PCA & LDA

Dimensionality Reduction in ML

Dimensionality Reduction

- Many modern data domains involve huge numbers of features / dimensions
  - Documents: thousands of words, millions of bigrams
  - Images: thousands to millions of pixels
  - Genomics: thousands of genes, millions of DNA polymorphisms

Why to reduce dimensions?

- High dimensionality has many costs
  - Redundant and irrelevant features degrade performance of some ML algorithms
  - Difficulty in interpretation and visualization
  - Computation may become infeasible
    - what if your algorithm scales as $O(n^3)$?
  - Curse of dimensionality

A Toy Example

	Educations	Jobs opportunities	Safety	Air quality	Entertainment	Connectivity	Foods
City-1	8	7	7	9	8	7	8
City-2	6	8	7	6	8	9	8
City-3	9	10	7	8	8	4	8
City-4	4	4	6	2	8	5	7
City-5	3	6	6	5	8	9	8
City-6	2	7	6	3	8	4	7
City-7	7	8	7	8	8	6	8
City-8	8	3	6	3	8	8	7
City-9	5	9	7	7	8	5	7
City-10	6	8	6	9	8	9	8

Another example

	Educations	Jobs opportunities	Safety	Air quality	Entertainment	Connectivity	Foods
City-1	8	7	7	9	8	7	8
City-2	6	8	7	6	8	9	8
City-3	9	10	7	8	8	4	8
City-4	4	4	6	2	8	5	7
City-5	3	6	6	5	8	9	8
City-6	2	7	6	3	8	4	7
City-7	7	8	7	8	8	6	8
City-8	8	3	6	3	8	8	7
City-9	5	9	7	7	8	5	7
City-10	6	8	6	9	8	9	8

Dimension reduction

- **Two approaches to perform dim. reduction $\Re^N \rightarrow \Re^M$ (M<N)**
  - **Feature selection**: choosing a subset of all the features

$$[x_1\ x_2 \ldots x_N] \xrightarrow{\text{feature selection}} [x_{i_1}\ x_{i_2} \ldots x_{i_M}]$$

  - **Feature extraction**: creating new features by combining existing ones

$$[x_1\ x_2 \ldots x_N] \xrightarrow{\text{feature extraction}} [y_1\ y_2 \ldots y_M] = f([x_{i_1}\ x_{i_2} \ldots x_{i_M}])$$

    - In either case, the goal is to find a low-dimensional representation of the data that preserves (most of) the information or structure in the data

- **Linear feature extraction**
  - The "optimal" mapping y=f(x) is, in general, a non-linear function whose form is problem-dependent
    - Hence, feature extraction is commonly limited to linear projections **y=Wx**

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_N \end{bmatrix} \xrightarrow{\text{linear feature extraction}} \begin{bmatrix} y_1 \\ y_2 \\ \\ y_M \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{M1} & w_{M2} & & w_{MN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \\ x_N \end{bmatrix}$$

Models

- Linear methods
  - Principal component analysis (PCA)
  - Multidimensional scaling (MDS)
  - Independent component analysis (ICA)

- Nonlinear methods
  - Kernel PCA
  - Locally linear embedding (LLE)
  - Laplacian eigenmaps (LEM)
  - Semidefinite embedding (SDE)

Statistics Preliminaries

Mean

- Let X_1,X_2, , … *X_n be n observations of a random variable X*

$$\mu = \overline{X} = E\left[X\right] = \frac{1}{n}\sum_{i=1}^{n} X_i$$

- Mean is a measure of central tendency (others are mode and median)

Standard Deviation

- Measure of variability (square root of variance)

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(X_i - \mu\right)^2}$$

Statistics Preliminaries

- **Covariance**
  - A measure of how two variables change together

$$\Sigma_{XY} = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - \mu_X\right)\left(Y_i - \mu_Y\right)^T$$

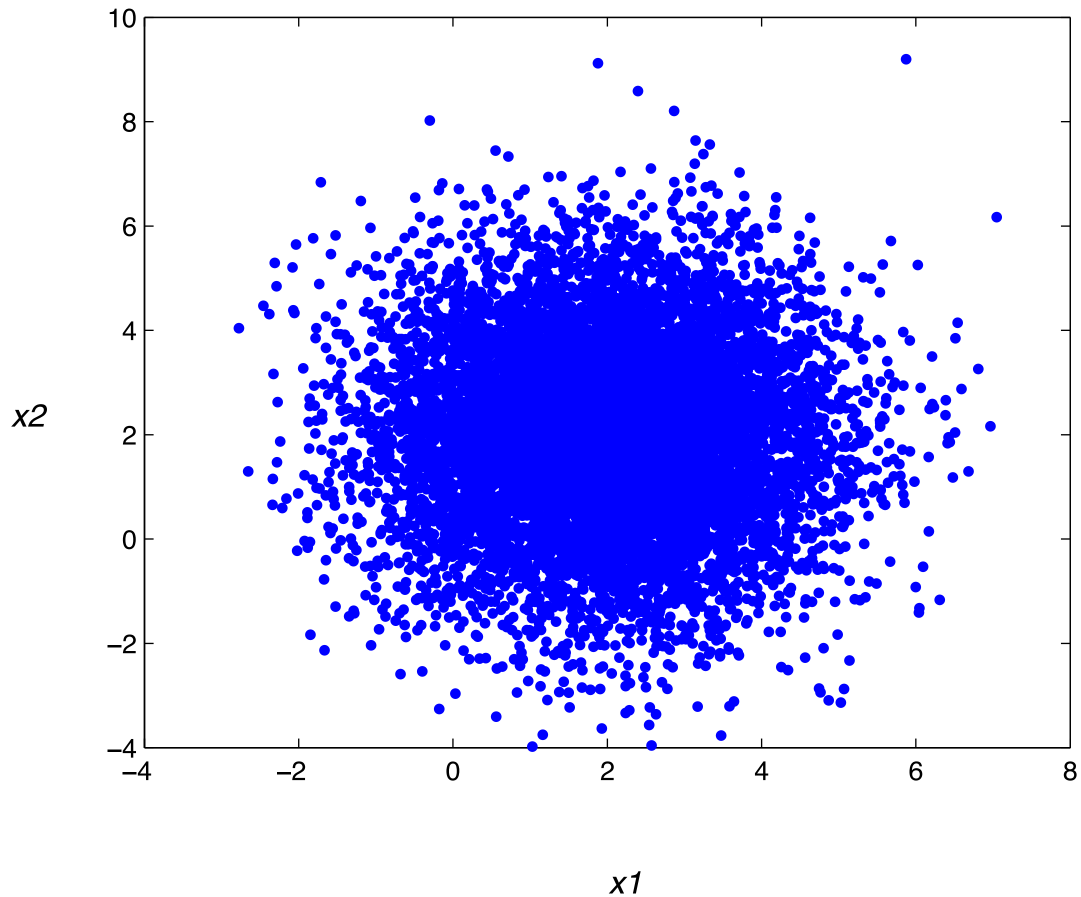
$$\Sigma_{XY} = E\left[\left(X - E[X]\right)\left(Y - E[Y]\right)\right]$$

- **Covariance Matrix**

$$\Sigma = \begin{pmatrix} \sigma_{X1}^2 & \Sigma_{X1X2} & \dots & \Sigma_{X1Xd} \\ \Sigma_{X1X2} & \sigma_{X2}^2 & \dots & \Sigma_{X2Xd} \\ . & & & . \\ . & & \dots & . \\ . & & & . \\ \Sigma_{X1Xd} & \Sigma_{X2Xd} & \dots & \sigma_{Xd}^2 \end{pmatrix}$$

Statistics Preliminaries – An Example

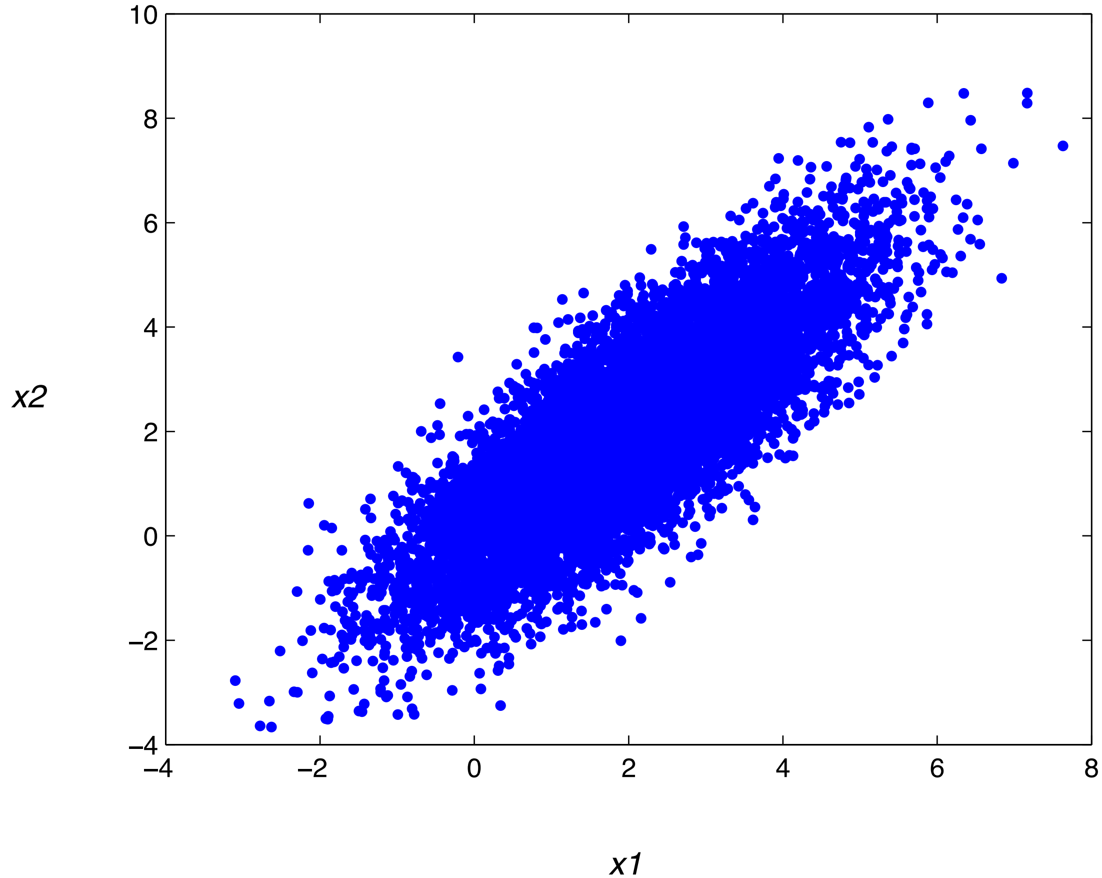
$$\mu = \begin{pmatrix} 2 \\ 3 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}$$



PCA – An Example

$$\mu = \begin{pmatrix} 2 \\ 3 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 2 & 2 \\ 2 & 3 \end{pmatrix}$$

Can you find a vector that would approximate this 2-D space?

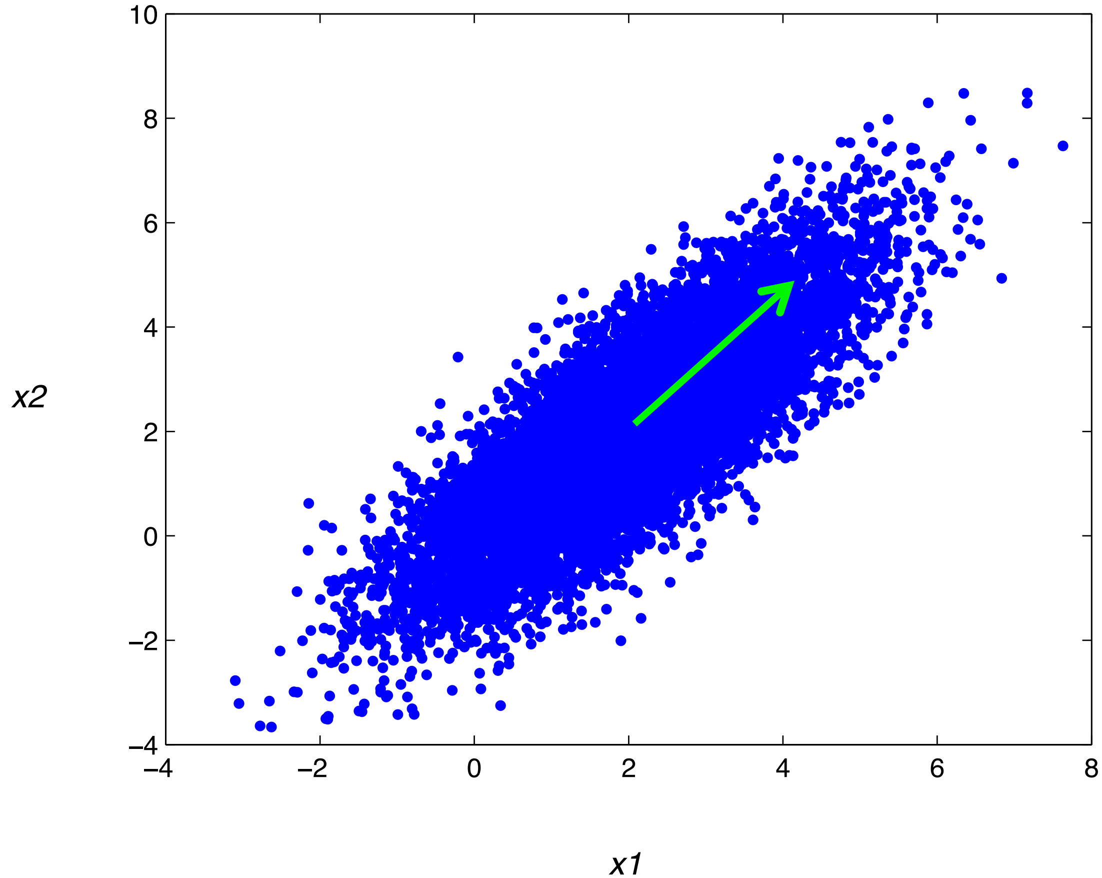


PCA – Let's build some intuition

$$\mu = \begin{pmatrix} 2 \\ 3 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 2 & 2 \\ 2 & 3 \end{pmatrix}$$

Can you find a vector that would approximate this 2-D space?

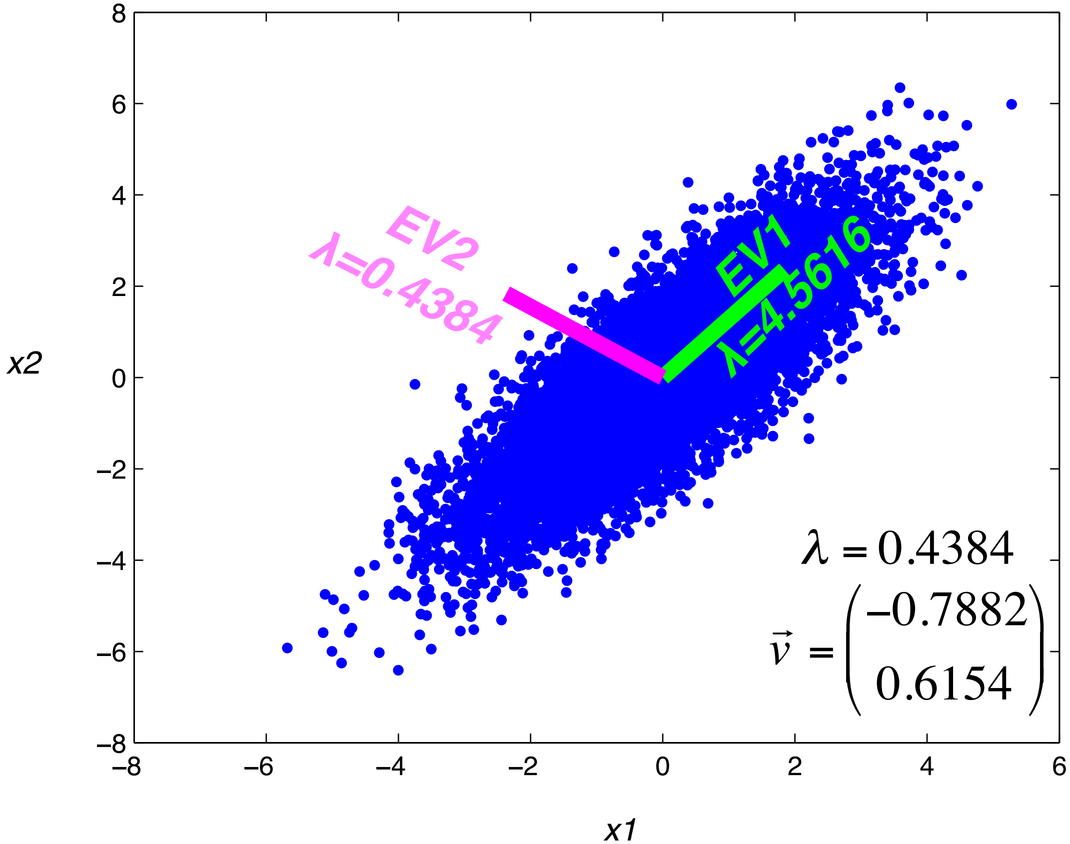
Green vector right?



PCA – Let's build some intuition

$$\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} 2 & 2 \\ 2 & 3 \end{pmatrix}$$

It would be nice to
diagonalize the covariance matrix
then you have only think about variance
Think eigenvectors of covariance matrix



$$\lambda = 0.4384 \qquad \vec{v} = \begin{pmatrix} -0.7882 \\ 0.6154 \end{pmatrix}$$

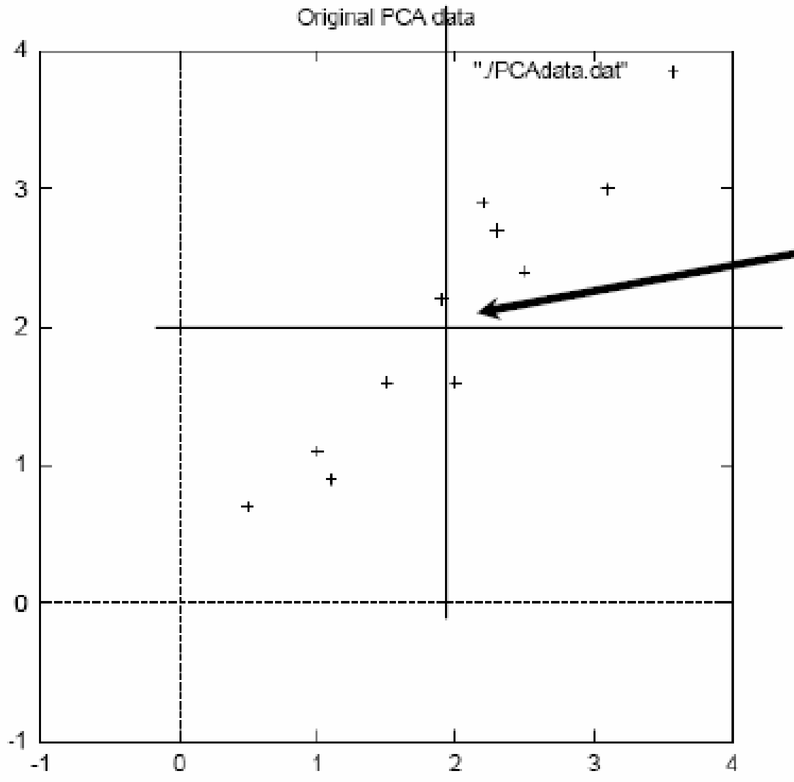
$$\lambda = 4.5616 \qquad \vec{v} = \begin{pmatrix} 0.6154 \\ 0.7882 \end{pmatrix}$$

A 2-D Example
Step-1

http://kybele.psych.cornell.edu/~edelman/Psych-465-Spring-2003/PCA-tutorial.pdf

- DATA:

x	y
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9



mean

this becomes the new origin of the data from now on

A 2-D Example

Step-2

- Calculate the covariance matrix

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

Step-3

Calculate the eigenvectors and eigenvalues of the covariance matrix

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

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

A 2-D Example

Step-4

- 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:

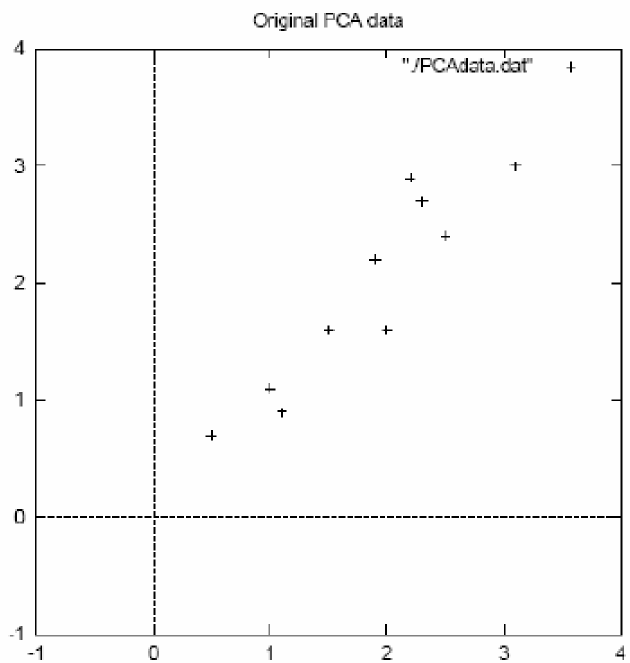
  $$\begin{pmatrix} -.677873399 & -.735178656 \\ -.735178656 & .677873399 \end{pmatrix}$$

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

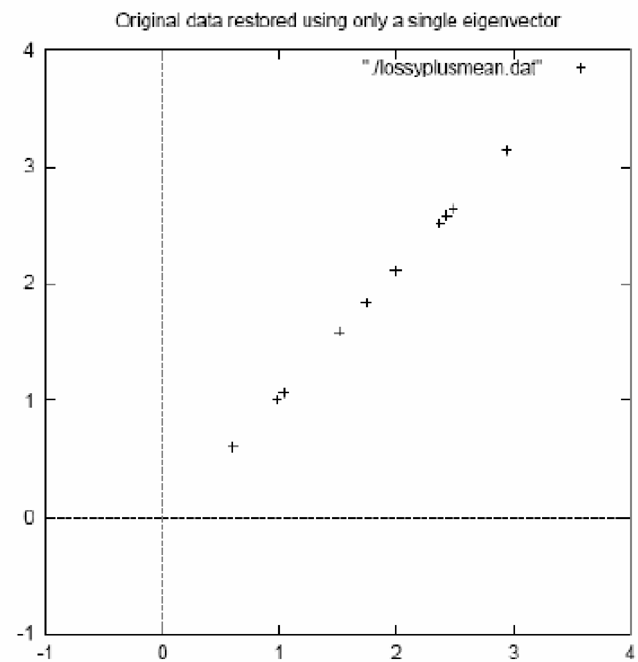
  $$\begin{pmatrix} -.677873399 \\ -.735178656 \end{pmatrix}$$

Transformed Data (Single eigenvector)

x
-.827970186
1.77758033
-.992197494
-.274210416
-1.67580142
-.912949103
.0991094375
1.14457216
.438046137
1.22382056



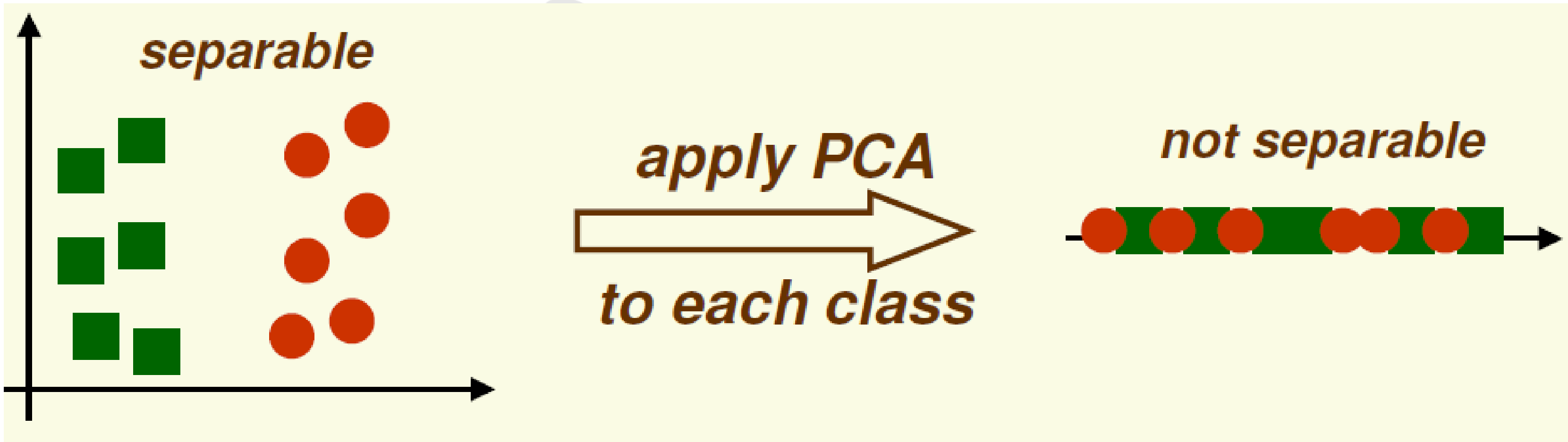
2D point cloud



Approximation using one eigenvector basis

Limitations of PCA?

- PCA projects the data in lower dimensional space spanned by the directions of maximum variance (most accurate *data)*



- However the directions of maximum variance may be useless for classification (linear…). LDA overcomes this limitation of PCA.

LDA→ Lower dimensions + preserve classification property

Image Source: www.csd.uwo.ca/~oveksler

Bias

- **LDA projects the data in lower dimensional space in such a way that patterns from different classes are well separated.**

- How to find such a line (in 2D), and lower dimensional space in general?

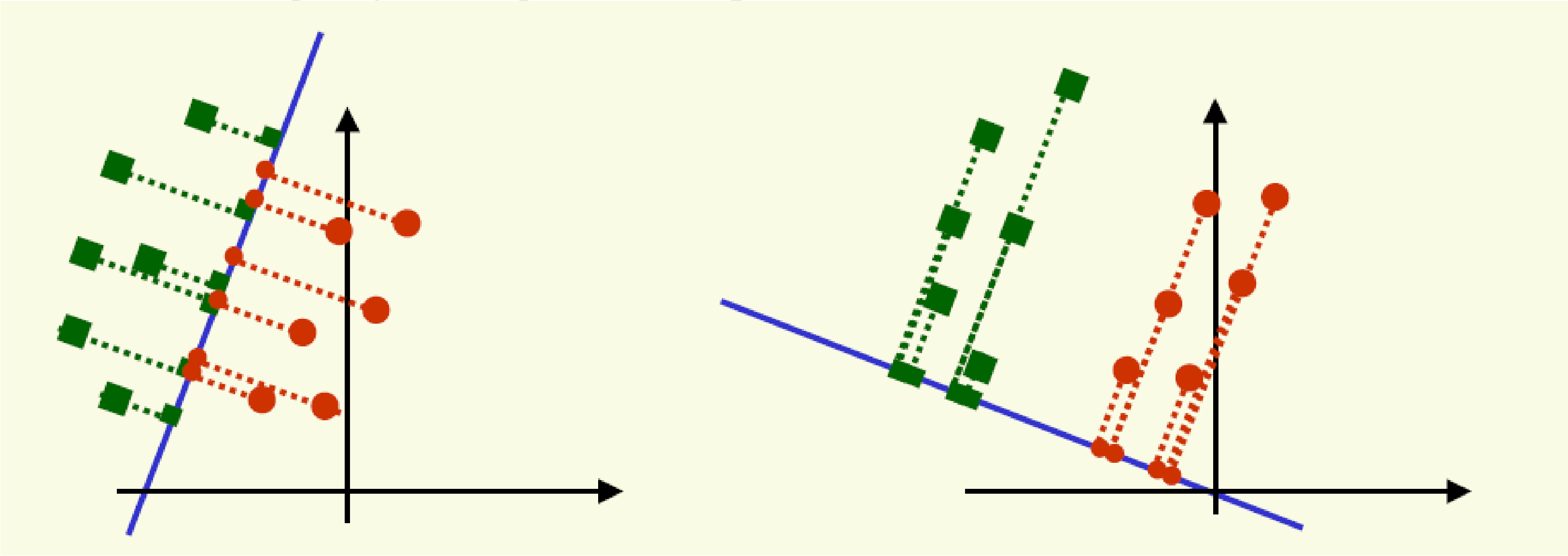
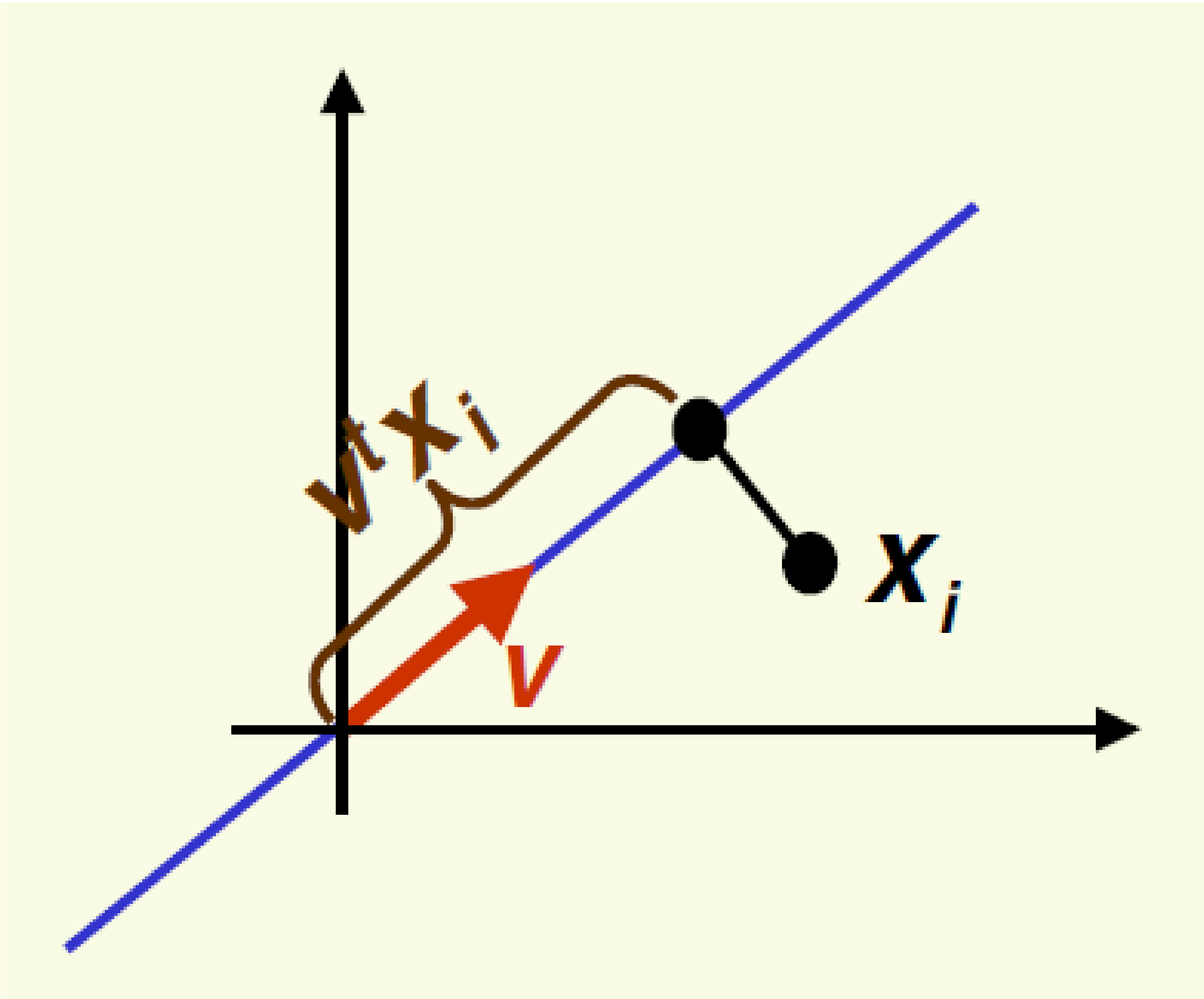


Image Source: www.csd.uwo.ca/~oveksler

Projection of a feature vector on to a space



LDA:

Input: We have a binary classification problem, where patterns are coming from a d-dimensional space.

→ n_1 samples are from class-1.

→ n_2 samples are from class-2.

$$n_1 + n_2 = n$$

Samples are given by vectors

$$x_1, x_2, x_3, \ldots, x_n$$

LDA...

* Let μ_1 and μ_2 be the means of patterns from class-1 and class-2, respectively in original space.

* Moreover, $\tilde{\mu}_1$ and $\tilde{\mu}_2$ are mean of patterns after projection.

Then,

$$\tilde{\mu}_1 = \frac{1}{n_1} \sum_{x_i \in C_1} V^T x_i = V^T \left(\frac{1}{n_1} \sum x_i \right) = \underline{V^T \mu_1}$$

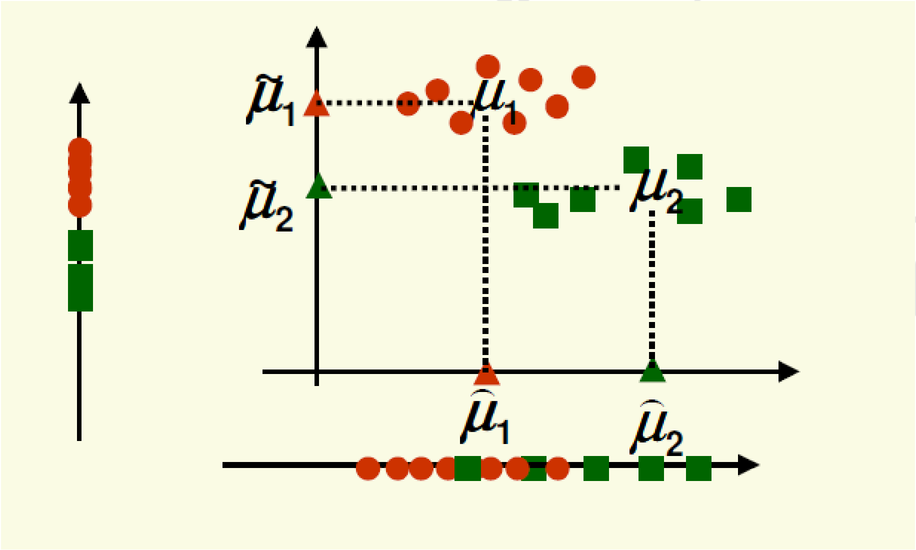
IIly $\underline{\tilde{\mu}_2 = V^T \mu_2}$

LDA..

Our aim is to have the patterns of these two classes away from each other after projection.

So maximizing distance between their means may be a good measure:

→ The larger $|\tilde{\mu}_1 - \tilde{\mu}_2|$, the better is separation.



* The vertical axis is a better direction

* However, $|\hat{\mu}_1 - \hat{\mu}_2| > |\tilde{\mu}_1 - \tilde{\mu}_2|$

LDA

Therefore, we are missing something. Yes, we are not considering variance of the classes.

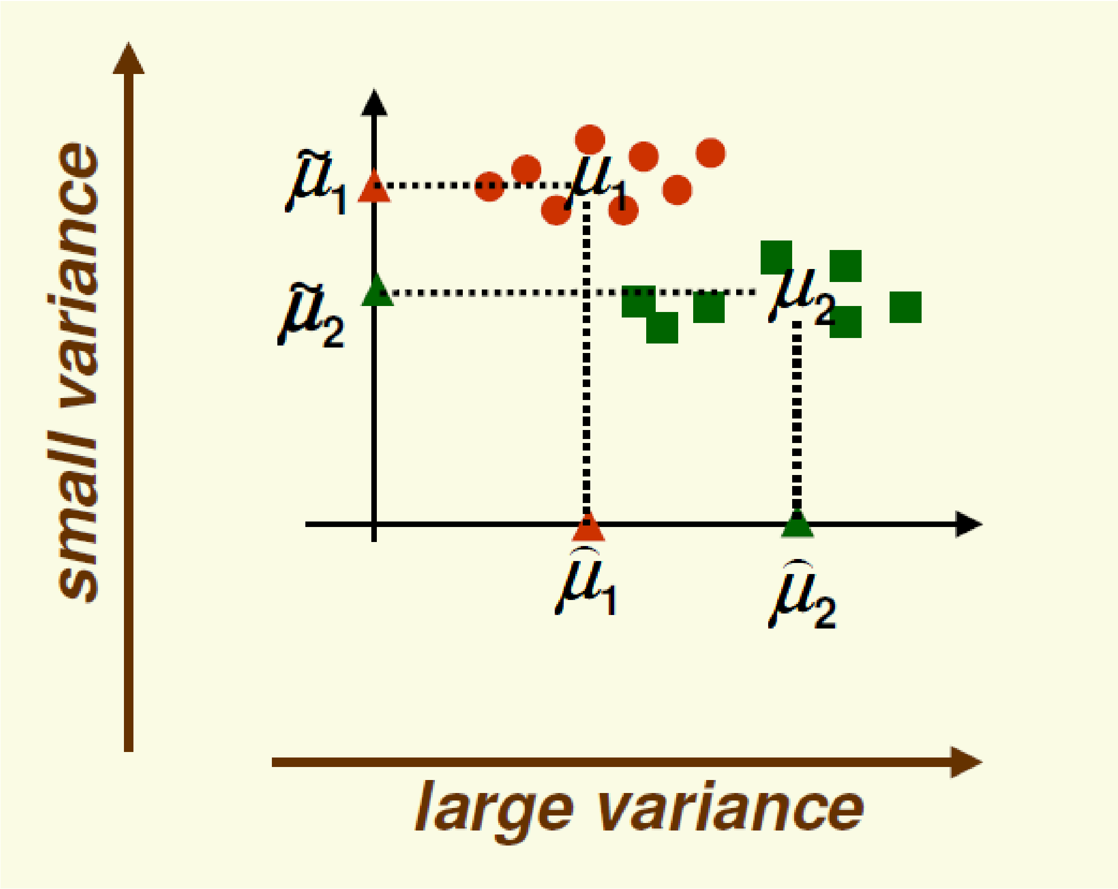


Image Source: www.csd.uwo.ca/~oveksler

LDA...

Define the scatters as follows:

$$\tilde{s}_1^2 = \sum_{y_i \in C_1} (y_i - \tilde{\mu}_1)^2 \quad \text{for } \underline{\text{class - 1}}$$

and $$\tilde{s}_2 = \sum_{y_i \in C_2} (y_i - \tilde{\mu}_2)^2 \quad \text{for } \underline{\text{class - 2}}$$

where, $y_i = V^T x_i$

Then, we need to Maximize $\underline{|\tilde{\mu}_1 - \tilde{\mu}_2|}$ and minimize $\underline{\tilde{s}_1^2}$ and $\underline{\tilde{s}_2^2}$

Objective function

want projected means are far from each other

$$J(v) = \frac{\overbrace{(\tilde{\mu}_1 - \tilde{\mu}_2)^2}^{\text{want projected means are far from each other}}}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

want scatter in class 1 is as small as possible, i.e. samples of class 1 cluster around the projected mean $\tilde{\mu}_1$

want scatter in class 2 is as small as possible, i.e. samples of class 2 cluster around the projected mean $\tilde{\mu}_2$

LDA

Therefore,

Our objective function can be rewritten as

$$\underset{V}{\arg\max}\ J(V) = \frac{(\tilde{\mu}_1 - \tilde{\mu}_2)^2}{\tilde{s}_1^2 + \tilde{s}_2^2} = \frac{V^T S_B V}{V^T S_W V}$$

Now

$$\frac{dJ(V)}{dV} = 0 \Rightarrow S_B V - \frac{\overbrace{V^T S_B V}^{} (S_W V)}{\underbrace{V^T S_W V}_{\to \lambda}} = 0$$

$$\Rightarrow S_B V = \lambda S_W V$$

$$\Rightarrow \boxed{(S_W^{-1} S_B) V = \lambda V}$$ Eigenvalue Problem

Simplification

$$\boxed{S_W^{-1} S_B V = \lambda V}$$

* But $S_B X$ for any vector X, and $(\mu_1 - \mu_2)$ are parallel, i.e.

$$S_B X = (\mu_1 - \mu_2)\underbrace{(\mu_1 - \mu_2)^T X}_{\to \alpha} = \underline{\alpha(\mu_1 - \mu_2)}$$

Hences

$$\boxed{V = S_W^{-1}(\mu_1 - \mu_2)} \quad \text{———} \quad (*)$$

<u>Summary</u>:- From $x_1, x_2, \cdots, x_n$, calculate S_W and $(\mu_1 - \mu_2) \Rightarrow$ Then $(*)$ gives you direction of the line.

Example

Consider the following data

$C_1: [(1,2),(2,3),(3,3),(4,5),(5,5)]^T$

$C_2: [(1,0),(2,1),(3,1),(3,2),(5,3),(6,5)]^T$

Then $\mu_1 = [3 \quad 3.6]^T$ and $\mu_2 = [3.3 \quad 2]^T$

Also,

$$S_1 = 4 * \mathrm{Cov}(C_1) = \begin{bmatrix} 10 & 8 \\ 8 & 7.2 \end{bmatrix}$$

$$S_2 = 4 * \mathrm{Cov}(C_2) = \begin{bmatrix} 17.3 & 16 \\ 16 & 16 \end{bmatrix}$$

$$S_W = S_1 + S_2 = \begin{bmatrix} 27.3 & 24 \\ 24 & 23.2 \end{bmatrix}$$

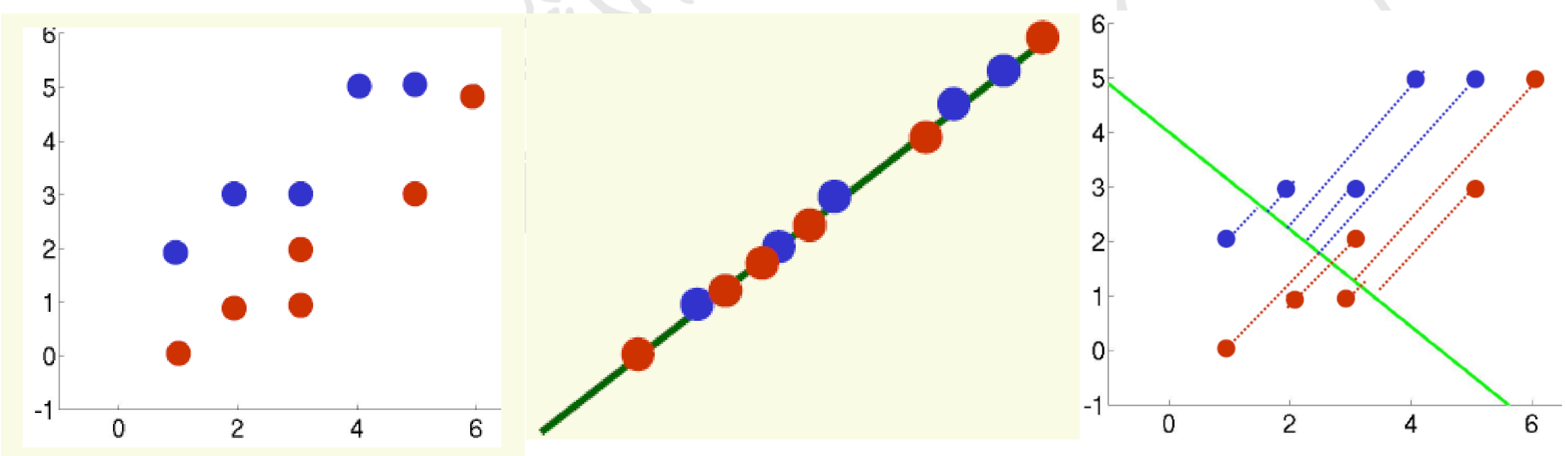
Example...

Hence, $$S_W^{-1} = \begin{bmatrix} .39 & -.41 \\ -.41 & .47 \end{bmatrix}$$

Therefore, the optimal line direction V

$$V = S_W^{-1}(\mu_1 - \mu_2) = \begin{bmatrix} -0.79 \\ 0.89 \end{bmatrix}$$

$$Y_1 = V^T C_1 \quad \text{and} \quad Y_2 = V^T C_2$$



THANKYOU