combination (scaling) of ONE variable

Capture the data patterns of SIX Numbers!

Noise

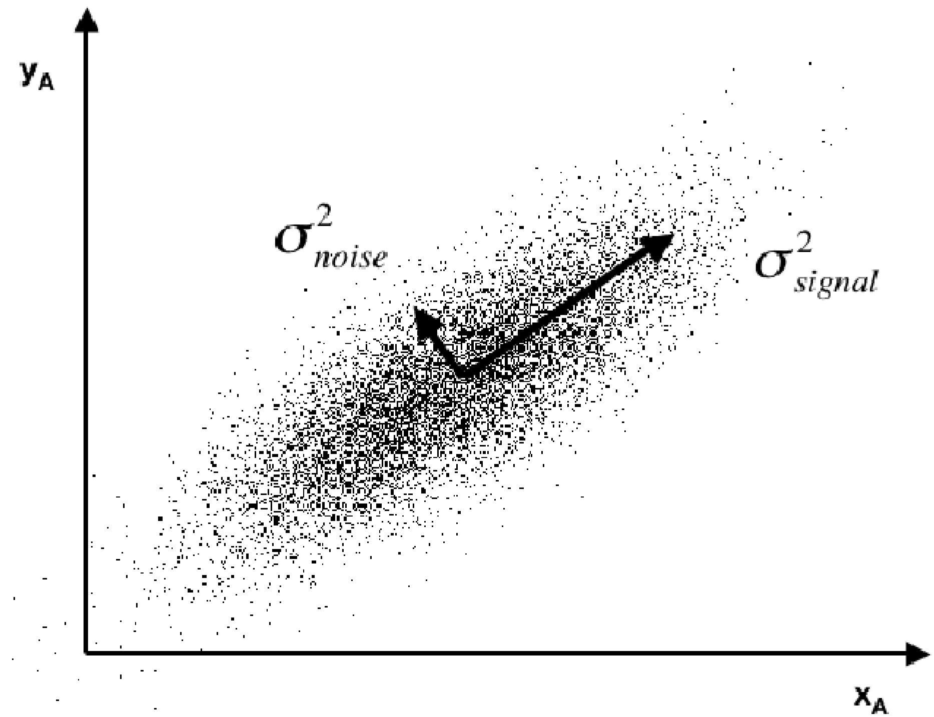


Figure 2: A simulated plot of (x_A, y_A) for camera A. The signal and noise variances σ^2_{signal} and σ^2_{noise} are graphically represented.

Redundancy

r1 and r2 entirely uncorrelated,

No redundancy in the two recordings

r1 and r2 strongly correlated,

high redundancy in the two recordings

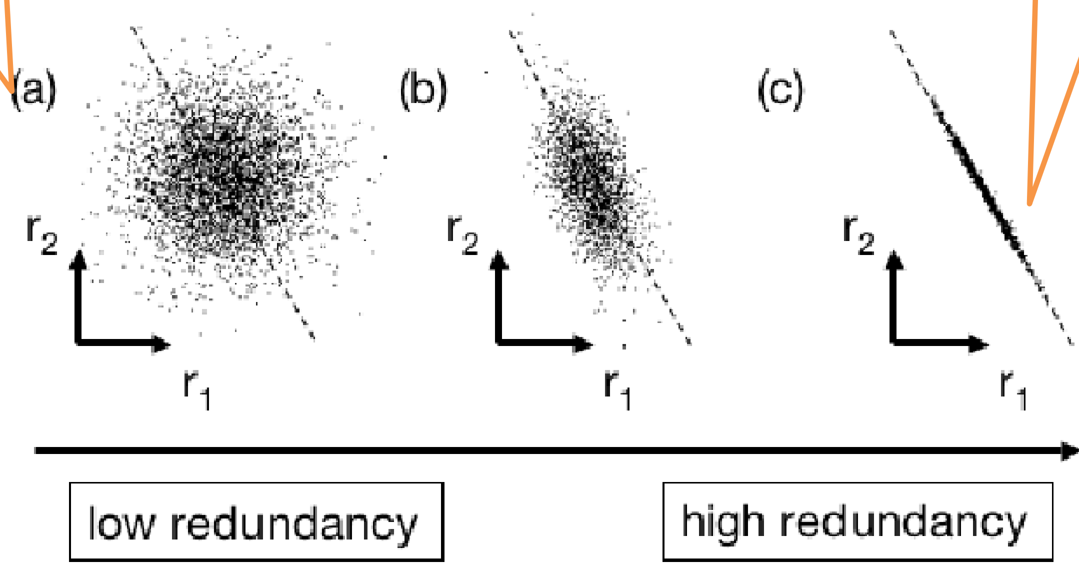


Figure 3: A spectrum of possible redundancies in data from the two separate recordings r_1 and r_2 (e.g. x_A, y_B). The best-fit line $r_2 = kr_1$ is indicated by the dashed line.

Covariance matrix

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ & & & \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}$$

One of the measurements of ALL samples (n samples)

One sample (m-d)

Covariance matrix

$$S_X = \frac{1}{n-1} XX^T$$

is the covariance matrix of the data

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ & & & \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}$$

One of the measurements of ALL samples

One sample

Covariance matrix

$$S_X = \frac{1}{n-1} X X^T$$

- S_x is an m x m square matrix, m is the dimensionality of the measures (feature vectors)
- The diagonal terms of S_x are the variance of particular measurement type
- The off-diagonal terms of S_x are the covariance between measurement types

Covariance matrix

$$S_X = \frac{1}{n-1} X X^T$$

- S_x is special.
- It describes all relationships between pairs of measurements in our data set.
- A larger covariance indicates large correlation (more redundancy), zero covariance indicates entirely uncorrelated data.

Covariance matrix

- Diagonalise the covariance matrix

- If our goal is to reduce redundancy, then we want each variable co-vary a little as possible

- Precisely, we want the covariance between separate measurements to be zero

Feature extraction/Dimensionality reduction

- Remove redundancy

$$Y = AX = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & & a_{1m} \\ & \cdots & & \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & \cdots & & a_{mm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

- Optimal covariance matrix S_Y - off-diagonal terms set zero
- **Therefore removing redundancy, diagonalises S_Y**

E… …nality reduction

How to find the transformation matrix

- Remove redu…

$$Y = AX = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & & a_{1m} \\ & \cdots & & \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & \cdots & & a_{mm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

- Optimal covariance matrix S_Y - off-diagonal terms set zero
- **Therefore removing redundancy, diagonalises S_Y**

Solving PCA: Diagonalising the Covariance Matrix

- There are many ways to diagonalizing S_Y, PCA choose the simplest method.

- PCA assumes all basis vectors are orthonormal. *P* is an orthonormal matrix

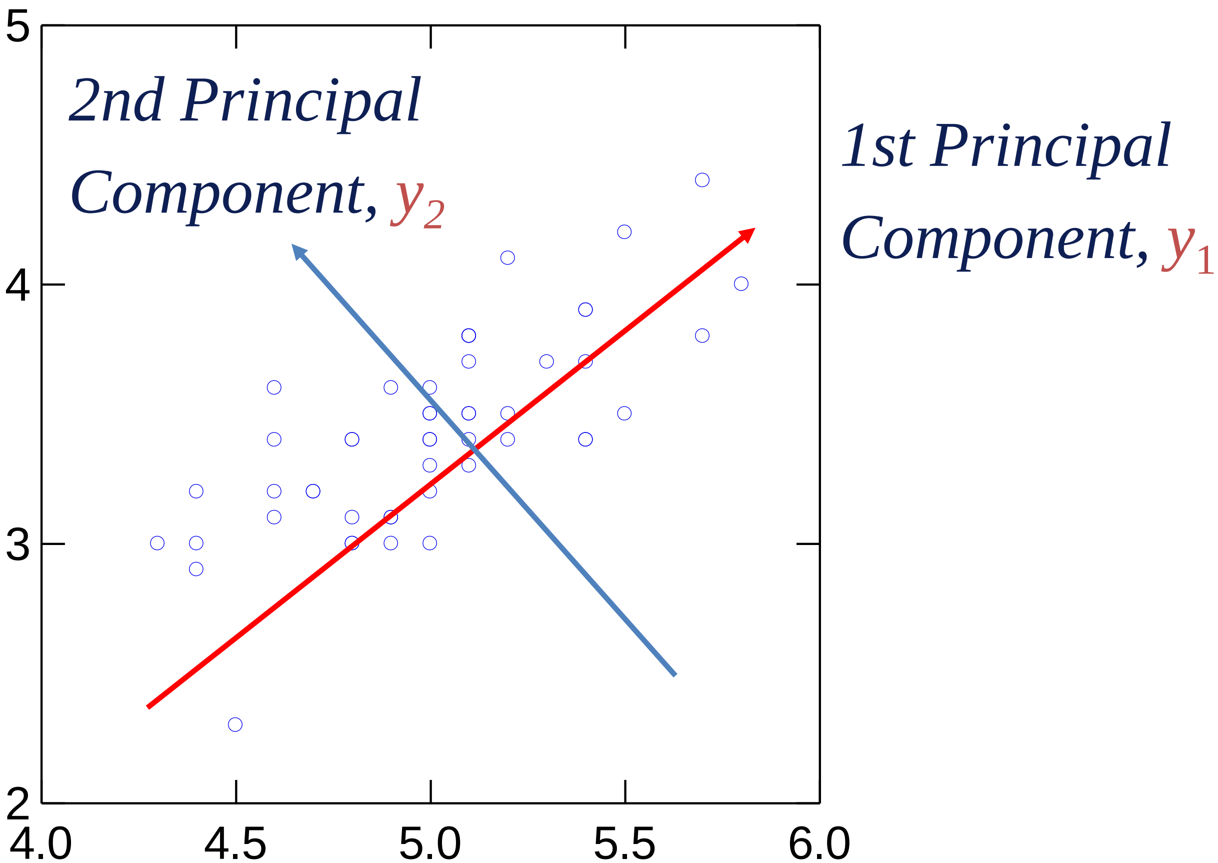
$$p_i = \begin{bmatrix} p_{i1} & p_{i2} & \cdots & p_{im} \end{bmatrix}$$

$$p_i p_j = \delta_{ij} \qquad \delta_{ij} = \begin{cases} 1 & if \quad i = j \\ 0 & if \quad i \neq j \end{cases}$$

- PCA assumes the directions with the largest variances are the most important or most ***principal***.

Solving PCA: Diagonalising the Covariance Matrix

- PCA works as follows
  - PCA first selects a normalised direction in m-dimensional space along which the variance of X is maximised – it saves the direction as p_1
  - It then finds another direction, along which variance is maximised subject to the orthonormal condition – it restricts its search to all directions perpendicular to all previous selected directions.
  - The process could continue until m directions are found. The resulting ORDERED set of p's are the ***principal components***
  - The variances associated with each direction p_i quantify how principal (important) each direction is – thus rank-ordering each basis according to the corresponding variance



$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} p_{11} & \cdots & & p_{1m} \\ & \cdots & & \\ \cdots & \cdots & \cdots & \cdots \\ p_{m1} & \cdots & & p_{mm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

Solving PCA Eigenvectors of Covariance

$$Y = PX \qquad S_Y = \frac{1}{n-1} YY^T$$

- Find some orthonormal matrix P such that S_Y is diagonalized.
- The row of P are the principal components of X

Solving PCA Eigenvectors of Covariance

$$S_Y = \frac{1}{n-1}YY^T = \frac{1}{n-1}(PX)(PX)^T$$

$$S_Y = \frac{1}{n-1}PXX^TP^T$$

$$S_Y = \frac{1}{n-1}P\left(XX^T\right)P^T$$

$$S_Y = \frac{1}{n-1}PAP^T$$

$$where \quad A = XX^T$$

- A is a symmetric matrix, which can be diagonalised by an orthonormal matrix of its eigenvectors.

Solving PCA Eigenvectors of Covariance

$$A = EDE^T$$

- D is a diagonal matrix, E is a matrix of eigenvectors of A arranged as columns
- The matrix A has r < = m orthonormal eigenvectors, where r is the rank of A.
- r is less than m when A is degenerate or all data occupy a subspace of dimension r < m

Solving PCA Eigenvectors of Covariance

$$A = EDE^T \qquad P \equiv E^T \qquad A = P^T DP$$

- Select the matrix P to be a matrix where each row p_i is an eigenvector of XX^T.

$$S_Y = \frac{1}{n-1} PAP^T$$

$$S_Y = \frac{1}{n-1} P\left(P^T DP\right)P^T$$

$$S_Y = \frac{1}{n-1} PP^T DPP^T = \frac{1}{n-1}\left(PP^T\right)D\left(PP^T\right)$$

$$S_Y = \frac{1}{n-1} D$$

Solving PCA Eigenvectors of Covariance

$$S_Y = \frac{1}{n-1} D$$

- The principal component of X are the eigenvectors of XX^T; or the rows of P

- The ith diagonal value of S_Y is the variance of X along p_i

PCA Procedures

- Get data (example)
- Step 1
  - Subtract the mean (example)
- Step 2
  - Calculate the covariance matrix
- Step 3
  - Calculate the eigenvectors and eigenvalues of the covariance matrix

A 2D Numerical Example

PCA Example – Data

- Original data

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

STEP 1

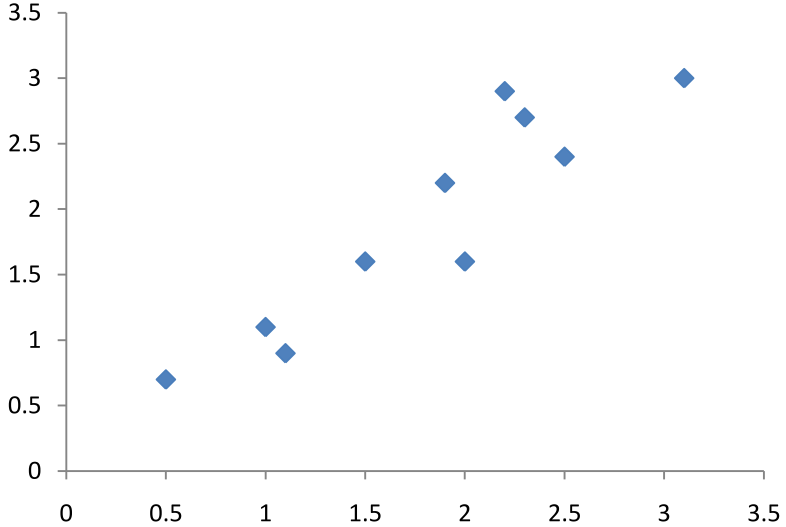
- Subtract the mean
- from each of the data dimensions. All the x values have average (x) subtracted and y values have average (y) subtracted from them. This produces a data set whose mean is zero.
- Subtracting the mean makes variance and covariance calculation easier by simplifying their equations. The variance and co-variance values are not affected by the mean value.

STEP 1

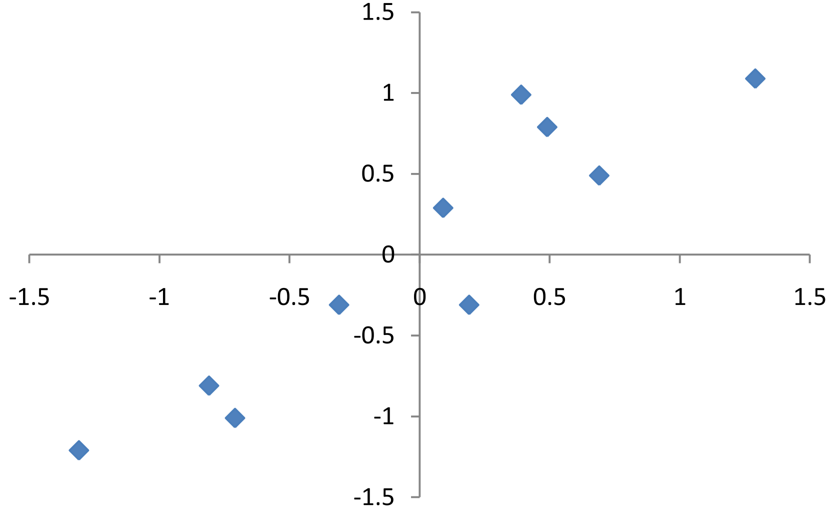
- Zero-mean data

0.69	0.49
-1.31	-1.21
0.39	0.99
0.09	0.29
1.29	1.09
0.49	0.79
0.19	-0.31
-0.81	-0.81
-0.31	-0.31
-0.71	-1.01

STEP 1



Original



Zero-mean

STEP 2

- Calculate the covariance matrix

$$\text{cov} = \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix}$$

- since the non-diagonal elements in this covariance matrix are positive, we should expect that both the x and y variable increase together.

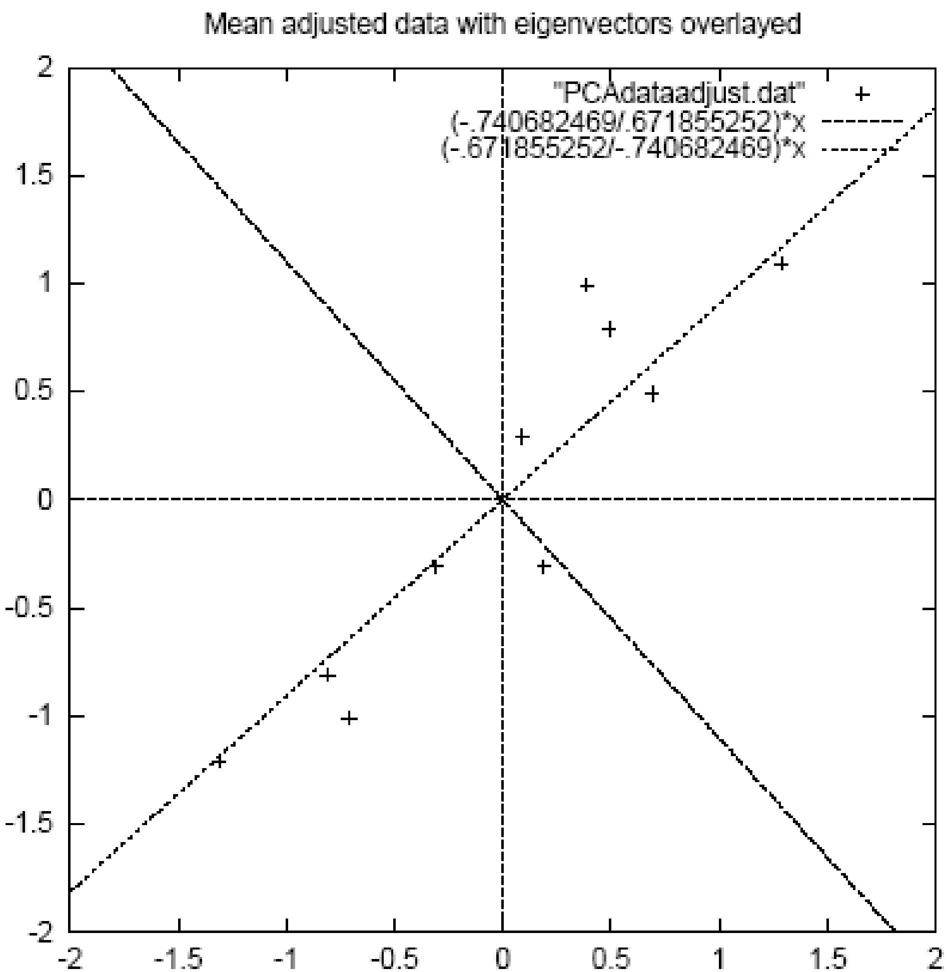
STEP 3

- Calculate the eigenvectors and eigenvalues of the covariance matrix

$$\text{eigenvalues} = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$\text{eigenvectors} = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

STEP 3



- eigenvectors are plotted as diagonal dotted lines on the plot.
- Note they are perpendicular to each other.
- Note one of the eigenvectors goes through the middle of the points, like drawing a line of best fit.
- The second eigenvector gives us the other, less important, pattern in the data, that all the points follow the main line, but are off to the side of the main line by some amount.

Figure 3.2: A plot of the normalised data (mean subtracted) with the eigenvectors of the covariance matrix overlayed on top.

Feature Extraction

- Reduce dimensionality and form *feature vector*
  - the eigenvector with the *highest* eigenvalue is the *principal component* of the data set.
  - In our example, the eigenvector with the larges eigenvalue was the one that pointed down the middle of the data.
  - Once eigenvectors are found from the covariance matrix, the next step is to order them by eigenvalue, highest to lowest. This gives you the components in order of significance.

Feature Extraction

- Eigen Feature Vector

  $$\text{FeatureVector} = (eig_1\ eig_2\ eig_3 \ldots\ eig_n)$$

  We can either form a feature vector with both of the eigenvectors:

  $$\begin{pmatrix} -.677873399 & -.735178656 \\ -.735178656 & .677873399 \end{pmatrix}$$

  or, we can choose to leave out the smaller, less significant component and only have a single column:

  $$\begin{pmatrix} -.677873399 \\ -.735178656 \end{pmatrix}$$

Eigen-analysis/ Karhunen Loeve Transform

Eigen Matrix

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} p_{11} & \cdots & & p_{1m} \\ & \cdots & & \\ \ldots & \cdots & \cdots & \ldots \\ p_{m1} & \cdots & & p_{mm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

Eigen-analysis/ Karhunen Loeve Transform

Back to our example: Transform data to eigen-space (x’ , y’)

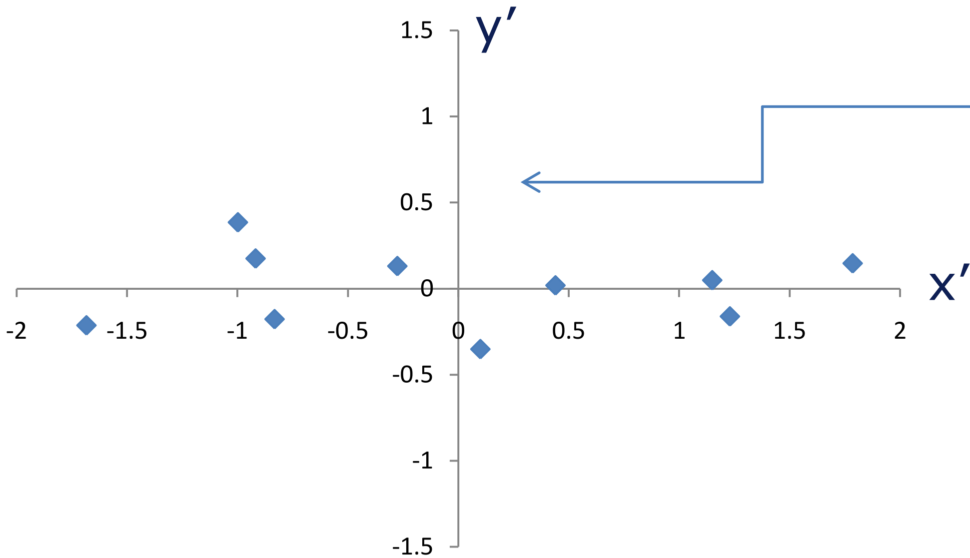
x’ = -0.68x - 0.74y   y’ = -0.74x + 0.68y

-.827970186	-.175115307
1.77758033	.142857227
-.992197494	.384374989
-.274210416	.130417207
-1.67580142	-.209498461
-.912949103	.175282444
.0991094375	-.349824698
1.14457216	.0464172582
.438046137	.0177646297
1.22382056	-.162675287

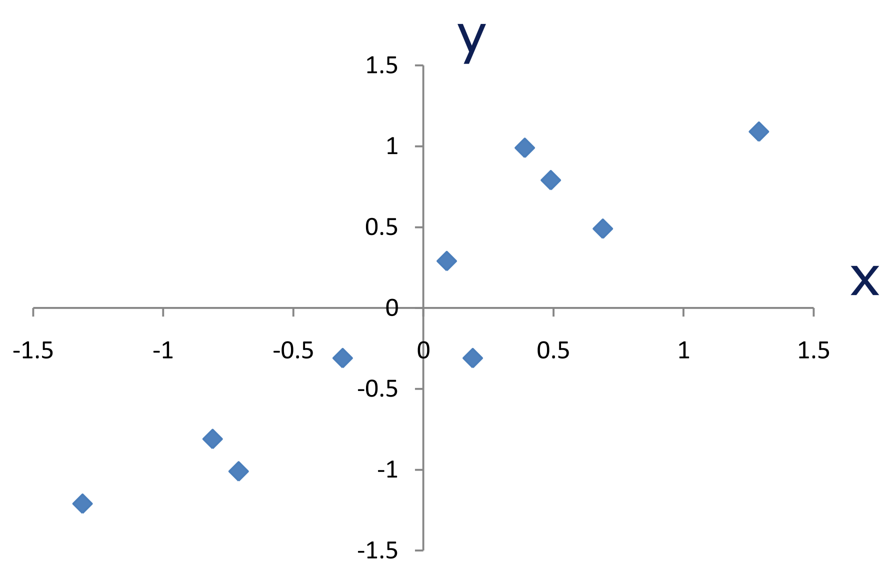
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -0.68 & -0.74 \\ -0.74 & 0.68 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

x	y
0.69	0.49
-1.31	-1.21
0.39	0.99
0.09	0.29
1.29	1.09
0.49	0.79
0.19	-0.31
-0.81	-0.81
-0.31	-0.31
-0.71	-1.01

Eigen-analysis/ Karhunen Loeve Transform



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -0.68 & -0.74 \\ -0.74 & 0.68 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



Reconstruction of original Data/Inverse Transformation

- Forward Transform

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -0.68 & -0.74 \\ -0.74 & 0.68 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- Inverse Transform

$$\begin{bmatrix} x_{construction} \\ y_{construction} \end{bmatrix} = \begin{bmatrix} -0.68 & -0.74 \\ -0.74 & 0.68 \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix}$$

Reconstruction of original Data/Inverse Transformation

- If we reduced the dimensionality, obviously, when reconstructing the data we would lose those dimensions we chose to discard.

- Thrown away the less important one, throw away y' and only keep x'

$$\begin{bmatrix} x_{reconstruction} \\ y_{reconstruction} \end{bmatrix} = \begin{bmatrix} -0.68 \\ -0.74 \end{bmatrix} \begin{bmatrix} x' \end{bmatrix}$$

Reconstruction of original Data/Inverse Transformation

x'

-.827970186

1.77758033

-.992197494

-.274210416

-1.67580142

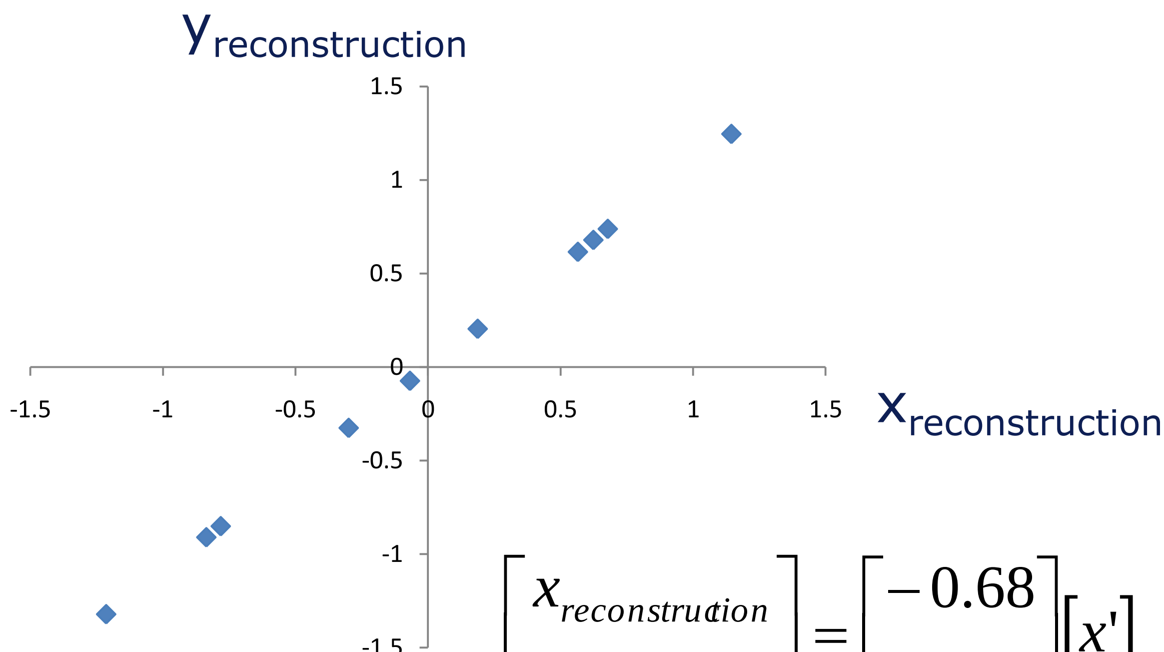
-.912949103

.0991094375

1.14457216

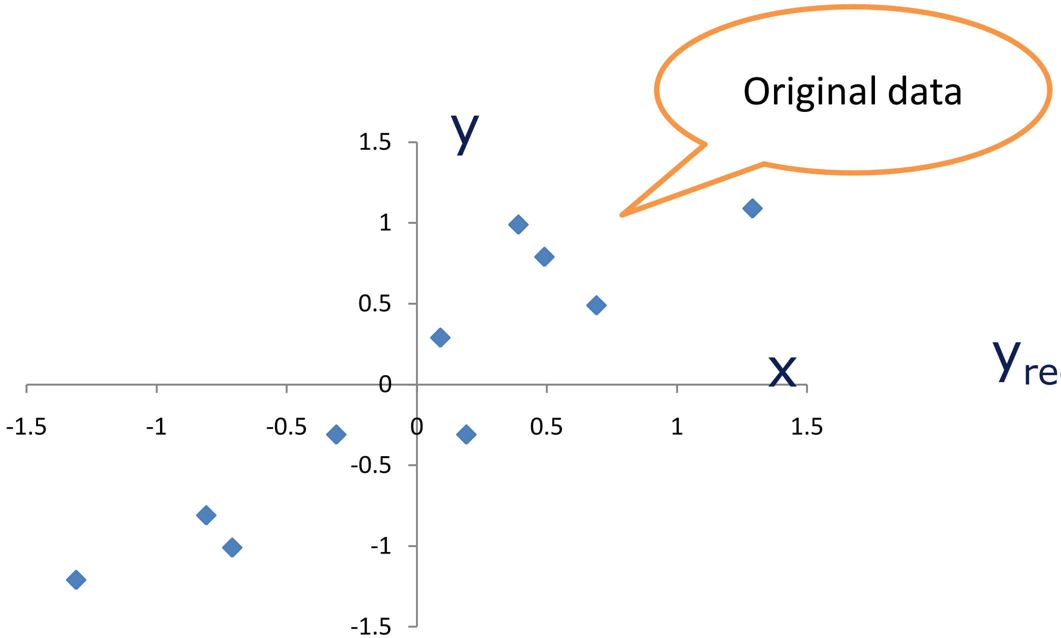
.438046137

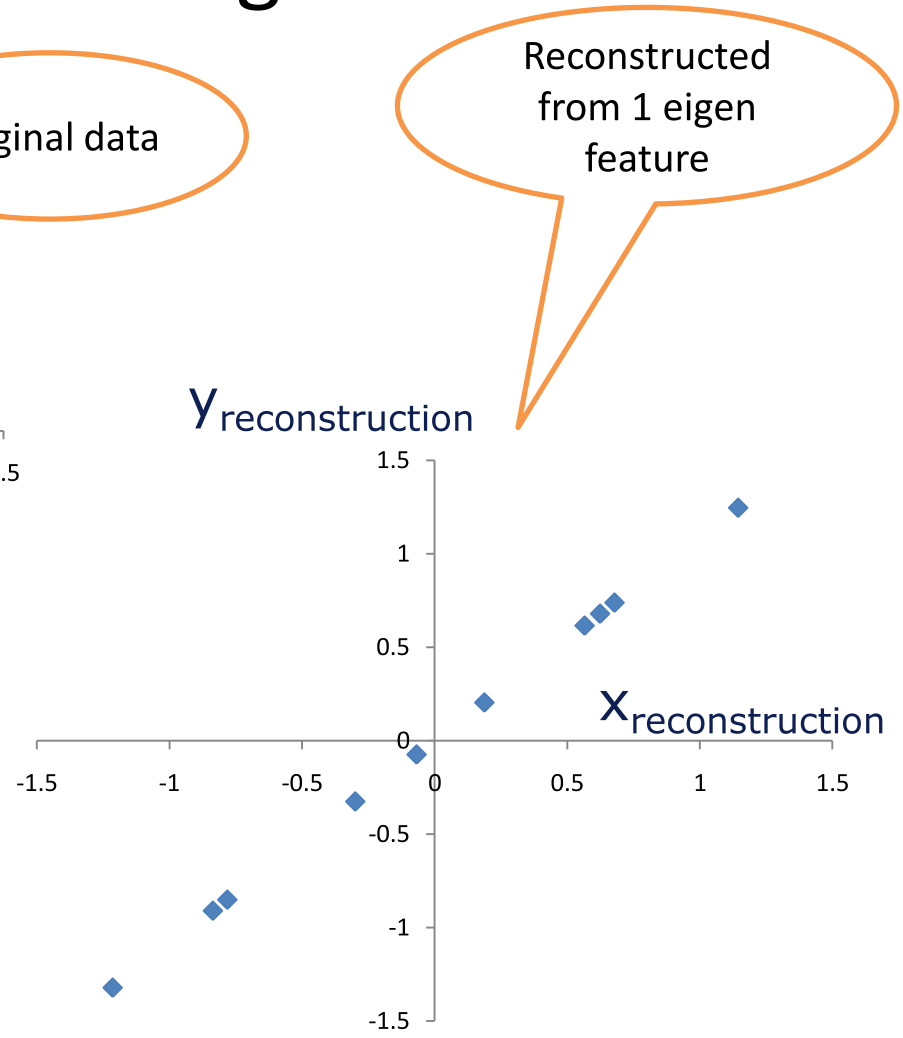
1.22382056



$$\begin{bmatrix} x_{reconstruction} \\ y_{reconstruction} \end{bmatrix} = \begin{bmatrix} -0.68 \\ -0.74 \end{bmatrix} [x']$$

Reconstruction of original Data





Feature Extraction/Eigen-features

Eigen
Feature
vector

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} p_{11} & \cdots & & p_{1m} \\ & \cdots & & \\ \ldots & \ldots & \ldots & \ldots \\ p_{m1} & \cdots & & p_{mm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix}$$

PCA Applications –General

1^{st} eigenvector

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ & & & \\ & & & \\ p_{m1} & p_{m2} & \cdots & p_{mm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \vdots \\ x_m \end{bmatrix}$$

$$Y = PX \qquad X = P^T Y$$

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} p_{11} & p_{21} & \cdots & p_{m1} \\ p_{12} & p_{22} & \cdots & p_{m2} \\ & & & \\ & & & \\ p_{1m} & p_{2m} & \cdots & p_{mm} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_i \\ \vdots \\ y_m \end{bmatrix}$$

m^{th} eigenvector

- Data compression/dimensionality reduction

PCA Applications -General

- Data compression/dimensionality reduction

$$p_i = \begin{bmatrix} p_{i1} & p_{i2} & \cdots & p_{im} \end{bmatrix}$$

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_i \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} y_1 p_1^T + y_2 p_2^T + \cdots + y_m p_m^T \end{bmatrix}$$

PCA Applications -General

- Data compression/dimensionality reduction
- Reduce the number of features needed for effective data representation by discarding those features having small variances
- The most interesting dynamics occur only in the first *l* dimensions (*l* << m).

$$\hat{X} = \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \vdots \\ \vdots \\ \hat{x}_m \end{bmatrix} = \begin{bmatrix} p_{11} & p_{21} & & p_{l1} \\ p_{12} & p_{22} & & p_{l2} \\ & & & \\ & & & \\ p_{1m} & p_{2m} & & p_{lm} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_l \end{bmatrix} = \left[y_1 p_1^T + y_2 p_2^T + \cdots + y_l p_l^T \right]$$

$$p_i = \left[p_{i1} \quad p_{i2} \quad \cdots \quad p_{im} \right]$$

$$X = \left[y_1 p_1^T + y_2 p_2^T + \cdots + y_m p_m^T \right]$$

PCA Applications -General

- Data compression/dimensionality reducti
- Reduce the number of features needed by discarding those features having small v
- The most interesting dynamics occur only i first *l* dimensions (*l* << m).

We know what can be thrown away; or do we?

$$\hat{X} = \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \vdots \\ \vdots \\ \hat{x}_m \end{bmatrix} = \begin{bmatrix} p_{11} & p_{21} & & p_{l1} \\ p_{12} & p_{22} & & p_{l2} \\ & & & \\ & & & \\ p_{1m} & p_{2m} & & p_{lm} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_l \end{bmatrix} = \left[y_1 p_1^T + y_2 p_2^T + \cdots + y_l p_l^T \right]$$

$$p_i = \left[p_{i1} \quad p_{i2} \quad \cdots \quad p_{im} \right] \qquad X = \left[y_1 p_1^T + y_2 p_2^T + \cdots + y_m p_m^T \right]$$

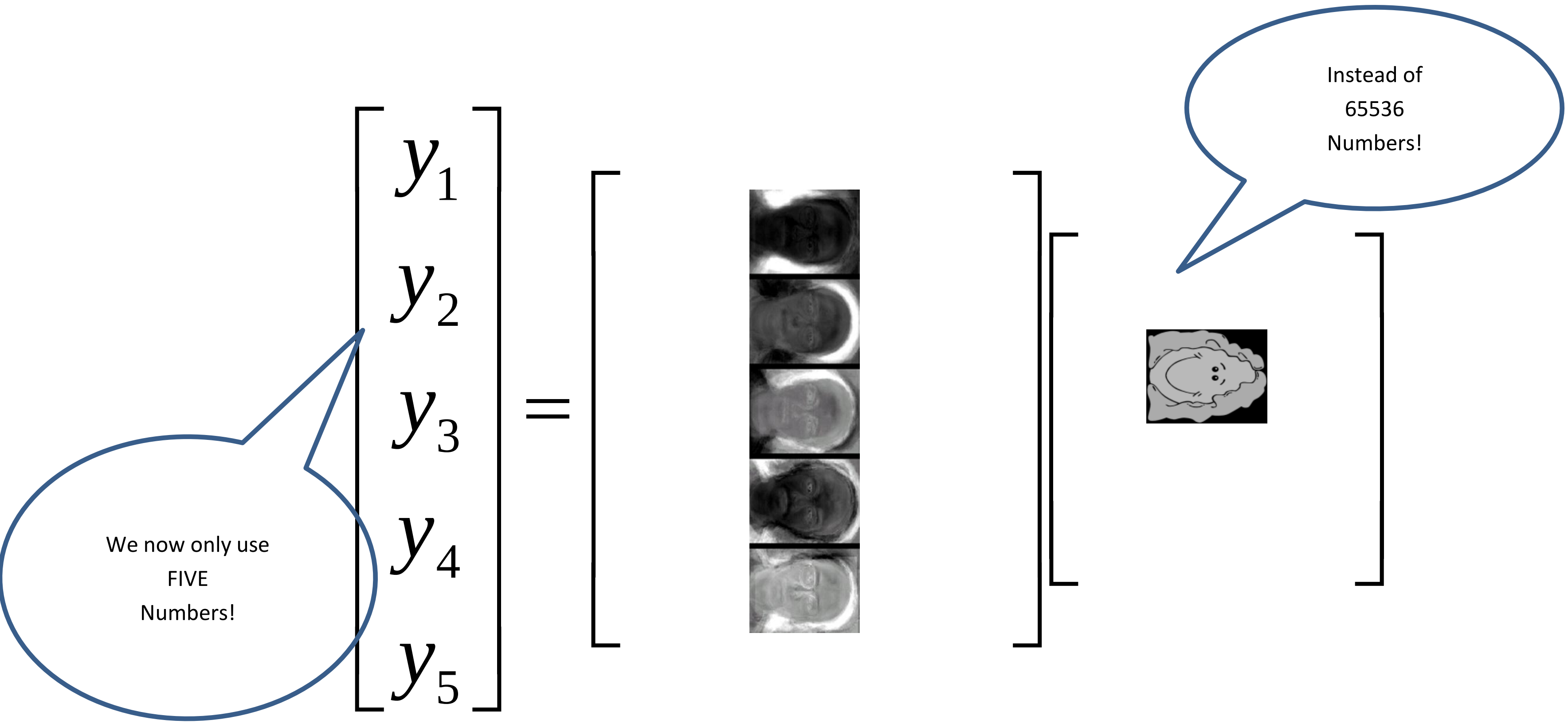
Eigenface Example

- A 256x256 face image, 65536 dimensional vector, X, representing the face images with much lower dimensional vectors for analysis and recognition
  - **Compute the covariance matrix, find its eigenvector and eigenvalue**
  - **Throw away eigenvectors corresponding to small eigenvalues, and keep the first *l* (*l* << m) principal components (eigenvectors)**

p_1 p_2 p_3 p_4 p_5

Eigenface Example

- A 256x256 face image, 65536 dimensional vector, X, representing the face images with much lower dimensional vectors for analysis and recognition



Eigen Analysis - General

– The same principle can be applied to the analysis of many other data types

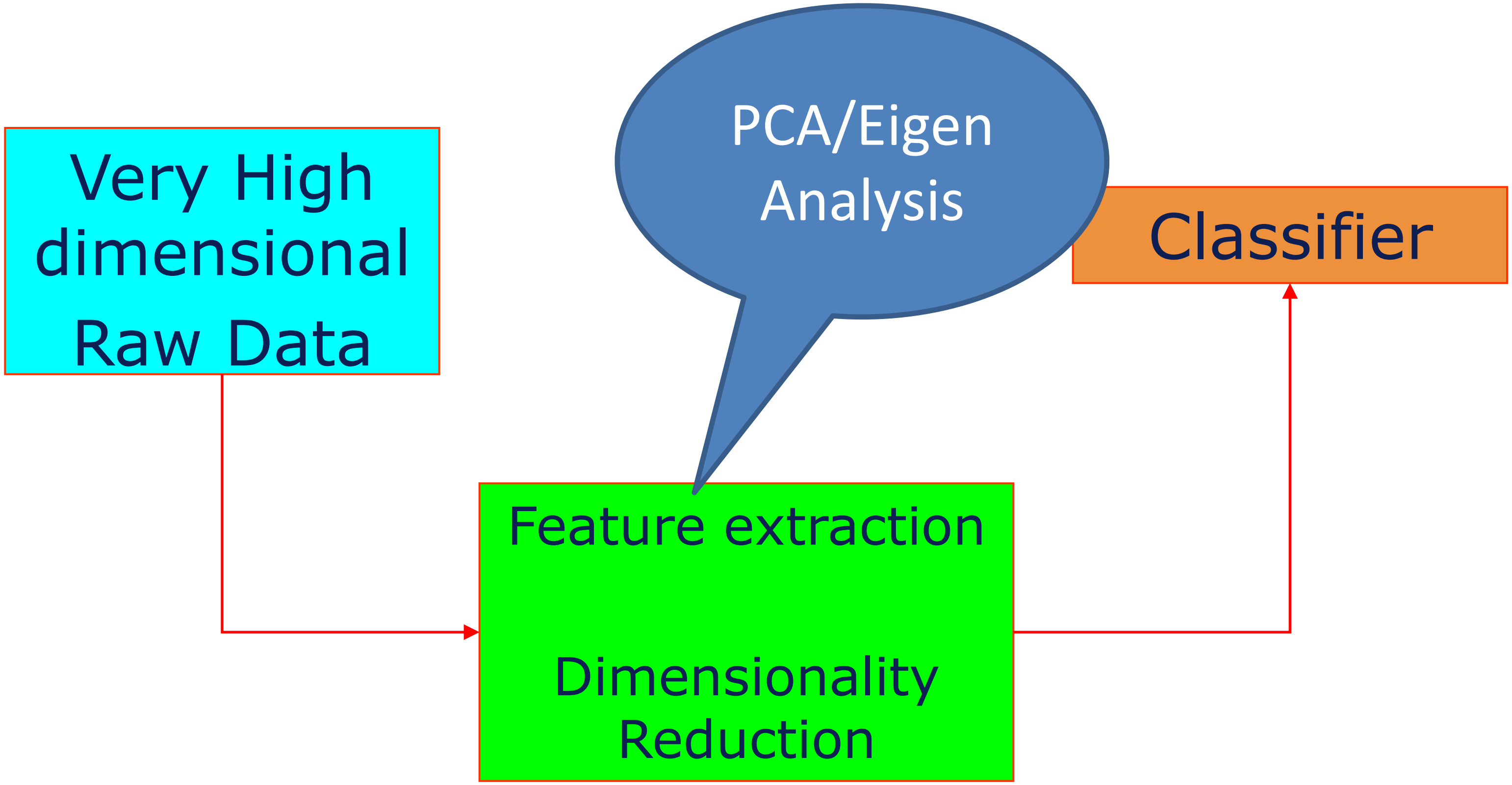
Reduce the dimensionality of biomarkers for analysis and classification

Raw data representation

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & & a_{1N} \\ & \cdots & & \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & \cdots & & a_{nN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_N \end{bmatrix}$$

Processing Methods

- General framework



PCA

- Some remarks about PCA
  - PCA computes projection directions in which variances of the data can be ranked
  - The first few principal components capture the most "energy" or largest variance of the data
  - In classification/recognition tasks, which principal component is more discriminative is unknown

PCA

- Some remarks about PCA
  - Traditional popular practice is to use the first few principal components to represent the original data.
  - However, the subspace spanned by the first few principal components is not necessarily the most discriminative.
  - Therefore, throwing away the principal components with small variances may not be a good idea!