UVA CS 4774: Machine Learning

Lecture 14: Dimension Reduction / Principal Component Analysis (PCA)

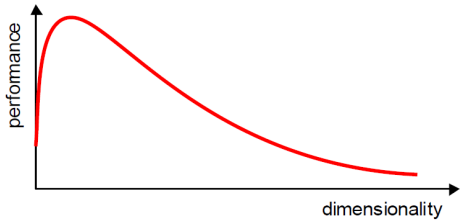
Dr. Yanjun Qi

University of Virginia

Department of Computer Science

Curse of Dimensionality

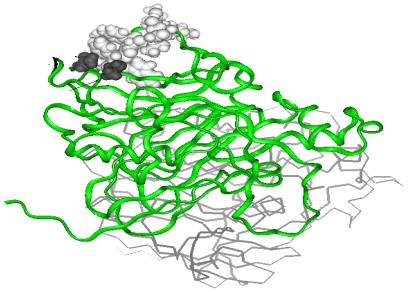
- Increasing the number of features will not always improve classification accuracy.



- In practice, the inclusion of more features might actually lead to worse performance.

- The number of training examples required increases exponentially with dimensionality **p**

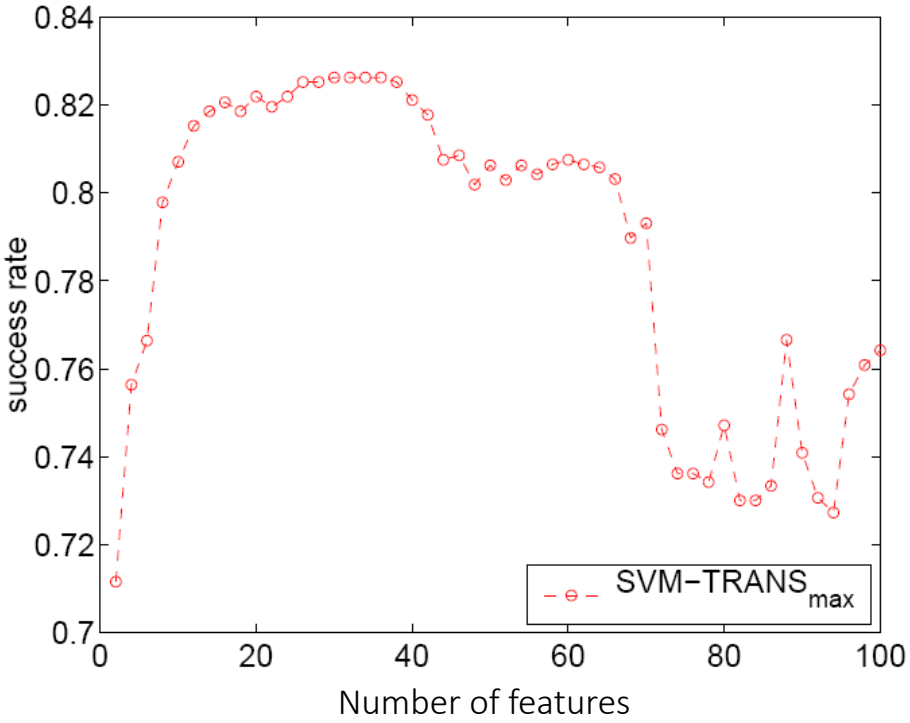
e.g., QSAR: Drug Screening



Binding to Thrombin
(DuPont Pharmaceuticals)

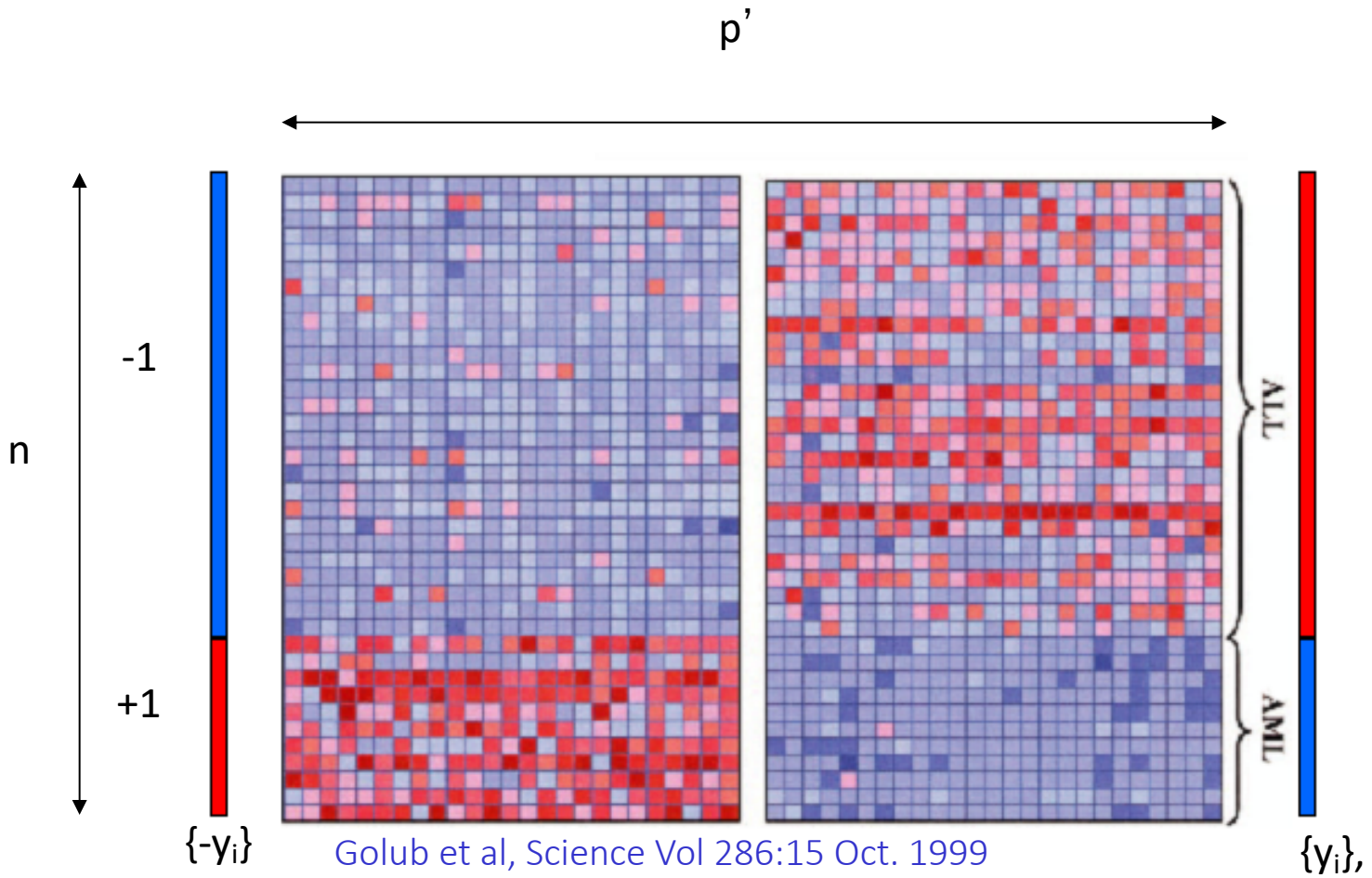
- 2543 compounds tested for their ability to bind to a target site on thrombin, a key receptor in blood clotting; 192 "active" (bind well); the rest "inactive". Training set (1909 compounds) more depleted in active compounds.

- 139,351 binary features, which describe three-dimensional properties of the molecule.



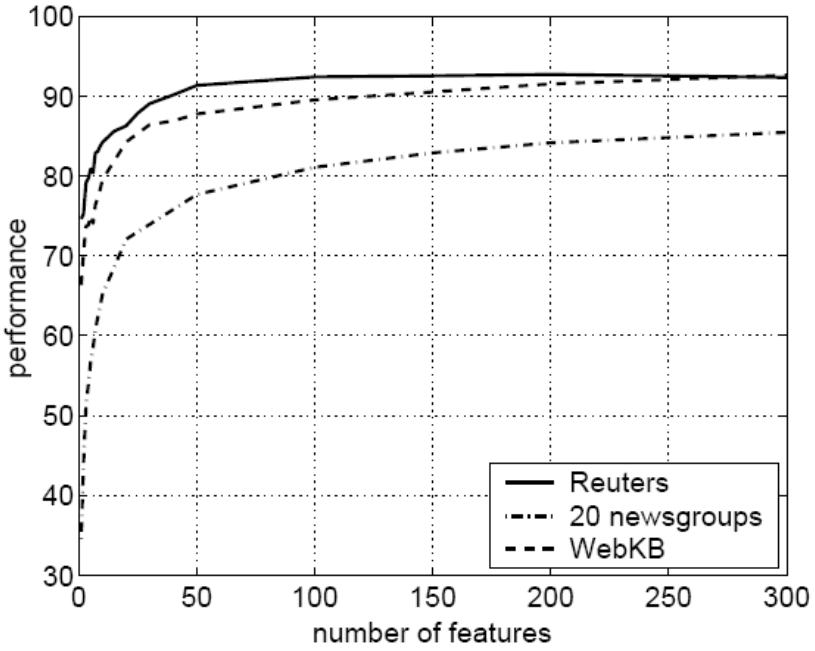
Weston et al, Bioinformatics, 2002

e.g., Leukemia Diagnosis



Golub et al, Science Vol 286:15 Oct. 1999

e.g., Text Categorization with many BOW featuers



Reuters: 21578 news wire, 114 semantic categories.

20 newsgroups: 19997 articles, 20 categories.

WebKB: 8282 web pages, 7 categories.

Bag-of-words: >100,000 features.

Bekkerman et al, JMLR, 2003

e.g., Movie Reviews and Revenues: An Experiment in Text Regression, Proceedings of HLT '10 (1.7k n / >3k features)

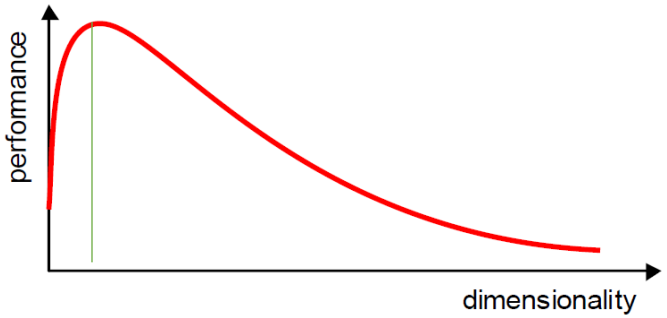
IV. Features	
I	Lexical n-grams (1,2,3)
II	Part-of-speech n-grams (1,2,3)
III	Dependency relations (nsubj,advmod,...)
Meta	U.S. origin, running time, budget (log), # of opening screens, genre, MPAA rating, holiday release (summer, Christmas, Memorial day,...), star power (Oscar winners, high-grossing actors)

e.g. counts of a ngram in the text

$n \approx 1700$ / $p > 30,000$

Dimensionality Reduction

- What is the objective?
  - Choose an optimum set of features of lower dimensionality to improve classification accuracy.



Dimension Reduction ➔ Simpler models

- Because:
  - Simpler to use (lower computational complexity)
  - Easier to train (needs less examples)
  - Less sensitive to noise
  - Easier to explain (more interpretable)
  - Generalizes better (lower variance)

Adapted from Dr. Christoph Eick Slides

Today: Dimensionality Reduction (Two Ways)

Feature extraction: finds a set of new features (i.e., through some mapping f()) from the existing features.

Feature selection: chooses a subset of the original features.

The mapping f() could be linear or non-linear

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ x_N \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_K \end{bmatrix}$$

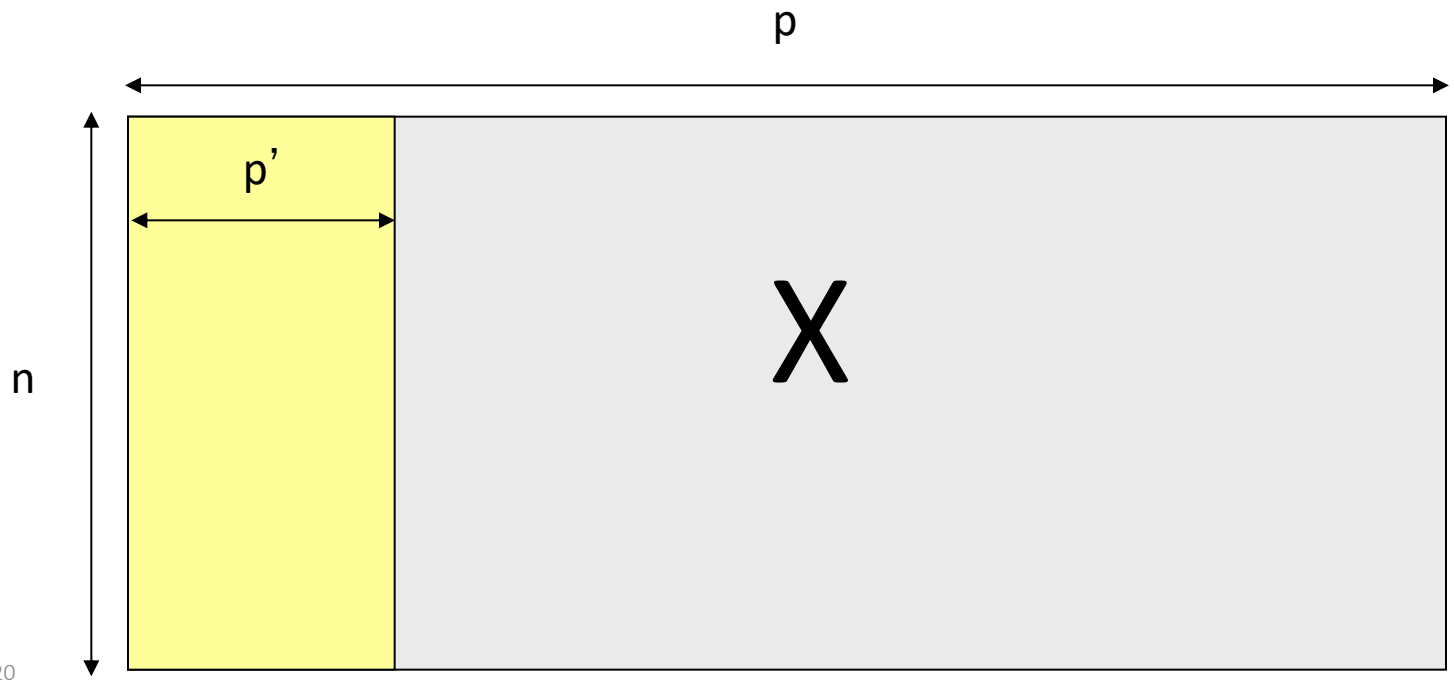
K<<N

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ x_N \end{bmatrix} \rightarrow \mathbf{y} = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ \cdot \\ \cdot \\ x_{i_K} \end{bmatrix}$$

K<<N

Feature Selection

- Select the most relevant ones to build better, faster, and easier to understand learning models.



From Dr. **Isabelle Guyon**

Summary: Feature Selection

- Filtering approach:

  ranks features or feature subsets independently of the predictor.
  - …using univariate methods: consider one variable at a time
  - …using multivariate methods: consider more than one variables at a time

- Wrapper approach:

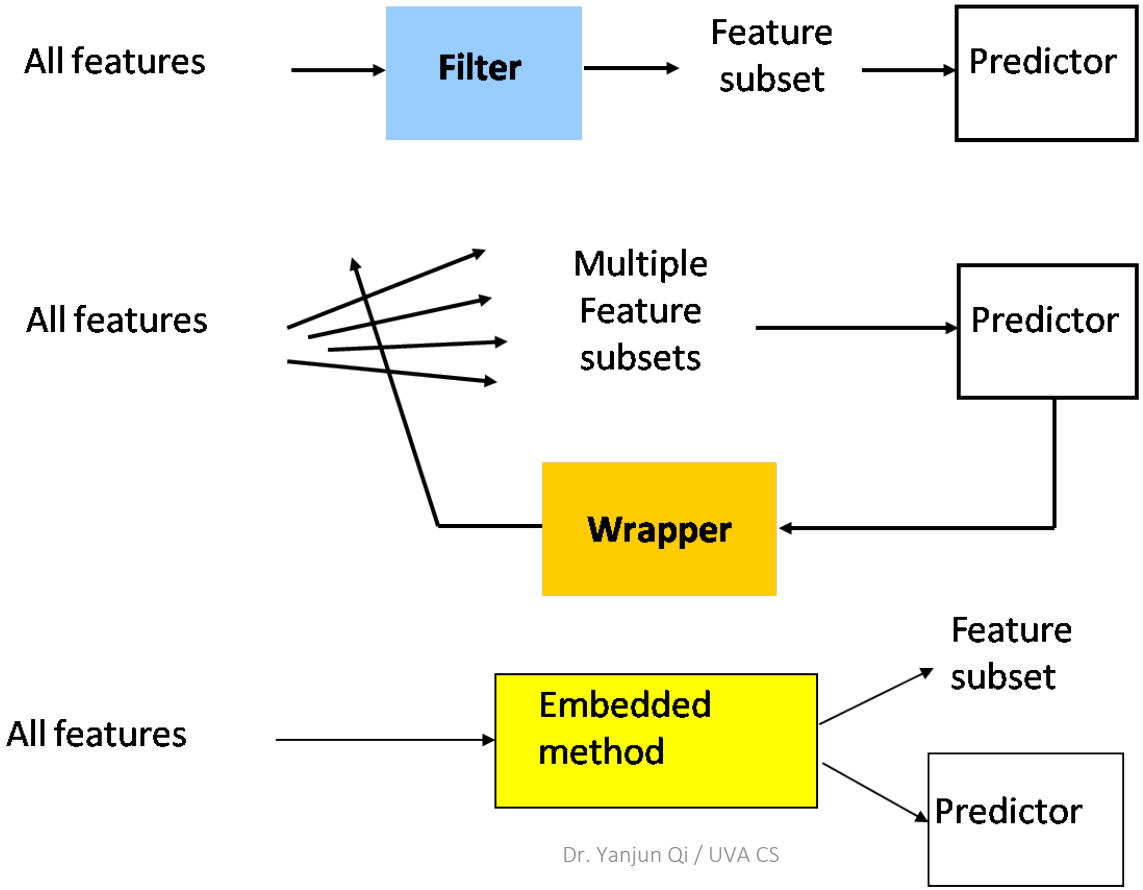
  uses a predictor to assess (many) features or feature subsets.

- Embedding approach:

  uses a predictor to build a (single) model with a subset of features that are internally selected.

Summary: filters vs. wrappers vs. embedding

- **Main goal**: rank subsets of useful features

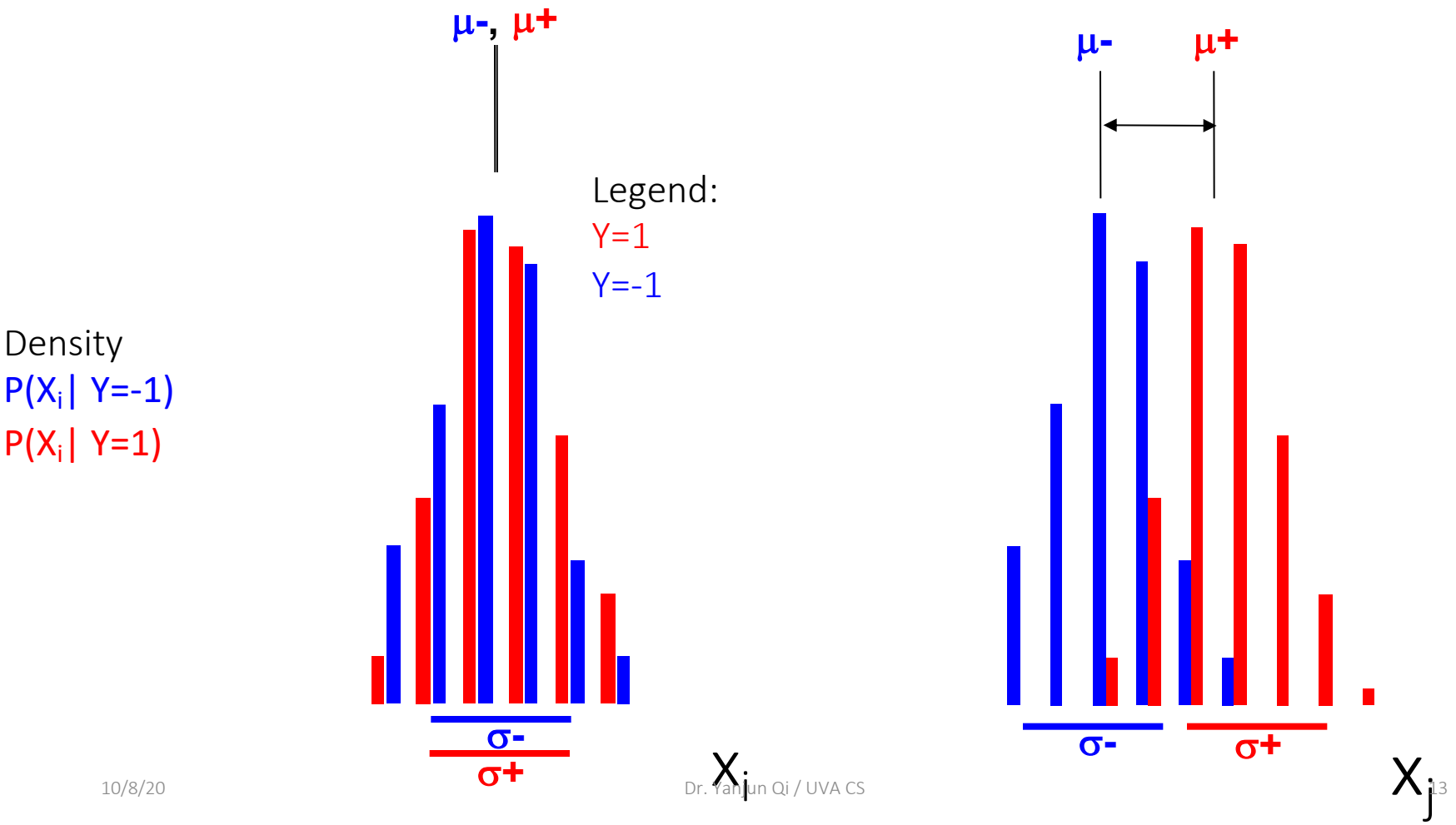


From Dr. **Isabelle Guyon**

(I) Filtering: univariate filtering

e.g. T-test

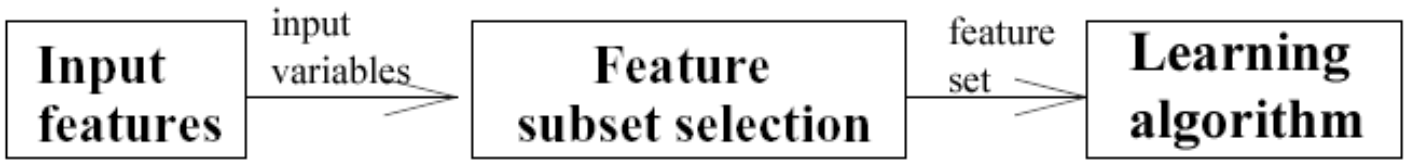
- Goal: determine the relevance of a given single feature for two classes of samples.



(I) Filtering : multi-variate:
Feature Subset Selection

Filter Methods

- Select subsets of variables as a pre-processing step, independently of the used classifier!!



- E.g. Group correlation
- E.g. Information theoretic filtering methods such as Markov blanket

(I) Filtering : Summary

Filter Methods

- usually fast
- provide generic selection of features, not tuned by given learner (universal)
- this is also often criticised (feature set not optimized for used learner)
- Often used as a preprocessing step for other methods

(I) Filtering : (many choices)

Method		X			Y			Comments
Name	Formula	B	M	C	B	M	C	
Bayesian accuracy	Eq. 3.1	+	s		+	s		Theoretically the golden standard, rescaled Bayesian relevance Eq. 3.2.
Balanced accuracy	Eq. 3.4	+	s		+	s		Average of sensitivity and specificity; used for unbalanced dataset, same as AUC for binary targets.
Bi-normal separation	Eq. 3.5	+	s		+	s		Used in information retrieval.
F-measure	Eq. 3.7	+	s		+	s		Harmonic of recall and precision, popular in information retrieval.
Odds ratio	Eq. 3.6	+	s		+	s		Popular in information retrieval.
Means separation	Eq. 3.10	+	i	+	+			Based on two class means, related to Fisher's criterion.
T-statistics	Eq. 3.11	+	i	+	+			Based also on the means separation.
Pearson correlation	Eq. 3.9	+	i	+	+	i	+	Linear correlation, significance test Eq. 3.12, or a permutation test.
Group correlation	Eq. 3.13	+	i	+	+	i	+	Pearson's coefficient for subset of features.
χ^2	Eq. 3.8	+	s		+	s		Results depend on the number of samples m.
Relief	Eq. 3.15	+	s	+	+	s	+	Family of methods, the formula is for a simplified version ReliefX, captures local correlations and feature interactions.
Separability Split Value	Eq. 3.41	+	s	+	+	s		Decision tree index.
Kolmogorov distance	Eq. 3.16	+	s	+	+	s	+	Difference between joint and product probabilities.
Bayesian measure	Eq. 3.16	+	s	+	+	s	+	Same as Vajda entropy Eq. 3.23 and Gini Eq. 3.39.
Kullback-Leibler divergence	Eq. 3.20	+	s	+	+	s	+	Equivalent to mutual information.
Jeffreys-Matusita distance	Eq. 3.22	+	s	+	+	s	+	Rarely used but worth trying.
Value Difference Metric	Eq. 3.22	+	s		+	s		Used for symbolic data in similarity-based methods, and symbolic feature-feature correlations.
Mutual Information	Eq. 3.29	+	s	+	+	s	+	Equivalent to information gain Eq. 3.30.
Information Gain Ratio	Eq. 3.32	+	s	+	+	s	+	Information gain divided by feature entropy, stable evaluation.
Symmetrical Uncertainty	Eq. 3.35	+	s	+	+	s	+	Low bias for multivalued features.
J-measure	Eq. 3.36	+	s	+	+	s	+	Measures information provided by a logical rule.
Weight of evidence	Eq. 3.37	+	s	+	+	s	+	So far rarely used.
MDL	Eq. 3.38	+	s		+	s		Low bias for multivalued features.

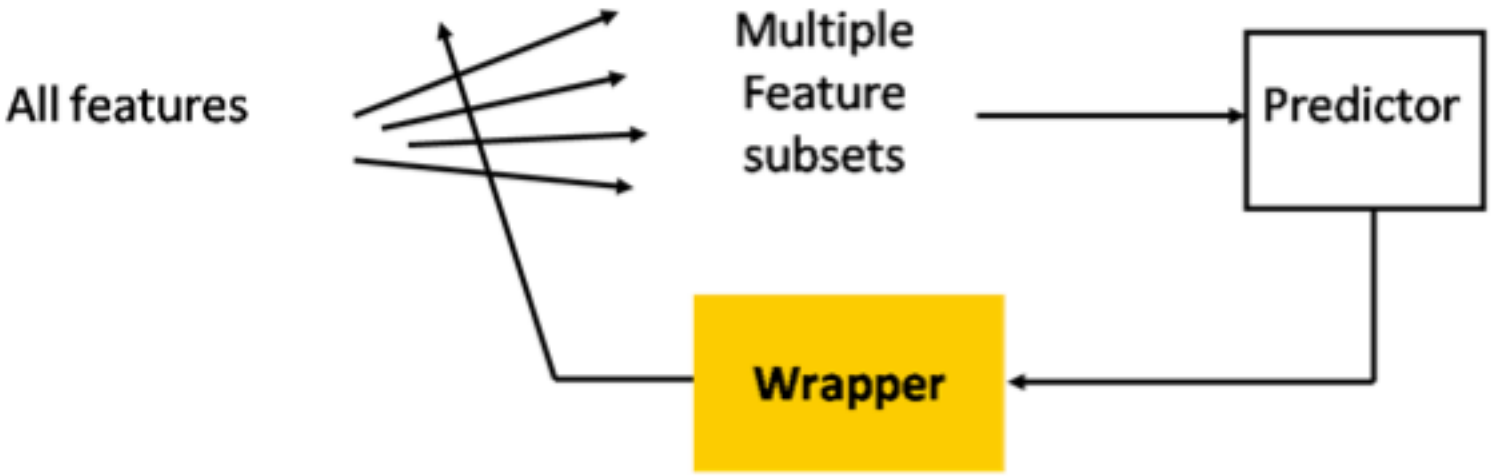
Guyon-Elisseeff, JMLR 2004; Springer 2006

(2) Wrapper

- Wrapper approach:

  uses a predictor to assess (many) features or feature subsets.

Wrapper Methods



(2) Wrapper : Feature Subset Selection

Wrapper Methods

- Learner is considered a black-box
- Interface of the black-box is used to score subsets of variables according to the predictive power of the learner when using the subsets.

- Results vary for different learners

(b). Search: even more search strategies for selecting feature subset

- **Forward selection** or **backward elimination.**
- **Beam search:** keep k best path at each step.
- **GSFS:** generalized sequential forward selection – when (n-k) features are left try all subsets of g features. More trainings at each step, but fewer steps.
- **PTA(l,r):** plus l , take away r – at each step, run SFS l times then SBS r times.
- **Floating search**: One step of SFS (resp. SBS), then SBS (resp. SFS) as long as we find better subsets than those of the same size obtained so far.

From Dr. **Isabelle Guyon**

(3) Embedded

- Embedding approach:

  uses a predictor to build a (single) model with a subset of features that are internally selected.

In practice…

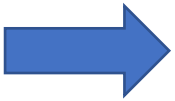
- No method is universally better:
  - wide variety of types of variables, data distributions, learning machines, and objectives.

- Feature selection is not always necessary to achieve good performance.

From Dr. **Isabelle Guyon**

Today: Dimensionality Reduction (Two Ways)

Feature extraction: finds a set of new features (i.e., through some mapping f()) from the existing features.



The mapping f() could be linear or non-linear

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ x_N \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_K \end{bmatrix}$$

K<<N

Feature selection: chooses a subset of the original features.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ x_N \end{bmatrix} \rightarrow \mathbf{y} = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ \cdot \\ \cdot \\ x_{i_K} \end{bmatrix}$$

K<<N

Feature Extraction

- Linear combinations are particularly attractive because they are simpler to compute and analytically tractable.

- Given $x \in R^N$, find an N x K matrix U such that:

$$y = U^T x \in R^K \text{ where } K<N$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ . \\ x_N \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix} \qquad U^T$$

This is a projection from the N-dimensional space to a K-dimensional space.

Feature Extraction (cont'd)

- From a mathematical point of view, finding an optimum mapping $\mathbf{y}=f(\mathbf{x})$ is equivalent to optimizing an **objective** function.

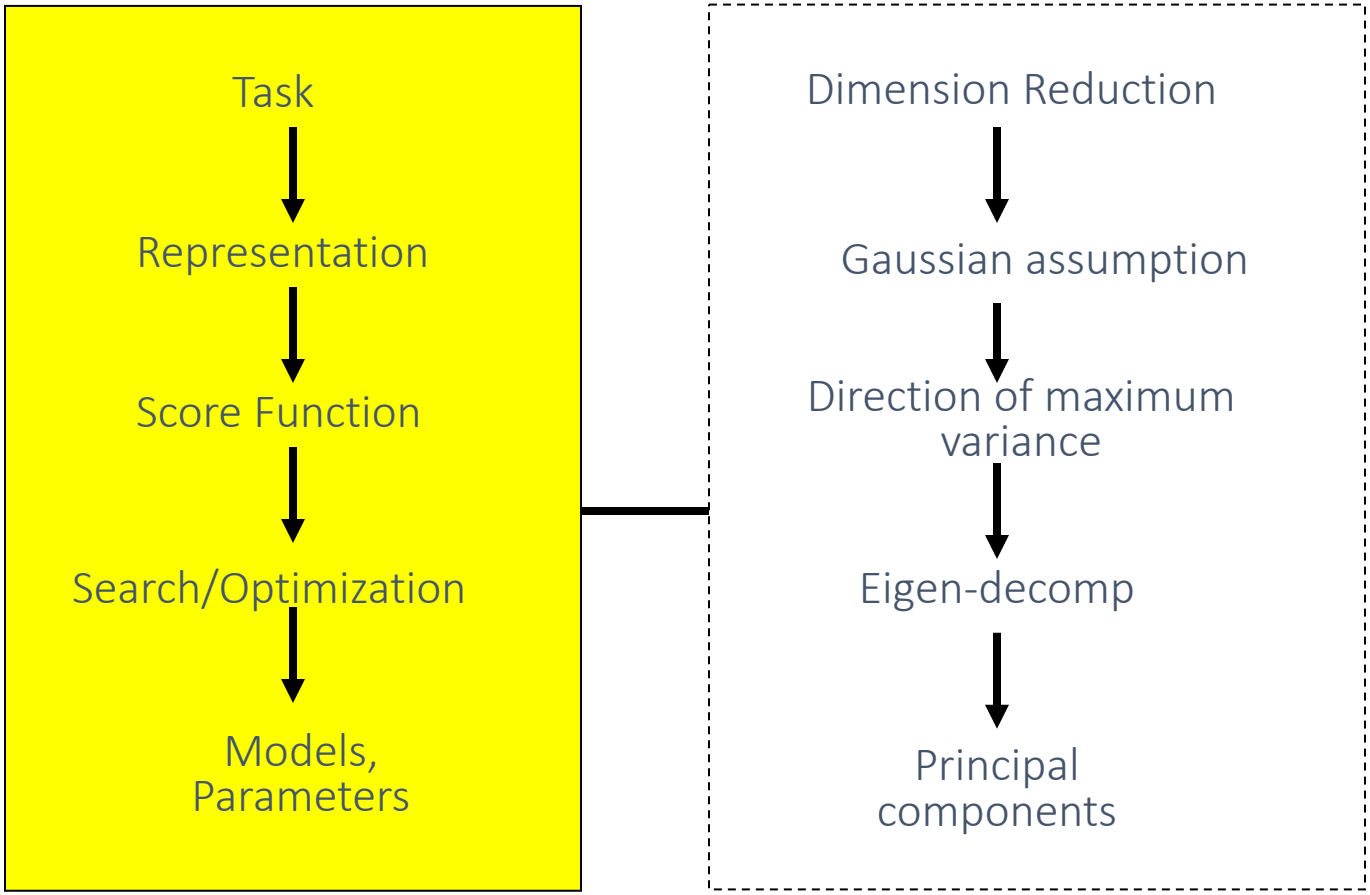
- Different methods use different objective functions, e.g.,
  - Information Loss: The goal is to represent the data as accurately as possible (i.e., no loss of information) in the lower-dimensional space.
  - Discriminatory Information: The goal is to enhance the class-discriminatory information in the lower-dimensional space.

Feature Extraction (cont'd)

- Commonly used linear feature extraction methods:
  - Principal Components Analysis (PCA): Seeks a projection that **preserves** as much **information** in the data as possible.
  - Linear Discriminant Analysis (LDA): Seeks a projection that **best discriminates** the data.

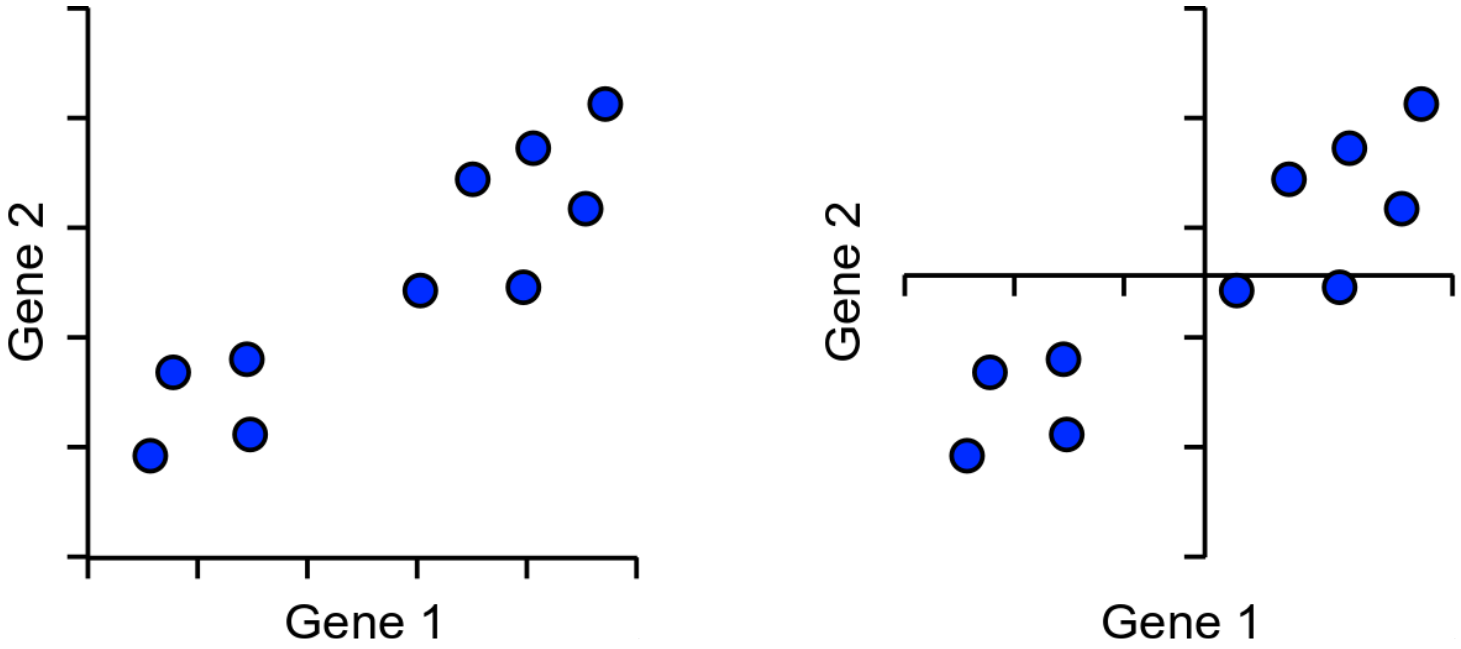
- More methods:
  - Retaining interesting directions (Projection Pursuit),
  - Making features as independent as possible (Independent Component Analysis or ICA),
  - Embedding to lower dimensional manifolds (Isomap, Locally Linear Embedding or LLE).

Principal Component Analysis



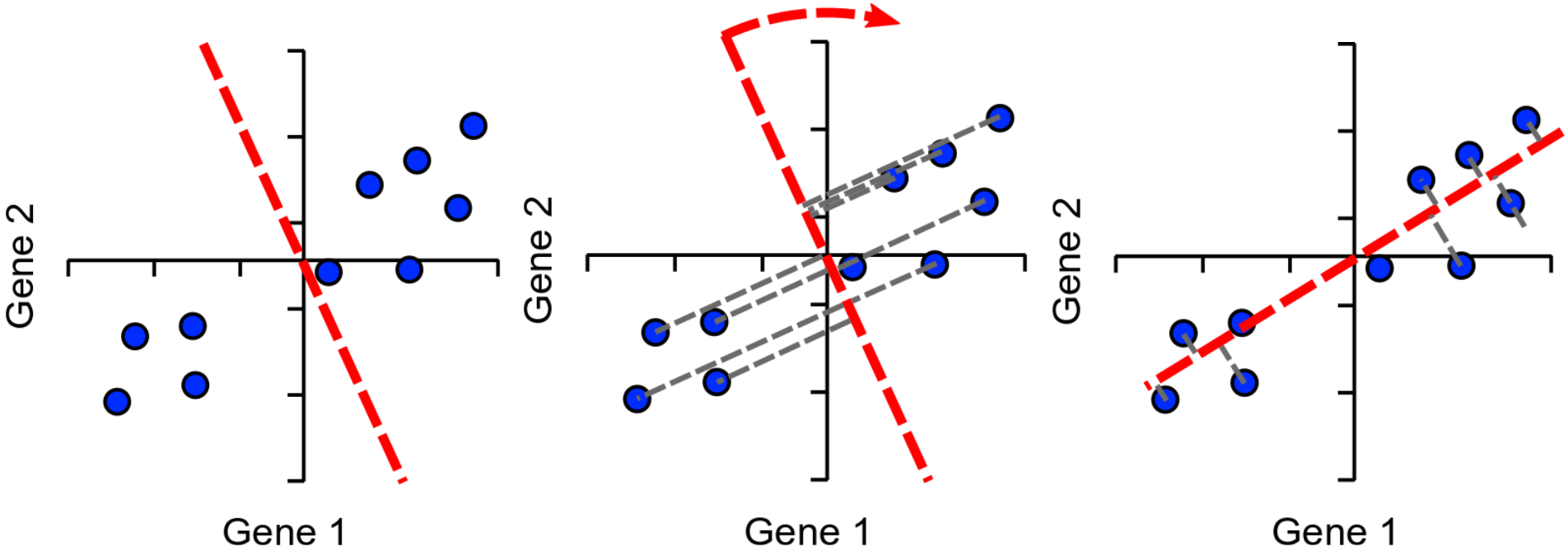
How does PCA work?

- Principal Components Analysis (PCA): approximating a high-dimensional data set with a lower-dimensional linear subspace



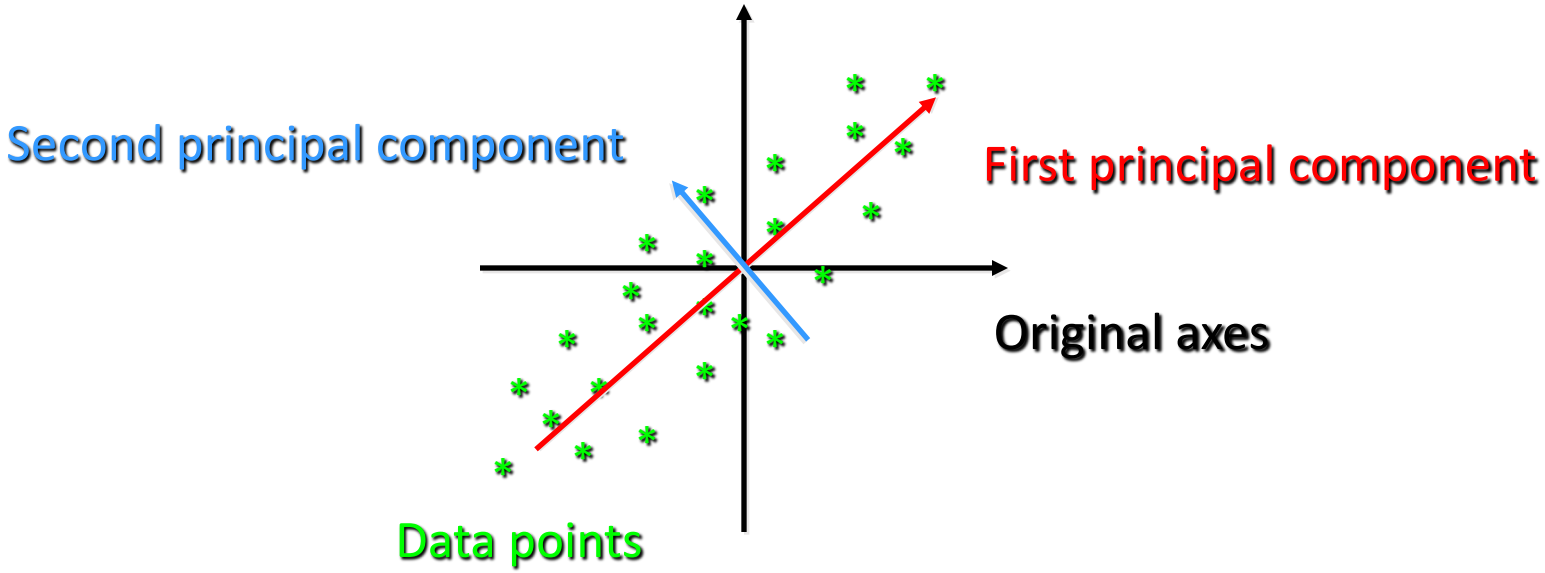
How does PCA work?

- Find line of best fit, passing through the origin

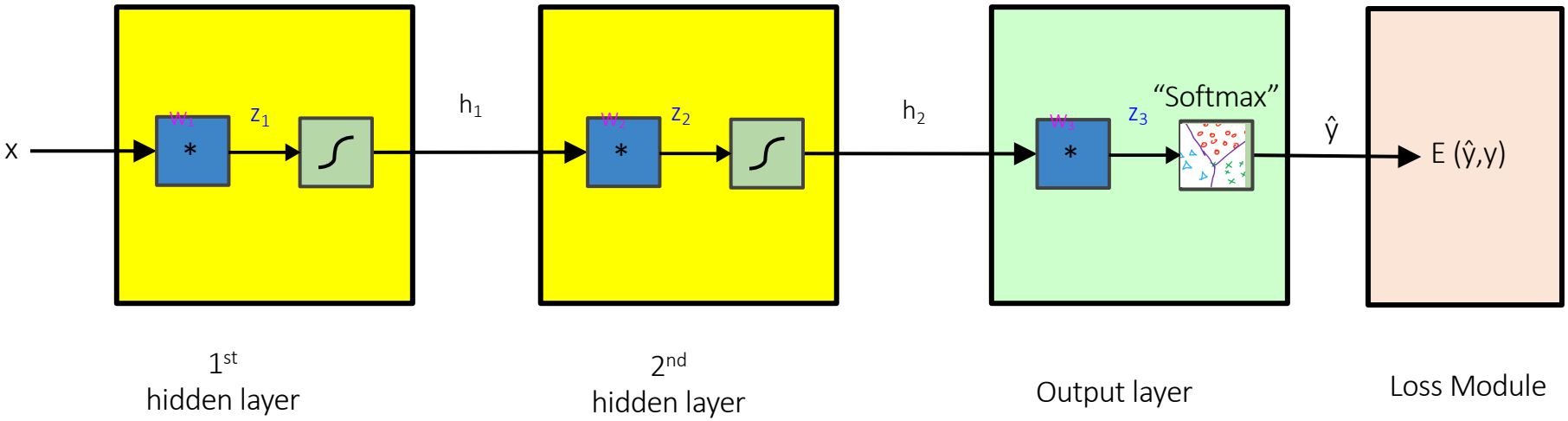


How does PCA work? Explaining Variance

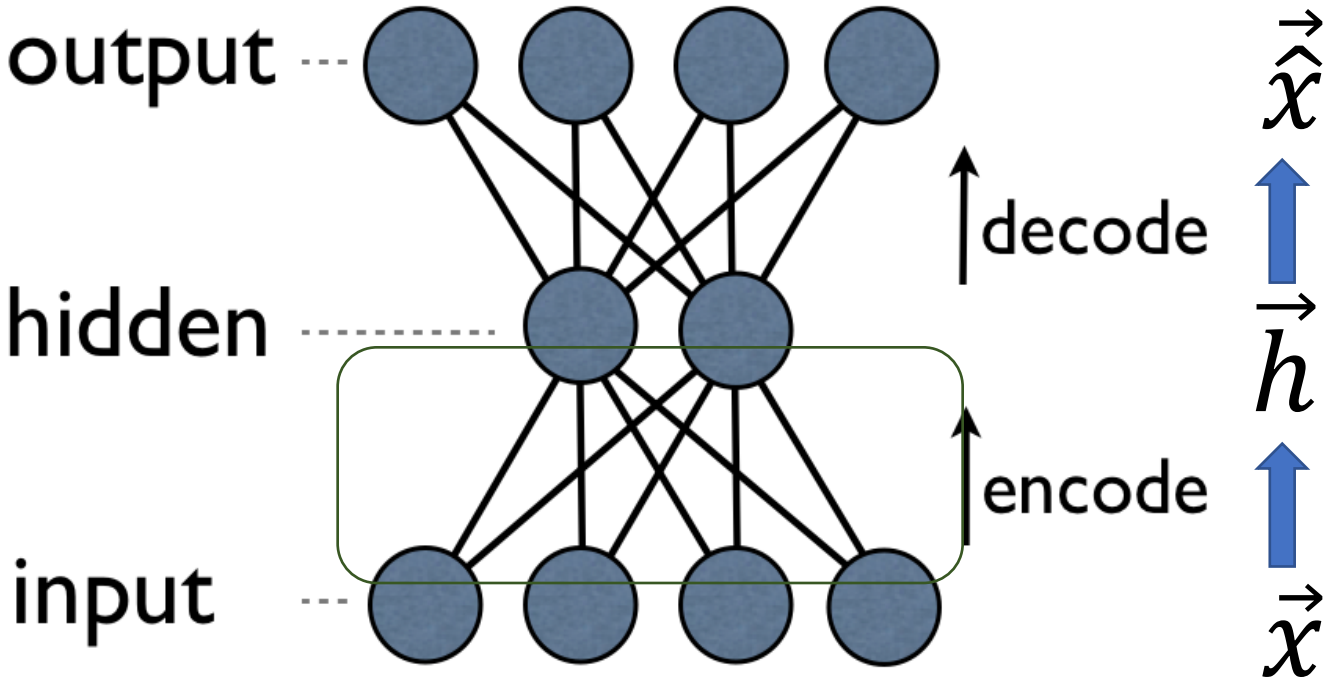
- Each PC always explains some proportion of the total variance in the data. Between them they explain everything
  - PC1 always explains the most
  - PC2 is the next highest etc. etc.



Recap: “Block View”



an auto-encoder-decoder is trained to reproduce the input



Minimize diff

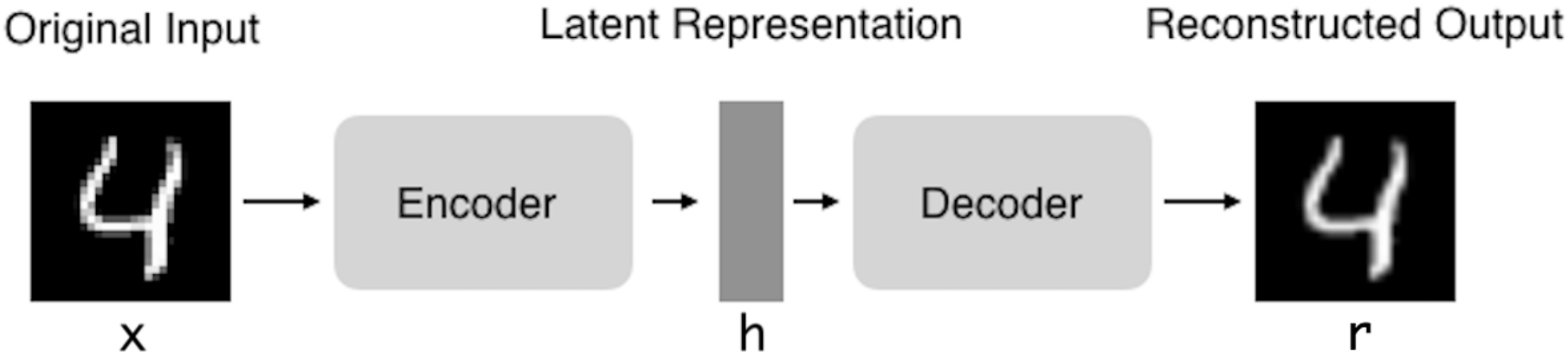
$$|\vec{\hat{x}} - \vec{x}|$$

Reconstruction Loss: force the 'hidden layer' units to become good / reliable feature detectors

https://www.macs.hw.ac.uk/~dwcorne/Teaching/introdl.ppt

Autoencoders: structure

- Encoder: compress input into a latent-space of usually smaller dimension. h = f(x)
- Decoder: reconstruct input from the latent space. r = g(f(x)) with r as close to x as possible

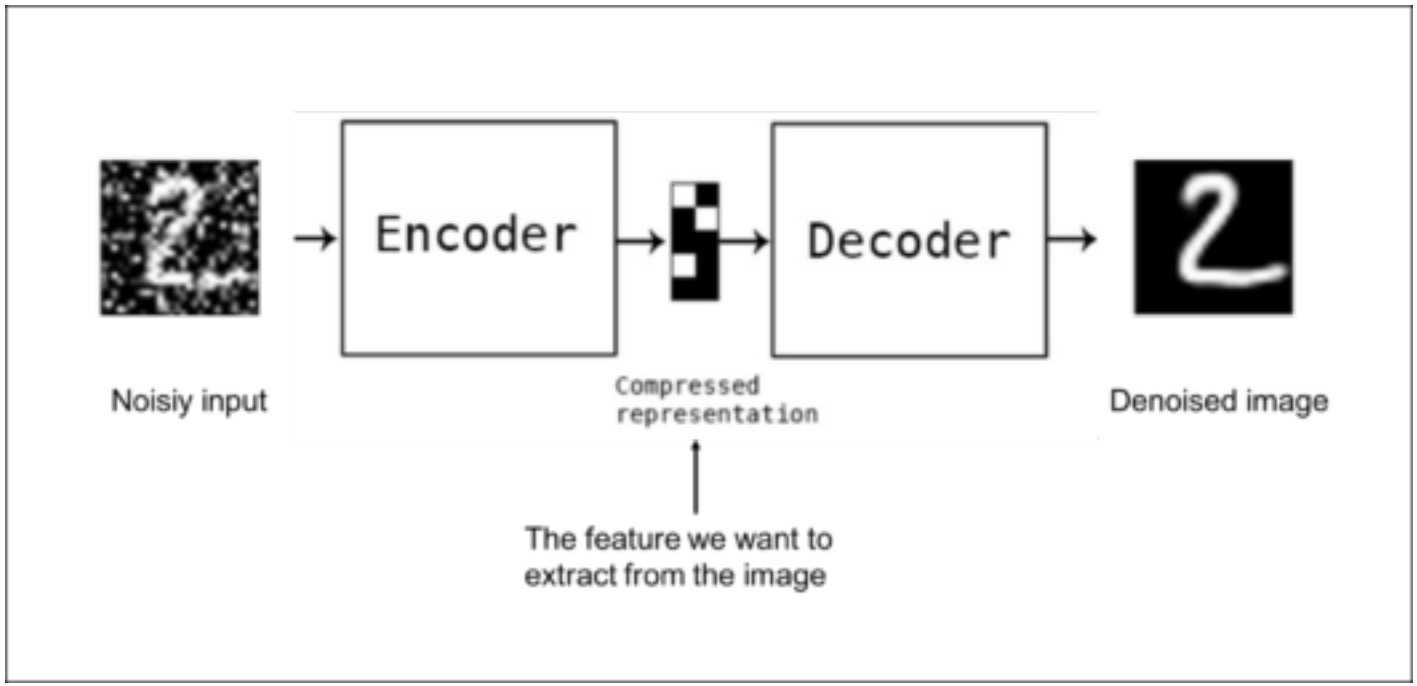


https://towardsdatascience.com/deep-inside-autoencoders-7e41f319999f

Autoencoders: many variations

- Denoising: input clean image + noise and train to reproduce the clean image.

- Neural network autoencoders
  - Can learn nonlinear dependencies
  - Can use convolutional layers
  - Can use transfer learning



https://www.edureka.co/blog/autoencoders-tutorial/

Today Recap: Dimensionality Reduction (Two Ways)

Feature extraction: finds a set of new features (i.e., through some mapping f()) from the existing features.

The mapping f() could be linear or non-linear

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ x_N \end{bmatrix} \xrightarrow{f(\mathbf{x})} \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_K \end{bmatrix}$$

K<<N

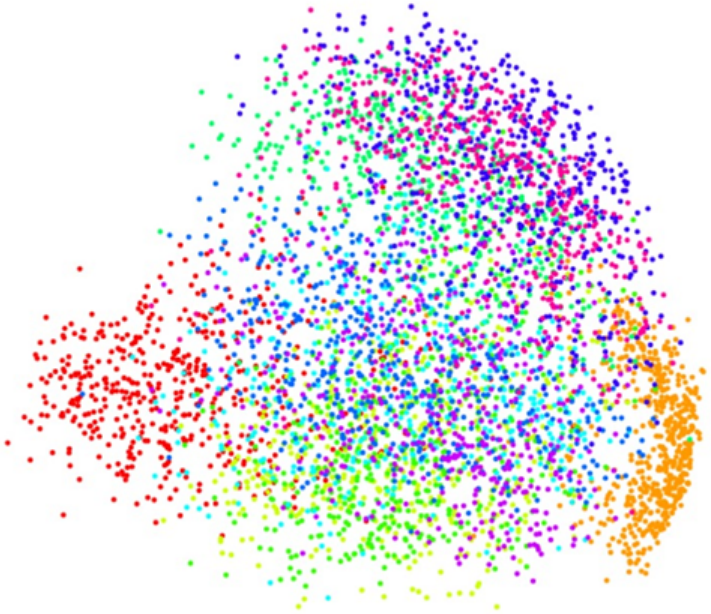
Feature selection: chooses a subset of the original features.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ . \\ x_N \end{bmatrix} \rightarrow \mathbf{y} = \begin{bmatrix} x_{i_1} \\ x_{i_2} \\ . \\ . \\ x_{i_K} \end{bmatrix}$$

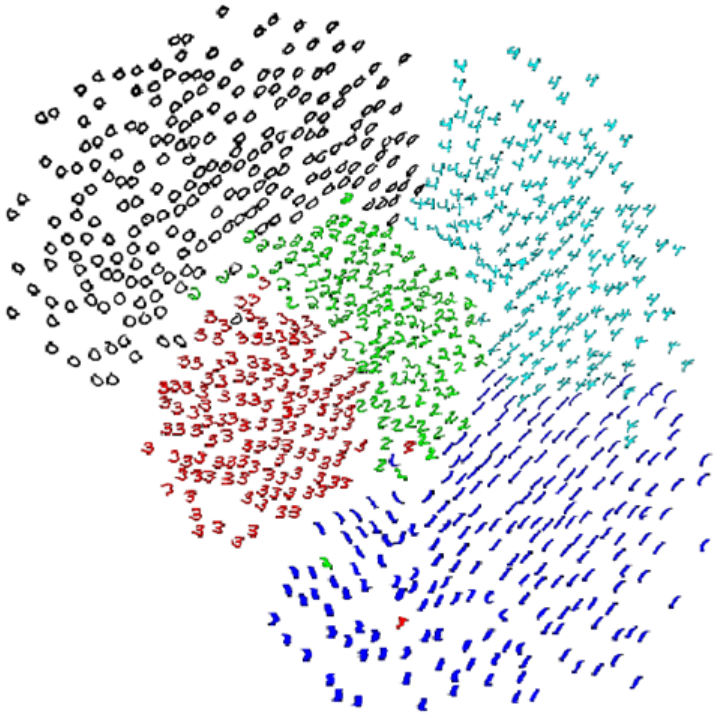
K<<N

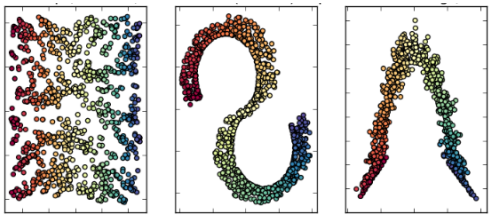
Many More: tSNE / UMPA

PCA on MNIST *(0-9)*



SNE on MNIST *(0-5)*





References

- Hastie, Trevor, et al. *The elements of statistical learning*. Vol. 2. No. 1. New York: Springer, 2009.
- Dr. S. Narasimhan's PCA lectures
- Prof. Derek Hoiem's eigenface lecture

Extra for PCA

Today

- Dimensionality Reduction (unsupervised) with Principal Components Analysis (PCA)
  - Review of eigenvalue, eigenvector
  - How to project samples into a line capturing the variation of the whole dataset ➔ Eigenvector / Eigenvalue of covariance matrix
  - PCA for dimension reduction
  - Eigenface ➔ PCA for face recognition

Review: Mean and Variance

- Variance:

$$Var(X) = E((X-\mu)^2)$$

  - Discrete RVs:

$$V(\mathrm{X}) = \sum_{v_i} (v_i - \mu)^2 \, \mathrm{P}(\mathrm{X} = v_i)$$

  - Continuous RVs:

$$V(\mathrm{X}) = \int_{-\infty}^{+\infty} (x-\mu)^2 \, f(x) dx$$

- Covariance:

$$Cov(X,Y) = E((X-\mu_x)(Y-\mu_y)) = E(XY) - \mu_x \mu_y$$

Review: Eigenvector / Eigenvalue

- The eigenvalues λ_i are found by solving the equation

$$det(C-\lambda I)=0$$

$$Cu=\lambda u \quad u\neq 0$$

- Eigenvectors are columns of the matrix U such that

$$C= U D U^{\top}$$

- Where

$$D= \begin{pmatrix} \lambda_1 0 \ldots\ldots 0 \\ 0 \quad \lambda_2 \ldots\ldots 0 \\ 0 \\ 0 \ldots\ldots\ldots\ldots \lambda_p \end{pmatrix}$$

From Dr. S. Narasimhan

Review: Eigenvalue, e.g.

- Let us take two variables with covariance $c>0$
- $\mathbf{C}=\begin{pmatrix}1 & c\\ c & 1\end{pmatrix}$ $\mathbf{C}-\lambda\mathbf{I}=\begin{pmatrix}1-\lambda & c\\ c & 1-\lambda\end{pmatrix}$

$$\det(\mathbf{C}-\lambda\mathbf{I})=(1-\lambda)^2-c^2$$

$\mathbf{C}\mathbf{u}=\lambda\mathbf{u}$, $\mathbf{u}\neq 0$

- Solving this we find $\lambda_1 = 1+c$

$$\lambda_2 = 1-c < \lambda_1$$

From Dr. S. Narasimhan

Review: Eigenvector, e.g.

- Any eigenvector U satisfies the condition

$$\mathbf{C}u = \lambda u$$

$$u = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \quad Cu = \begin{pmatrix} 1 & c \\ c & 1 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} a_1 + ca_2 \\ ca_1 + a_2 \end{pmatrix} = \begin{pmatrix} \lambda a_1 \\ \lambda a_2 \end{pmatrix}$$

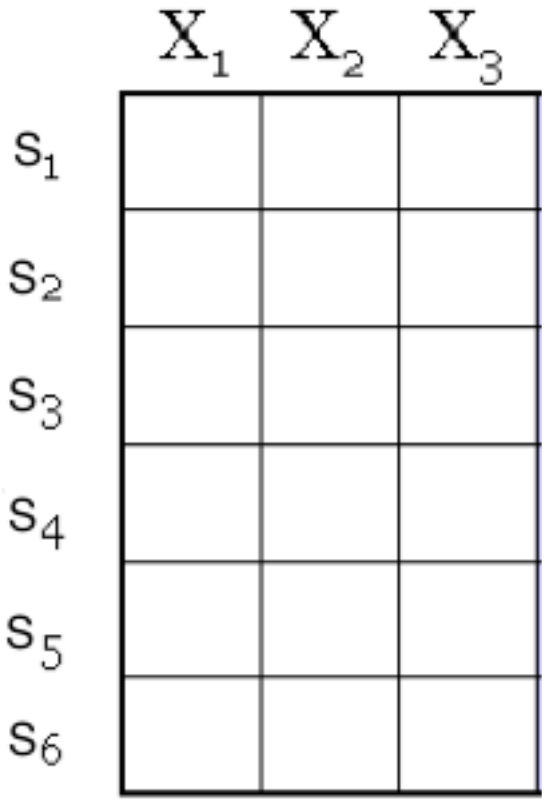
Solving we find $u_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$. $u_2 = \begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}$

In practice, much more advance methods, e.g. power method

From Dr. S. Narasimhan

Today

- Dimensionality Reduction (unsupervised) with Principal Components Analysis (PCA)
  - Review of eigenvalue, eigenvector
  - How to project samples into a line capturing the variation of the whole dataset ➔ Eigenvector / Eigenvalue of covariance matrix
  - PCA for dimension reduction
  - Eigenface ➔ PCA for face recognition



d

a data matrix of n observations on p variables $x_1, x_2, \ldots x_p$

- **Data***/points/instances/examples/samples/records*: [rows]
- **Features***/attributes/dimensions/independent variables/covariates/predictors/regressors*: [columns]

The 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.

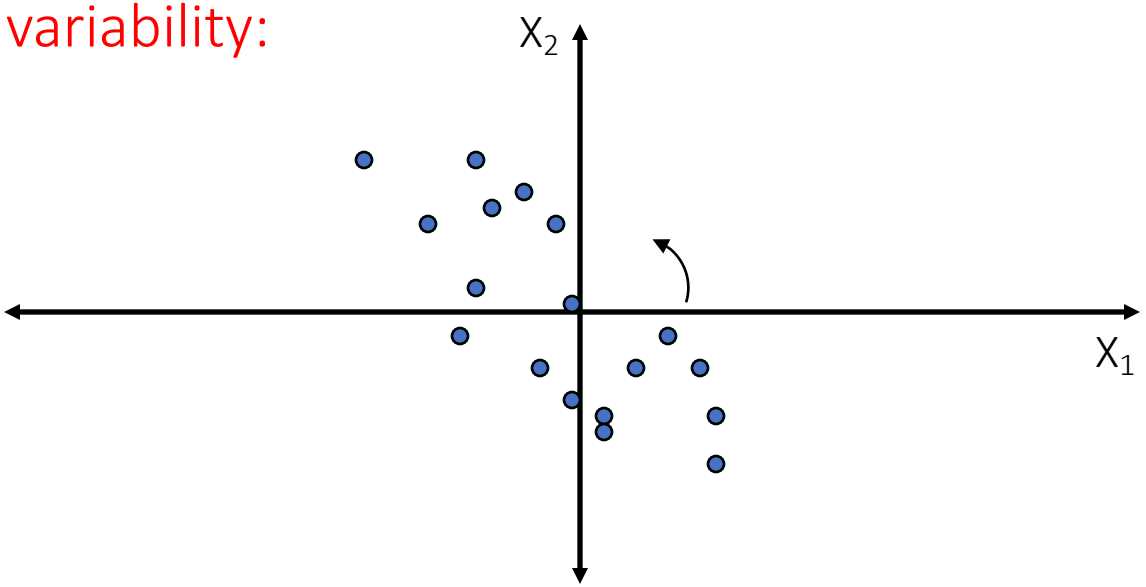
PCA is introduced by Pearson (1901) and Hotelling (1933)

Trick: Rotate Coordinate Axes

Suppose we have a sample population measured on p random variables $X_1,\ldots,X_p$.

Our goal is to develop a new set of K (K<p) axes

(linear combinations of the original p axes) in the directions of greatest variability:



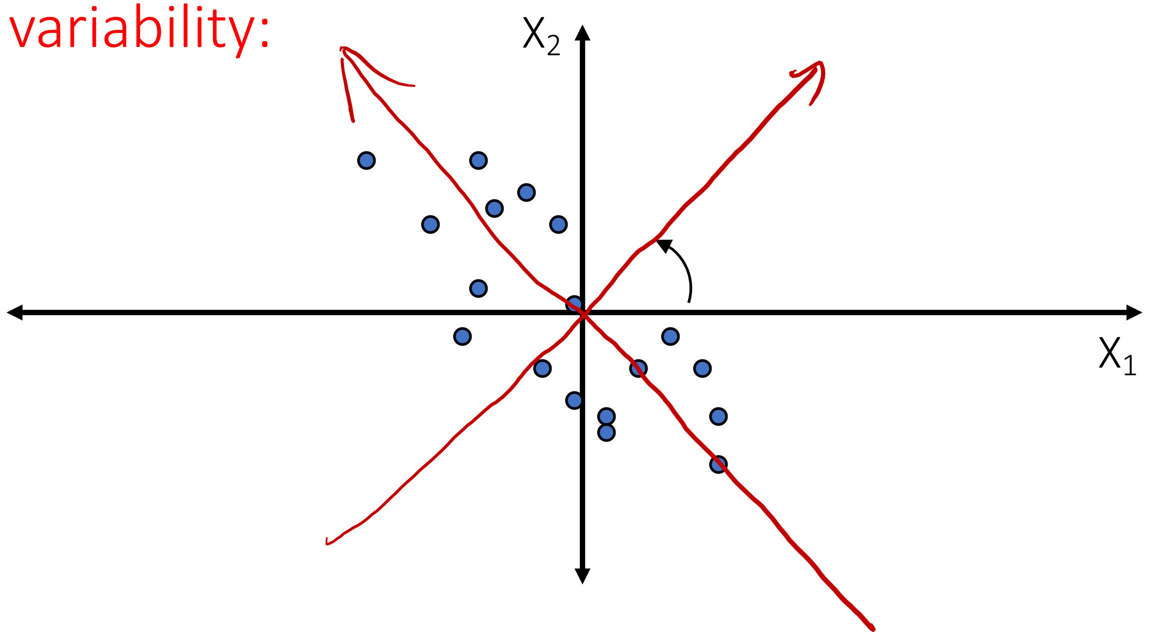
This could be accomplished by rotating the axes (if data is centered).

Trick: Rotate Coordinate Axes

Suppose we have a sample population measured on p random variables $X_1,\ldots,X_p$.

Our goal is to develop a new set of K (K<p) axes

(linear combinations of the original p axes) in the directions of greatest variability:



This could be accomplished by rotating the axes (if data is centered).

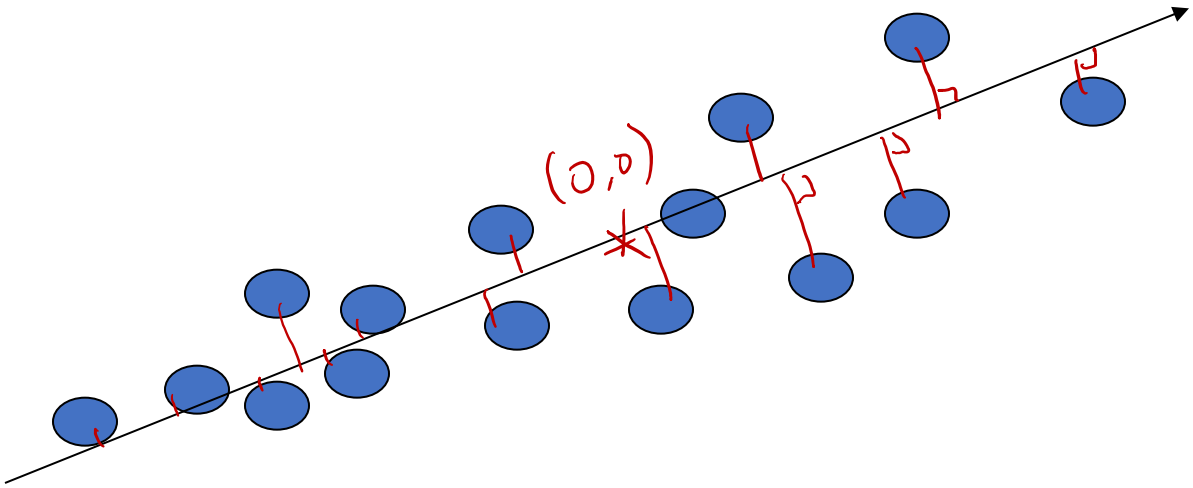
Algebraic Interpretation

- Given n points in a p dimensional space,
- for large p, how to project on to a lower-dimensional (K<p) space while preserving broad trends in the data and allowing it to be visualized?

FROM NOW we assume Data matrix is centered: ➔ (we subtract the mean along each dimension, and center the original axis system at the centroid of all data points, for simplicity)

Algebraic Interpretation – (k=1)

- Given n points in a p dimensional space, how to project on to a 1 dimensional space?

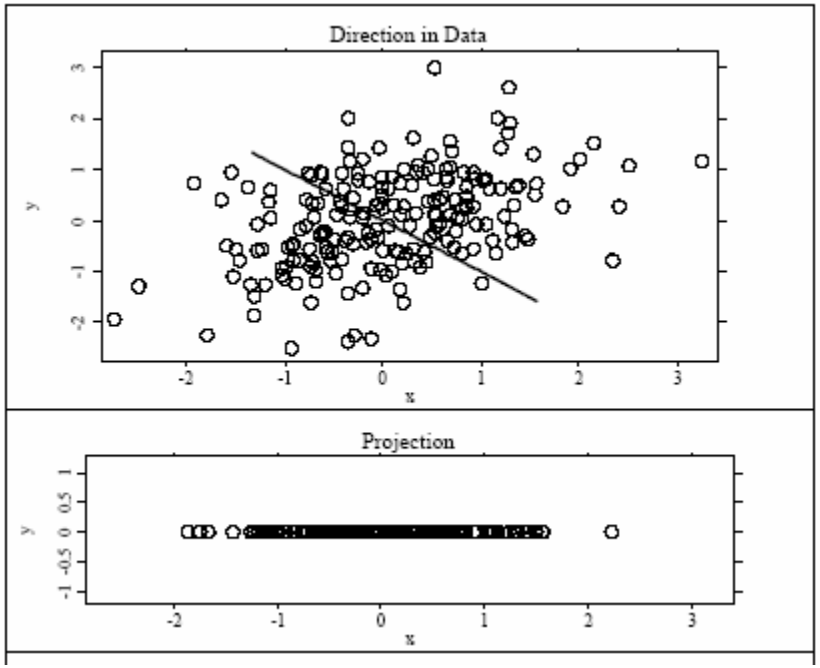


- Choose a line that fits the data so the points are spread out well along the line

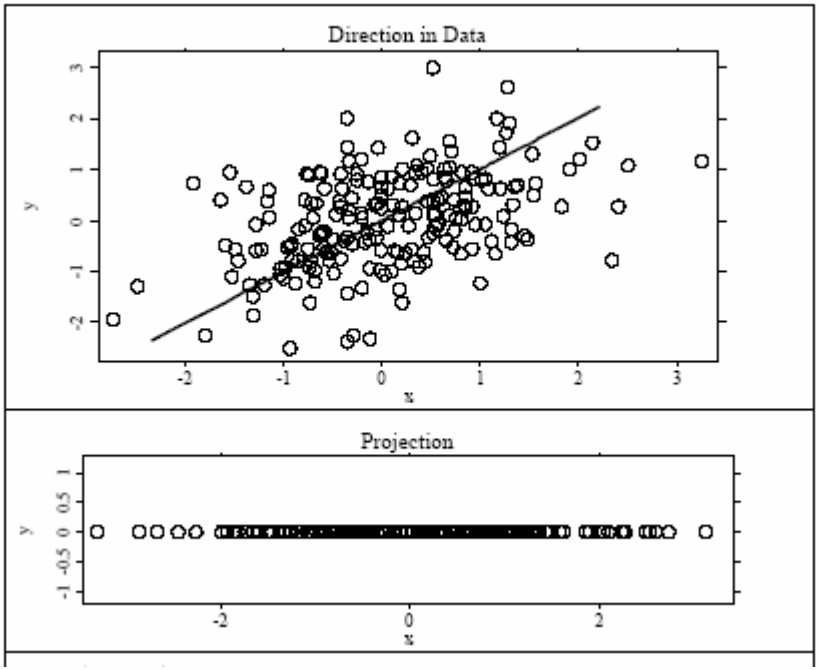
From Dr. S. Narasimhan

Let us see it on a figure

Good

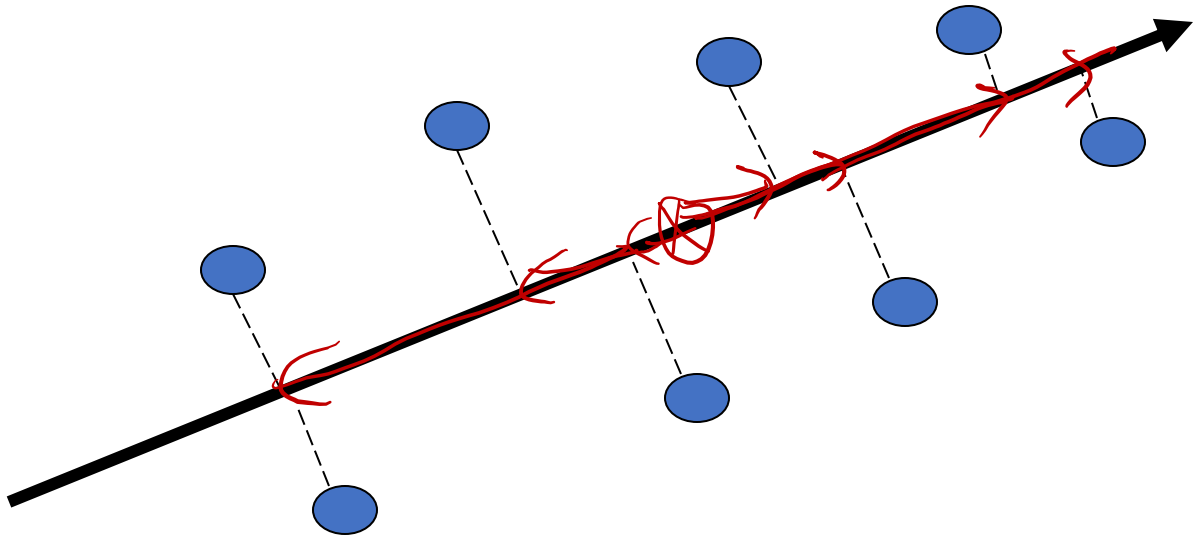


Better



Algebraic Interpretation – (k=1)

- Formally, to find a line that ➔ Maximizing the sum of squares of data samples' projections on that line



From Dr. S. Narasimhan

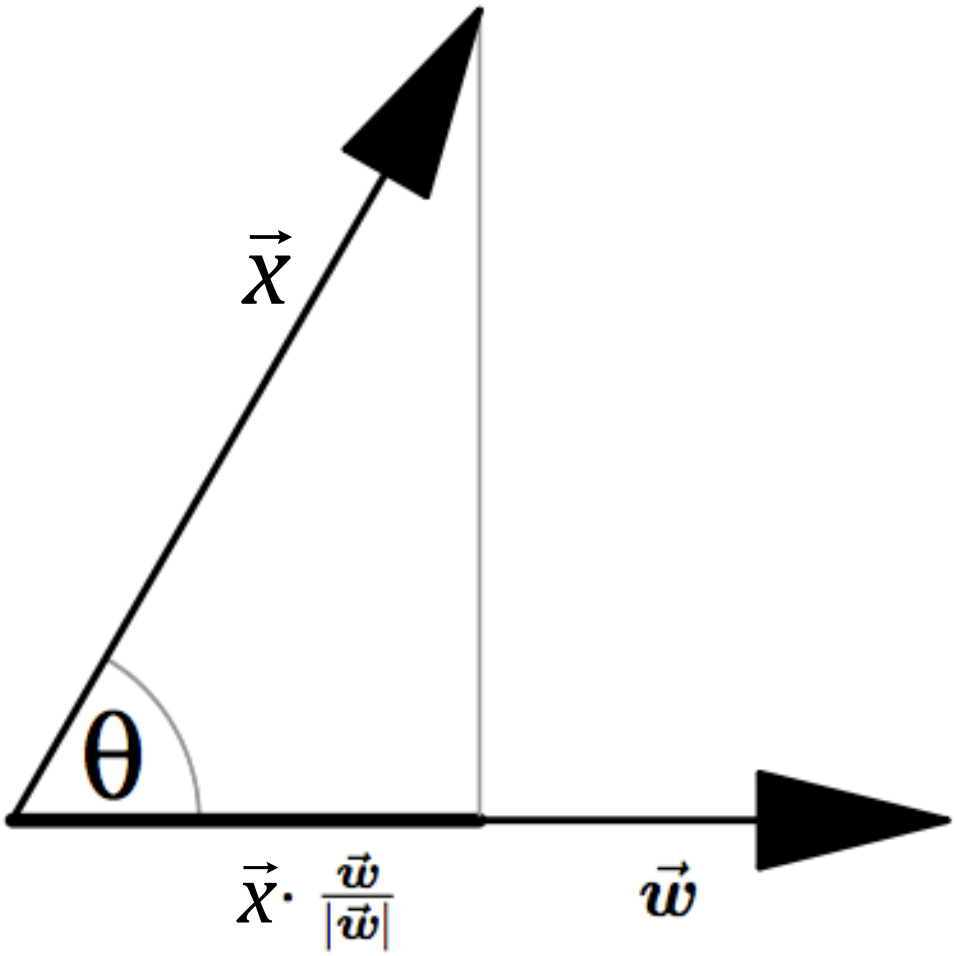
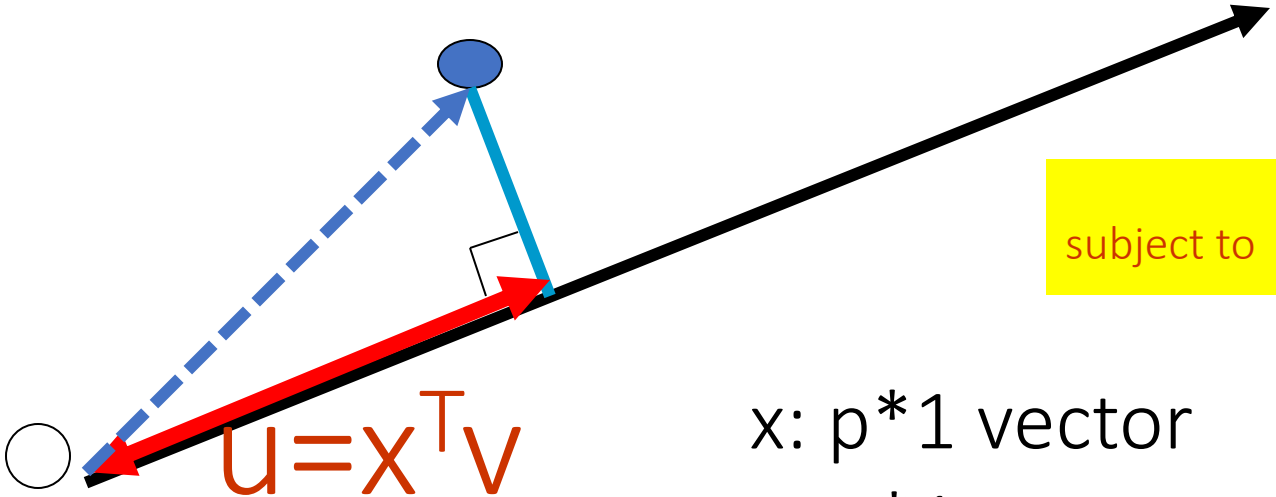


Figure 1: The dot product is fundamentally a projection.

Algebraic Interpretation – 1D

- Formally, to find a line (direction) that ➔ Maximizing the sum of squares of data samples' projections on that line



subject to $v^Tv = 1$

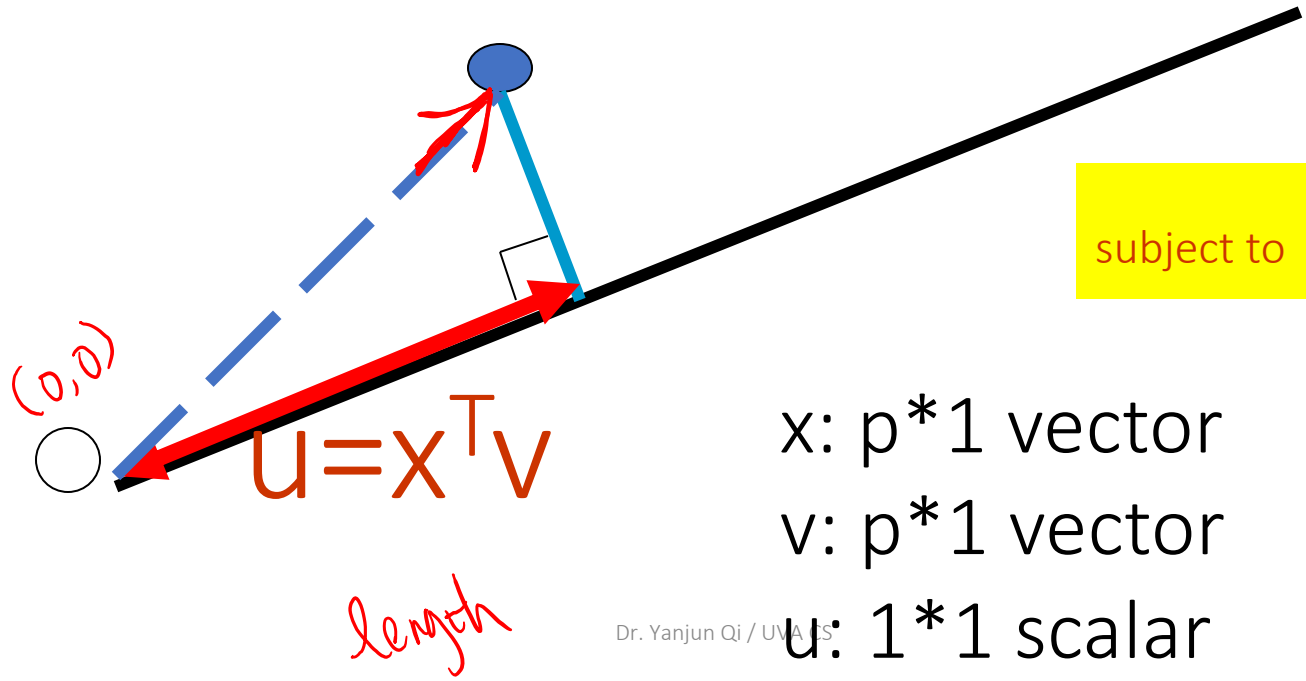
x: p*1 vector

v: p*1 vector

u: 1*1 scalar

Algebraic Interpretation – 1D

- Formally, to find a line (direction) that ➔ Maximizing the sum of squares of data samples' projections on that line



subject to $v^Tv = 1$

x: p*1 vector

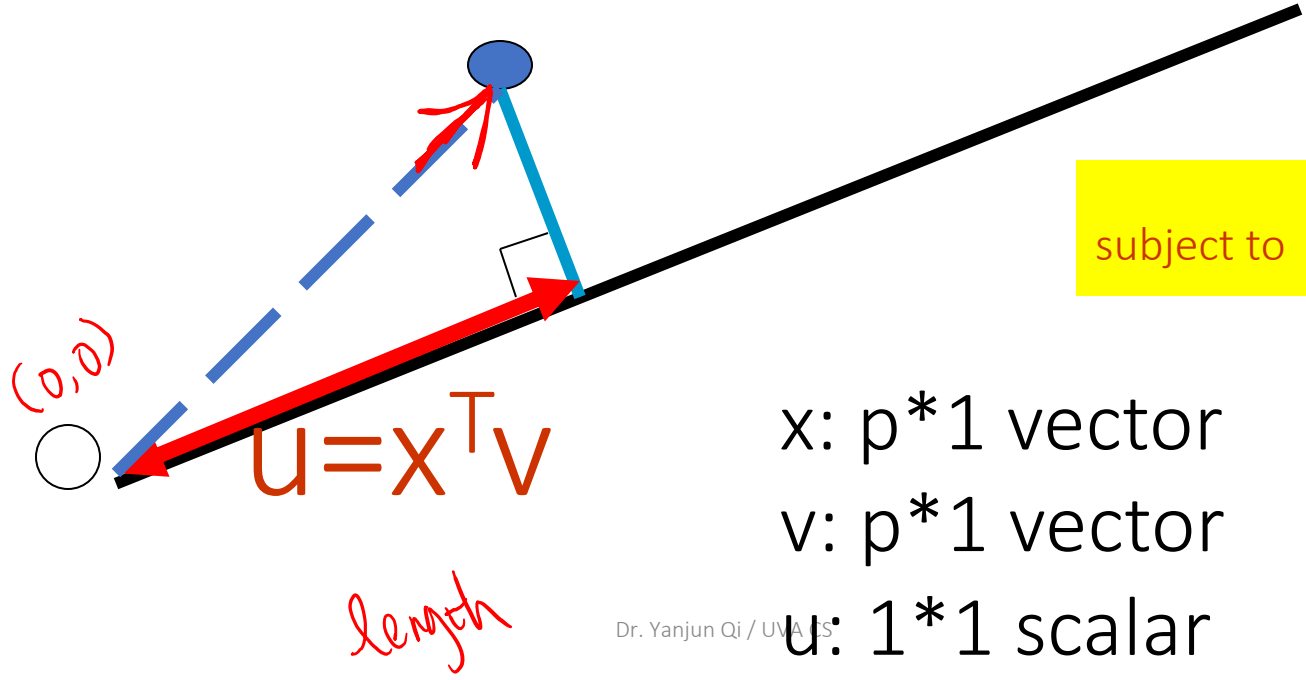
v: p*1 vector

u: 1*1 scalar

Algebraic Interpretation – 1D

- Formally, to find a line (direction) that ➔ Maximizing the sum of squares of data samples' projections on that line

size of x's projection on vector v ➔ $u=x^Tv = v^Tx$



subject to $v^Tv = 1$

x: p*1 vector

v: p*1 vector

u: 1*1 scalar

Algebraic Interpretation – 1D case

$$\arg\max_{v} \sum_{u_i} (u_i)^2$$

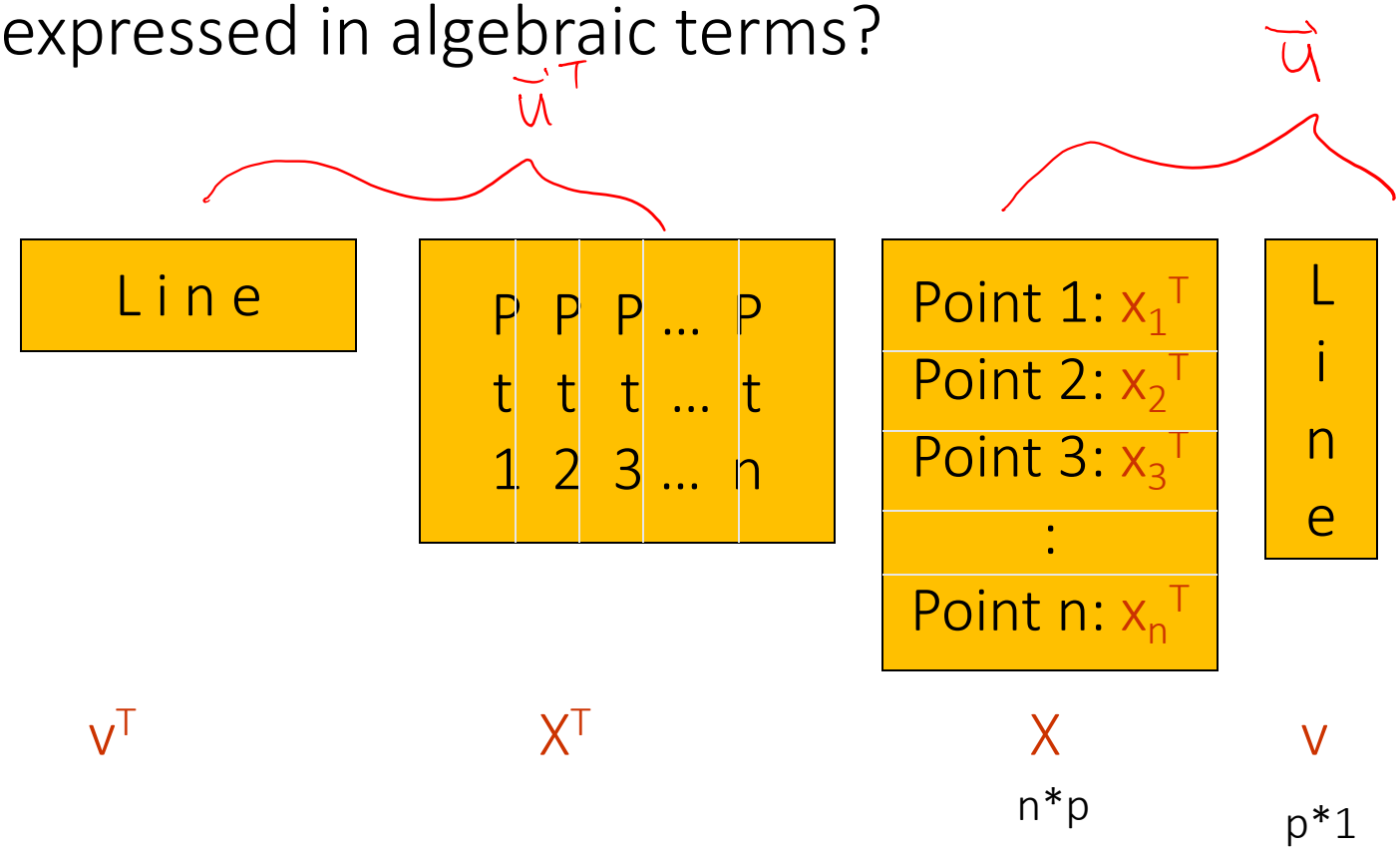
$$\max \left\{ \sum_{i=1}^{n} u_i^2 \right\} =$$

$$= [u_1, u_2, u_3, \ldots, u_n] \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} = \vec{u}^T \vec{u} \qquad \vec{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix}$$

Here $\vec{u}$ = vector $= \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} x_1^T v \\ x_2^T v \\ \vdots \\ x_n^T v \end{bmatrix} = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{bmatrix} v = \underset{n \times p}{X} \; \underset{p \times 1}{v}$

Algebraic Interpretation – 1D

- How is the sum of squares of projection lengths expressed in algebraic terms?



From Dr. S. Narasimhan

Algebraic Interpretation – 1D

- How is the sum of squares of projection lengths expressed in algebraic terms?

$$\max(\ v^T X^T X v),\ \text{subject to}\ \ v^T v = 1$$

Algebraic Interpretation – 1D

- Rewriting this:

$\max(v^T X^T Xv)$, subject to $v^T v = 1$

$$v^T X^T X v = \lambda = \lambda\, v^T v = v^T (\lambda v)$$

$$<=> \quad v^T (X^T X v - \lambda v) = 0$$

factor λ.

Algebraic Interpretation – 1D

- Rewriting this: $\max(v^T X^T Xv)$, subject to $v^T v = 1$

$$v^T X^T X v = \lambda = \lambda\, v^T v = v^T (\lambda v)$$

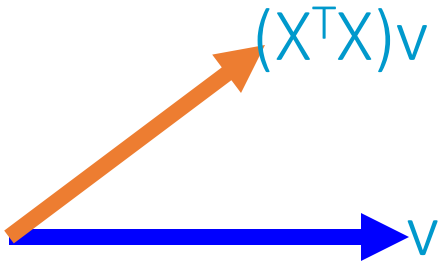
$$<=> \quad v^T (X^T X v - \lambda v) = 0$$

- Show that the maximum value of $v^T X^T X v$ is obtained for those vectors / directions satisfying $X^T X v = \lambda v$

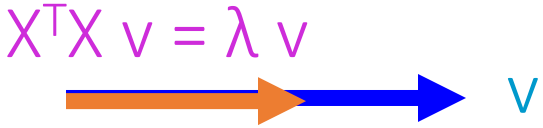
- So, find the largest λ and associated v such that the matrix $X^T X$ when applied to v, yields a new vector which is in the same direction as v, only scaled by a factor λ.

Algebraic Interpretation – 1D

- $(X^TX)v$ points in some other direction (different from v) in general



➔ *If* v is an eigenvector and λ is corresponding eigenvalue



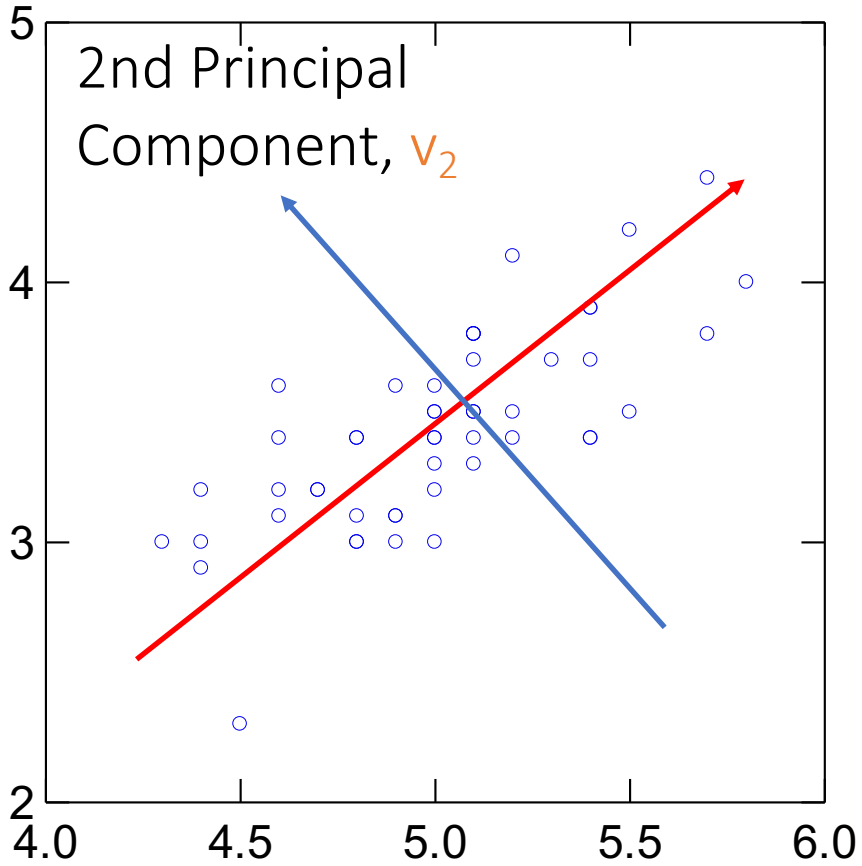
So, find the largest λ and associated v such that the matrix X^TX when applied to v, yields a new vector which is in the same direction as v, only scaled by a factor λ.

Algebraic Interpretation – beyond 1D

- For matrices of the form (symmetric) $X^T X$
  - All eigenvalues are non-negative
  - See Handout-1 "linear algebra review" / Page 18,19,20
  - $\lambda_1 \ldots \lambda_p$ *are the eigenvalues, ordering from large to small,*
    - *i.e. Ordered by the PC's importance*

PCA Eigenvectors ➔ Principal Components

$\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \cdots \geq \lambda_p$ [PCA]



PCA (k=1) : How the sum of squares of projection lengths relates to VARIANCE ?

size of one sample x's projection on vector v

➔ $u=x^Tv = v^Tx$

- In a new coordinate system with v as axis, u is the position of sample x on this axis

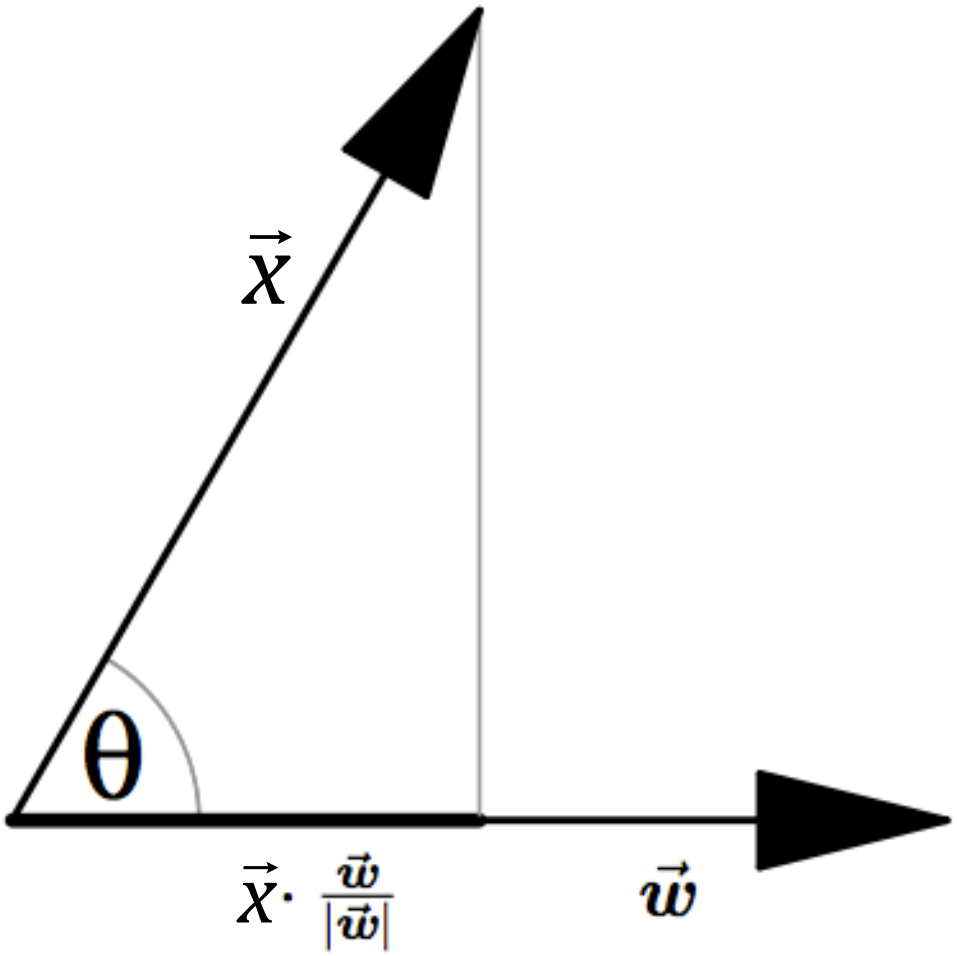
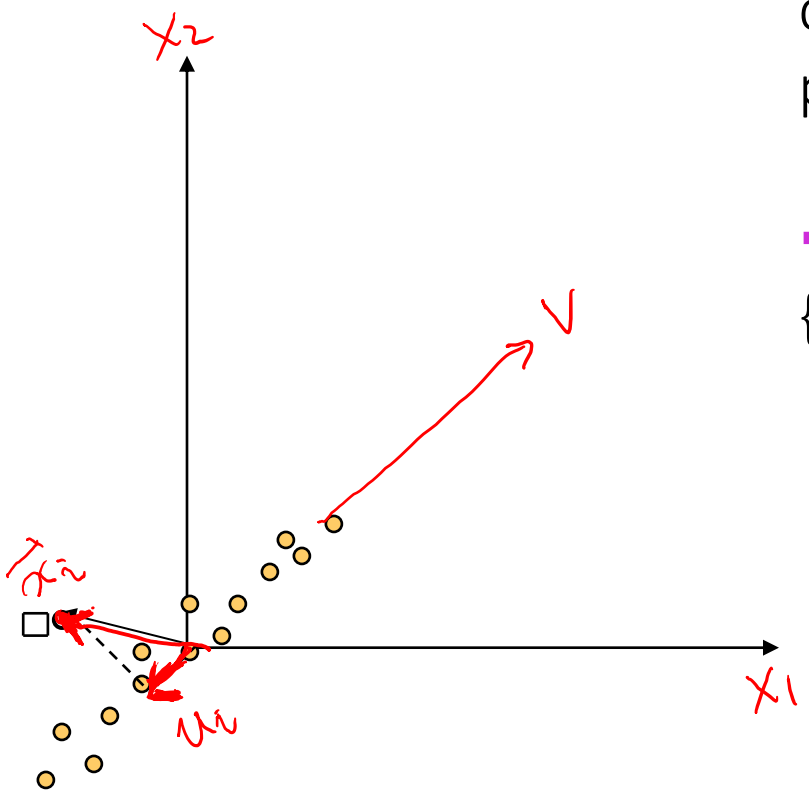


Figure 1: The dot product is fundamentally a projection.

PCA (k=1) : How the sum of squares of projection lengths relates to VARIANCE ?



Consider the variation along direction v considering all of the points {x_1, x_2,…, x_n}:

➔ The variance of all positions {u_1, u_2,…, u_n}

convert x_i onto V coordinate

➔ $u_i = (x_i)^T V$

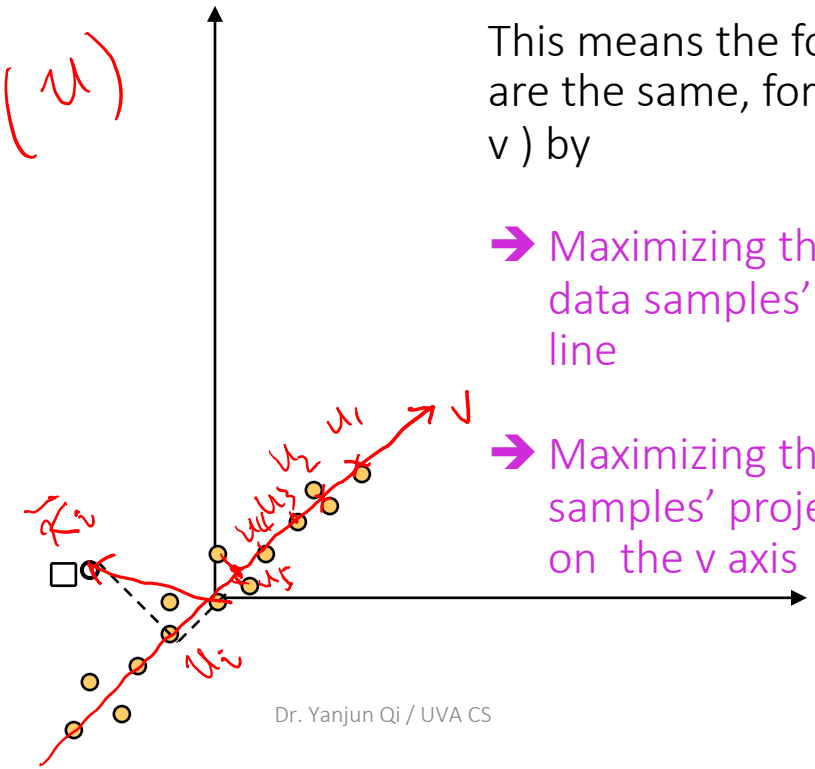
From Dr. S. Narasimhan

How the sum of squares of projection lengths relates to VARIANCE ?

$$Var(u) = \sum_{u_i} (u_i - \mu)^2 \, \mathrm{P}(u = u_i) = \sum_{u_i} (u_i)^2$$

Assuming centered data matrix

$$\text{argmax}_v \sum u_i^2 = \text{argmax}_v \, Variance(u)$$

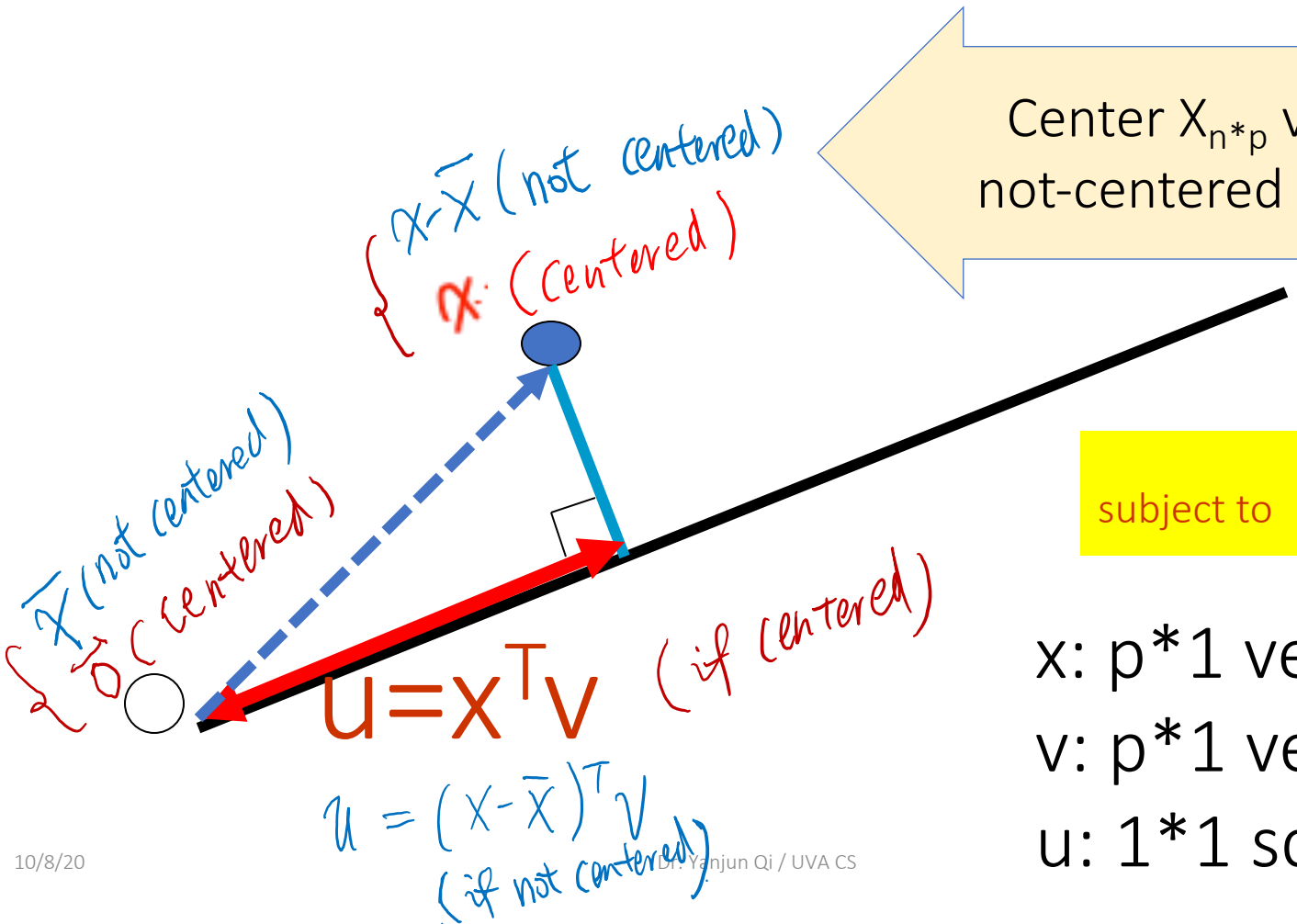


This means the following two objectives are the same, for finding a line (direction v) by

➔ Maximizing the sum of squares of data samples' projections on that v line

➔ Maximizing the variance of data samples' projected representations on the v axis

Centered Vs. Not Centered Formulation



subject to $v^T v = 1$

x: p*1 vector

v: p*1 vector

u: 1*1 scalar

Today

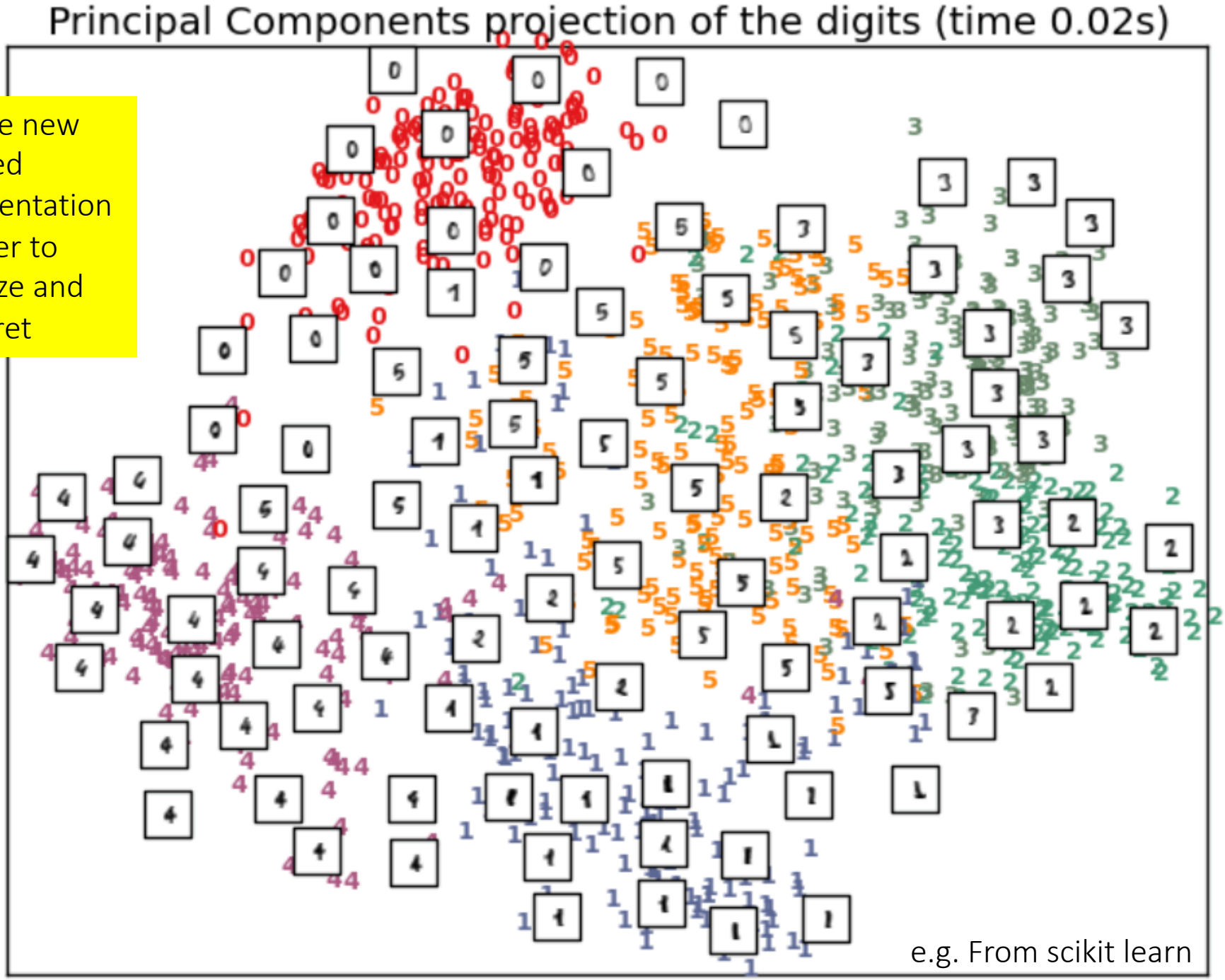
- Dimensionality Reduction (unsupervised) with Principal Components Analysis (PCA)
  - Review of eigenvalue, eigenvector
  - How to project samples into a line capturing the variation of the whole dataset ➔ Eigenvector / Eigenvalue of covariance matrix
  - PCA for dimension reduction
  - Eigenface ➔ PCA for face recognition

Applications

- Uses:
  - Data Visualization
  - Data Reduction
  - Data Classification
  - Trend Analysis
  - Factor Analysis
  - Noise Reduction

- Examples:
  - How many unique “sub-sets” are in the sample?
  - How are they similar / different?
  - What are the underlying factors that influence the samples?
  - How to best present what is “interesting”?
  - Which “sub-set” does this new sample rightfully belong?
  - …….

From Dr. S. Narasimhan


Principal Components projection of the digits (time 0.02s)
e.g. the new reduced representation is easier to visualize and interpret
e.g. From scikit learn

Interpretation of PCA

From p original coordinates: $X_1, X_2, \ldots, X_p$:

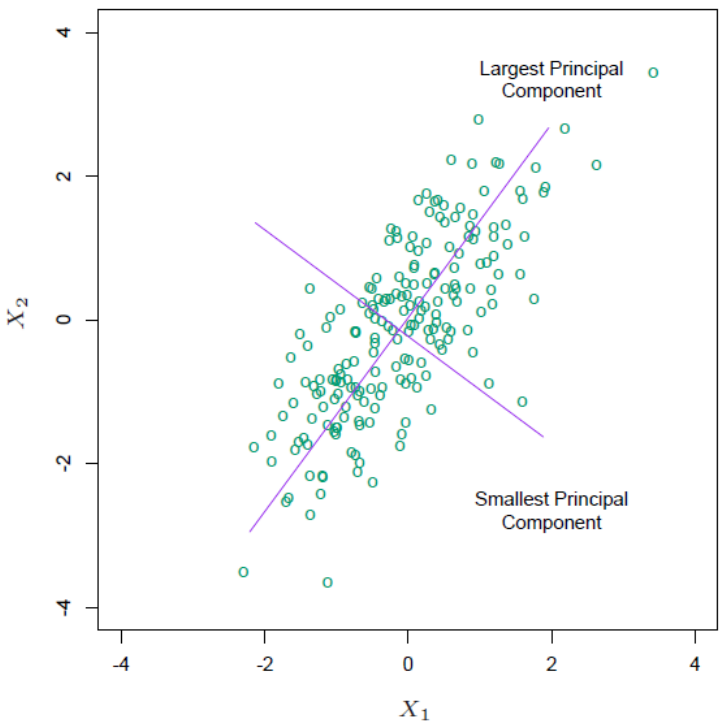
Produce k new coordinates : $V_1, V_2, \ldots, V_k$:

$$V_1 = a_{11}X_1 + a_{12}X_2 + \ldots + a_{1p}X_p$$

$$V_2 = a_{21}X_1 + a_{22}X_2 + \ldots + a_{2p}X_p$$

…

When p=2



Interpretation of PCA

v_k's are Principal Components

such that:

v_k are uncorrelated (orthogonal) from each other
v_1 explains as much as possible of original variance in data set
v_2 explains as much as possible of remaining variance
etc.

v_k : k^{th} PC retains the k^{th} greatest fraction of the variation in the samples

From Dr. S. Narasimhan

$$Var\left(u^k\right) = \sum_{i=1}^{n}\left(u^k{}_i\right)^2 = v_k{}^T X^T X v_k$$

- The new variables (PCs) have a variance equal to their corresponding eigenvalue, since

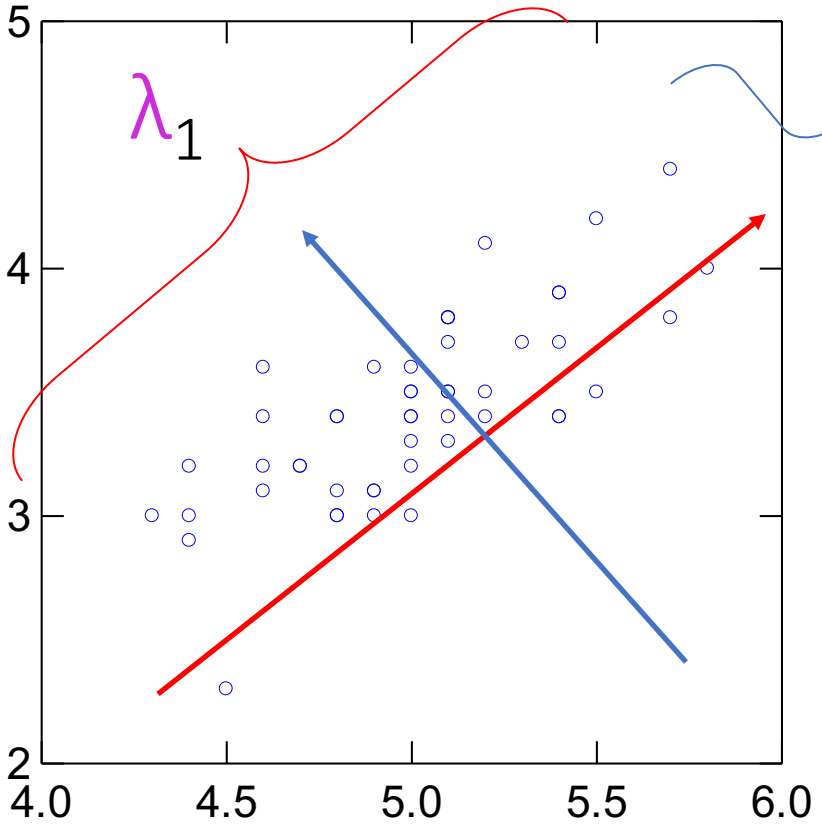
$$Var(u^k) = v_k{}^T X^T X v_k = v_k{}^T \lambda_k v_k = \lambda_k v_k{}^T v_k = \lambda_k$$

for all $k=1...p$

- Small λ_k ⇔ small variance ⇔ data change little in the direction of component v_k

PCA is useful for finding new, more informative, uncorrelated features; it reduces dimensionality by rejecting low variance features

PCA Eigenvalues



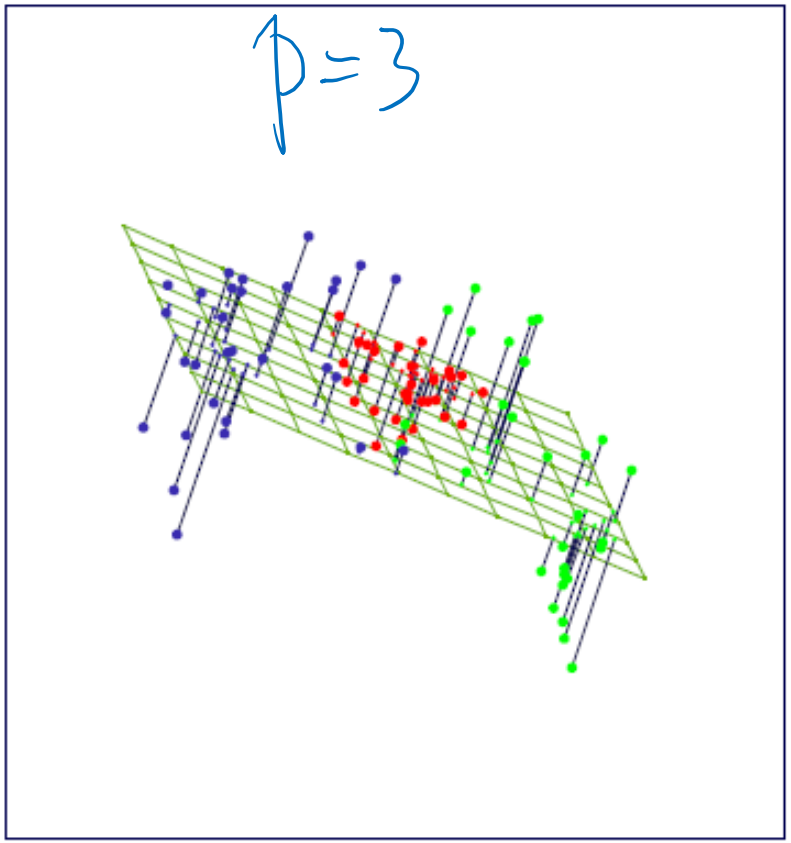
PCA Summary until now

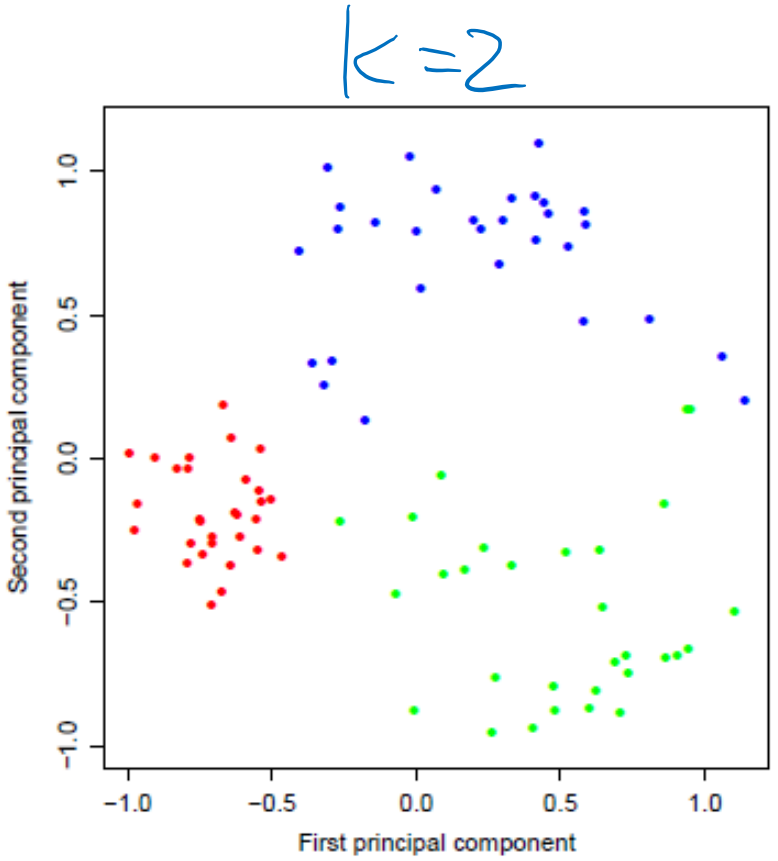
- Rotates multivariate dataset into a new configuration which is easier to interpret

- PCA is useful for finding new, more informative, uncorrelated features; it reduces dimensionality by rejecting low variance features

  - ✓ PCA compresses (i.e. perform projection) the data points by only using the top few eigenvectors.
  - ✓ This corresponds to choosing a "linear subspace" represent points on a line, plane, or "hyper-plane"

PCA for dimension reduction
e.g. p=3 ➔ (pick top k=2 PCs)





corresponds to choosing a "2D linear plane"

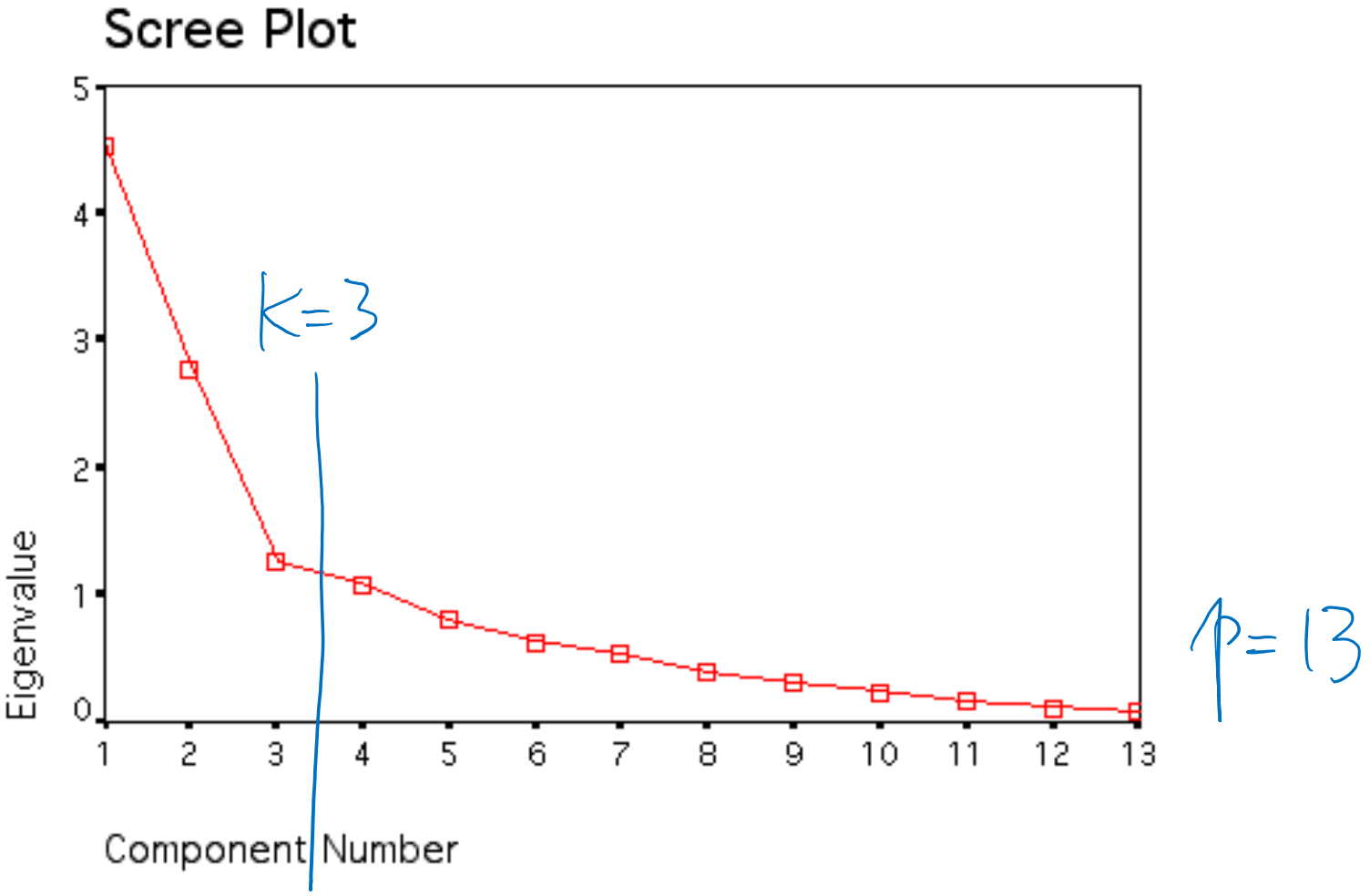
How many components to keep?

- I. Variance: Enough PCs to have a cumulative variance explained by the PCs that is >50-70%

- II. Scree plot: represents the ability of PCs to explain the variation in data, e.g. keep PCs with eigenvalues >1

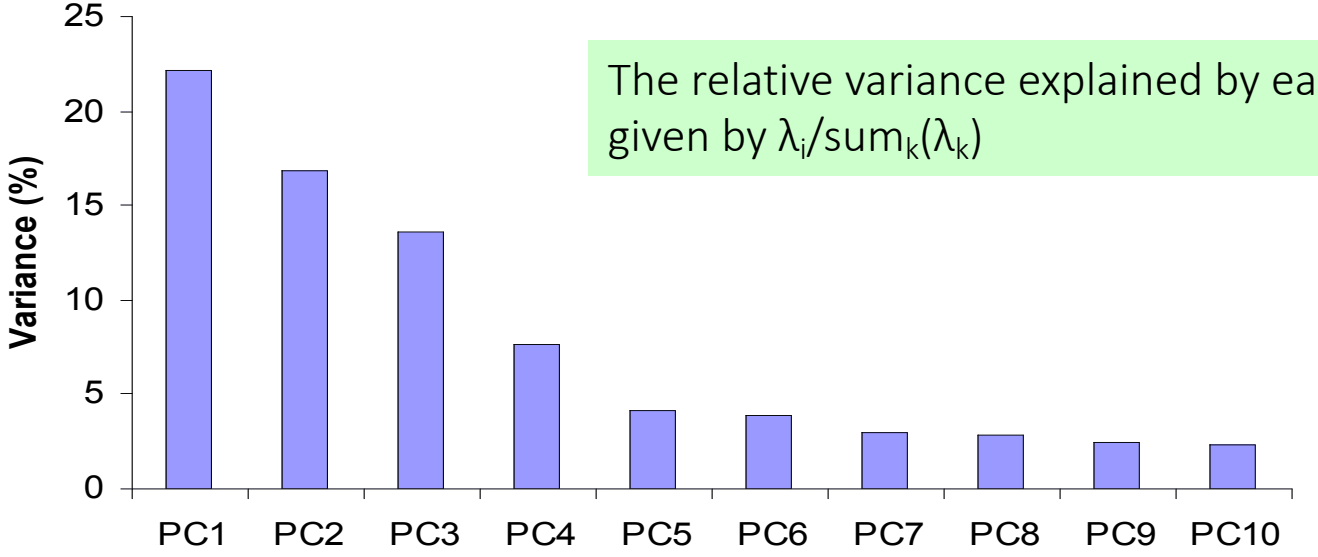
From Dr. S. Narasimhan

e.g. check eigenvalue (I)



e.g. check percentage of kept variance

Can ignore the components of lesser significance.



The relative variance explained by each PC is given by $\lambda_i / \text{sum}_k(\lambda_k)$

You do lose some information, but if the eigenvalues are small, you don't lose much

- p dimensions in original data
- Calculate p eigenvectors and eigenvalues
- choose only the first k eigenvectors, by keep enough variance
- final projected data set has only k dimensions

Why to Reduce Dimension?

- PCA as a general dimensionality reduction technique

- Preserves most of variance with a much more compact representation
  - Lower storage requirements (eigenvectors + a few numbers (k) per sample)
  - Faster matching (since matching within a lower-dim)

(1) Limitations of PCA

- PCA is not effective for some datasets.
- For example, if the data is a set of strings
- (1,0,0,0,…), (0,1,0,0…),…,(0,0,0,…,1) then the eigenvalues do not fall off as PCA requires.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

eigenvalue = [1, 1, 1]

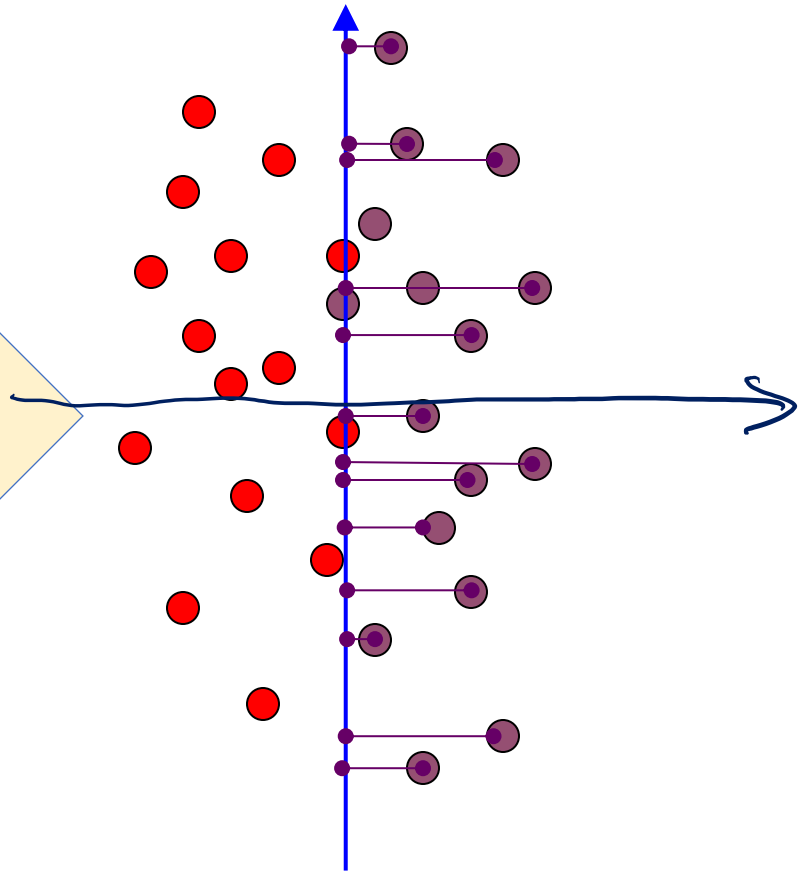
(2) PCA and Discrimination

- The direction of maximum variance is not always good for classification (Example 1)

For this case:

+ Ideal for capturing global variance !

+ Not ideal for discrimination

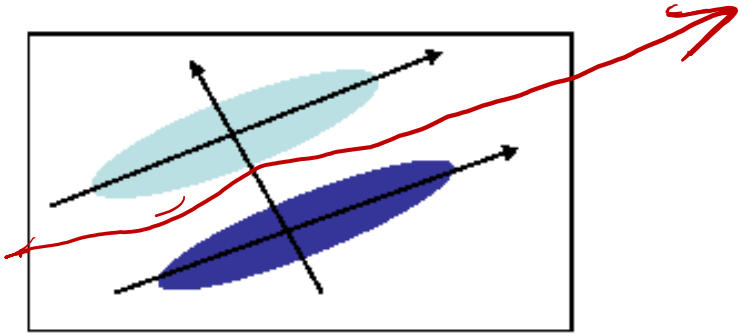


First PC

From Prof. Derek Hoiem

PCA and Discrimination

- PCA may not find the best directions for discriminating between two classes. (Example 2)
- Example:
  - suppose the two classes have 2D Gaussian densities as ellipsoids.
  - 1st eigenvector is best for representing the probabilities / overall data trend
  - 2nd eigenvector is best for discrimination.



From Prof. Derek Hoiem

Algebraic Review

- How many eigenvectors are there?
- For Real Symmetric Matrices
  - except in degenerate cases when eigenvalues repeat, there are p eigenvectors

    $u_1,\ldots,u_p$ are the eigenvectors

    $\lambda_1 \ldots \lambda_p$ are the eigenvalues, large to small, ordered by its value

  - all eigenvectors are mutually orthogonal and therefore form a new basis space
    - Eigenvectors for distinct eigenvalues are mutually orthogonal
    - Eigenvectors corresponding to the same eigenvalue have the property that any linear combination is also an eigenvector with the same eigenvalue; one can then find as many orthogonal eigenvectors as the number of repeats of the eigenvalue.

From Dr. S. Narasimhan

Today

- Dimensionality Reduction (unsupervised) with Principal Components Analysis (PCA)
  - Review of eigenvalue, eigenvector
  - How to project samples into a line capturing the variation of the whole dataset ➔ Eigenvector / Eigenvalue of covariance matrix
  - PCA for dimension reduction
  - Eigenface ➔ PCA for face recognition

Example 1: Application to image, e.g. a task of face recognition

1. Treat pixels as a vector

$$\mathbf{x}$$

2. Recognize face by 1-nearest neighbor

$$\mathbf{y}_1 \cdots \mathbf{y}_n$$

A face-image database of totally n different people

$$k = \underset{k}{\operatorname{argmin}} \left\| \mathbf{y}_k^{\,T} - \mathbf{x} \right\|$$

From Prof. Derek Hoiem

Example 1: the space of all face images

- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
  - Slow and lots of storage

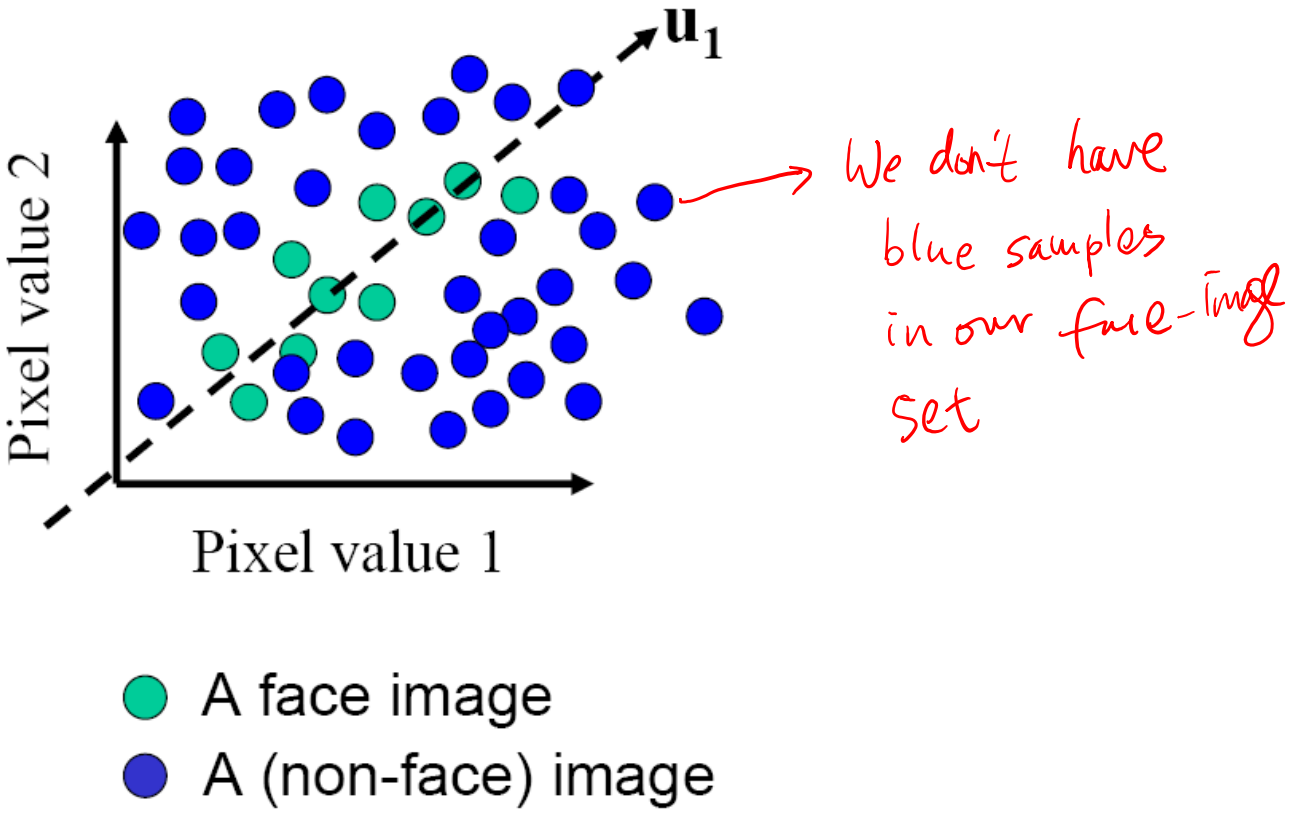
$p = 10,000$

- But very few 10,000-dimensional vectors are valid face images
- We want to effectively model the subspace of face images

From Prof. Derek Hoiem

Example 1:The space of all face images

- Eigenface idea: construct a low-dimensional linear subspace that best explains the variation in the set of face images



From Prof. Derek Hoiem

Example 1: Application to Faces, e.g. Eigenfaces (PCA on face images)

1. Compute covariance matrix of face images

2. Compute the principal components ("eigenfaces")
   - K eigenvectors with largest eigenvalues

3. Represent all face images in the dataset as linear combinations of eigenfaces
   - Perform nearest neighbors on these projected low-d coefficients

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

Example 1: Application to Faces

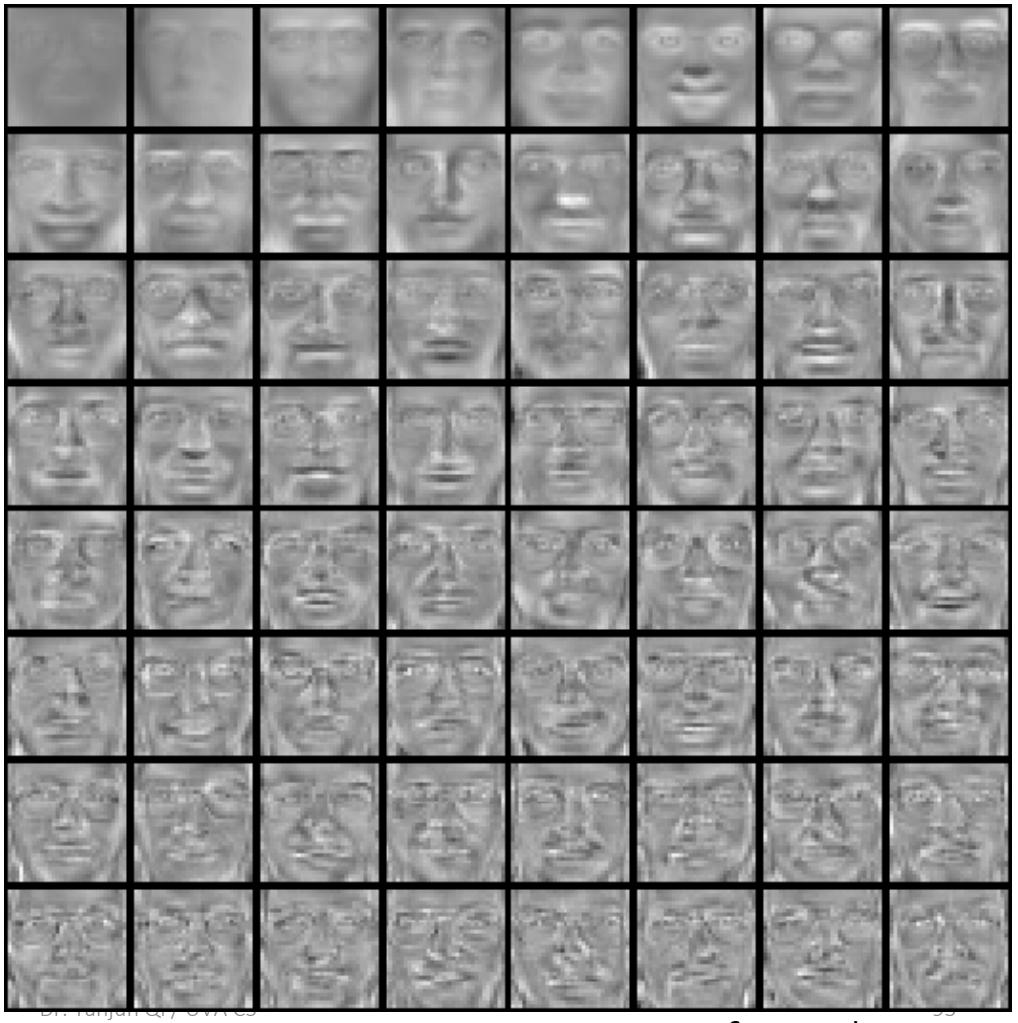
Training images

Example 1: Eigenfaces example

$C = (X - \bar{X})^T (X - \bar{X})$

Top eigenvectors: $u_1, \ldots u_k$

$k = 64$

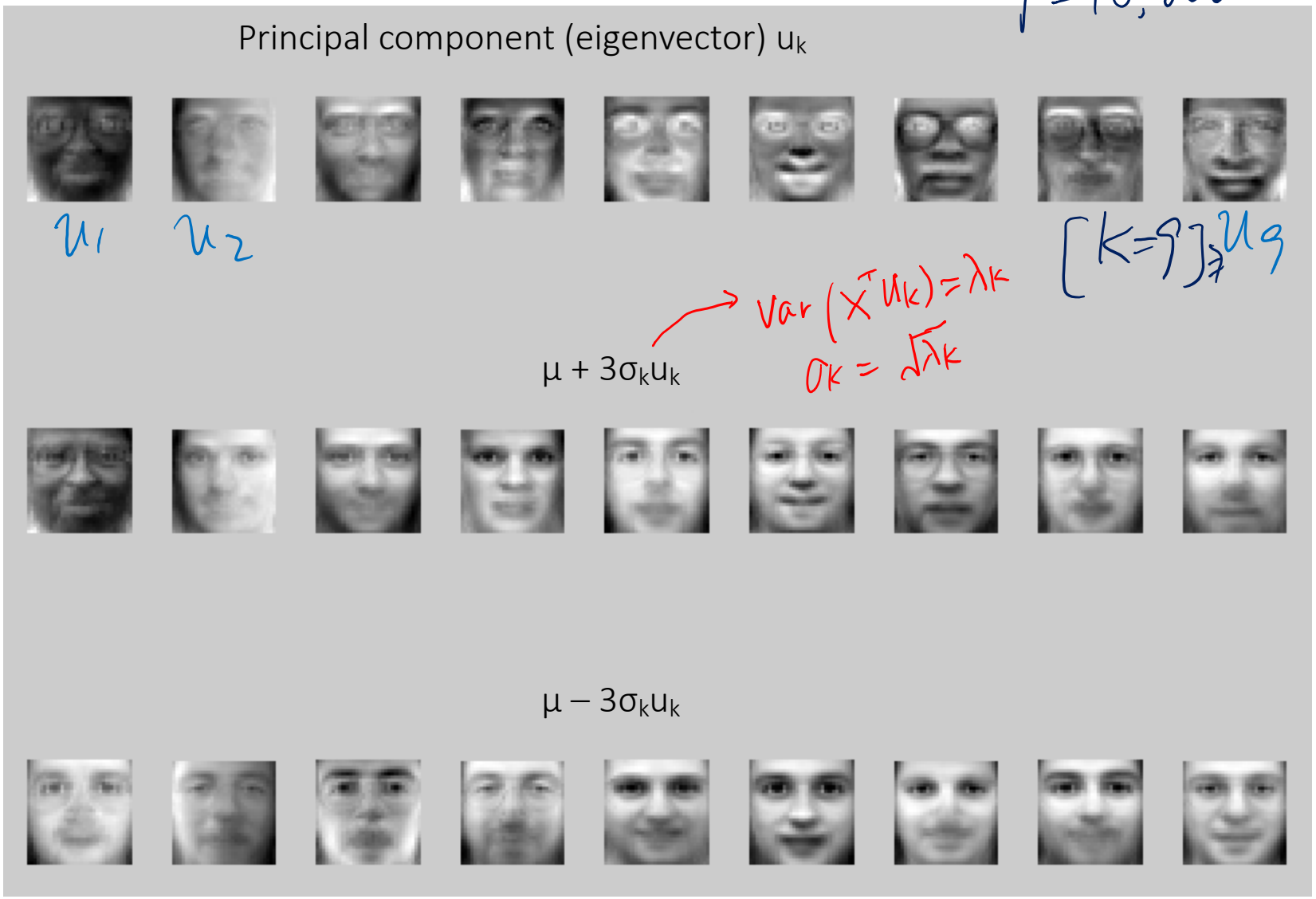


Mean: μ

$$\bar{X} = \mu = \frac{1}{N} \sum_{k=1}^{N} x_k$$

From Prof. Derek Hoiem

Example 1: Visualization of eigenfaces



$p = 10,000$

$[k = 9] \Rightarrow u_9$

$\text{Var}(X^T u_k) = \lambda_k$

$\sigma_k = \sqrt{\lambda_k}$

From Prof. Derek Hoiem

Example 1: Representation and reconstruction of original x

- Face x in "face space" coordinates:

$$\mathbf{x} \rightarrow \left[\mathbf{u}_1^{\mathrm{T}}(\mathbf{x}-\mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x}-\mu)\right]$$

$$g = \left[w_1, \ldots, w_k \right]$$

New representation

Remarkably few eigenvector terms are needed to give a fair likeness of most people's faces.

➔ subtract the mean along each dimension, in order to center the original axis system at the centroid of all data points

Representation and reconstruction

- Face x in "face space" coordinates:

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x}-\mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x}-\mu)]$$

$$= \quad w_1, \ldots, w_k$$

New representation

- Reconstruction:

$$\Rightarrow \|x - \hat{x}\|^2 \quad \text{reconstruction error}$$

$\hat{x}$ = μ + $w_1u_1+w_2u_2+w_3u_3+w_4u_4+ \ldots$ $+w_ku_k$

A human face may be considered to be a linear combination of these standardized eigen faces

From Prof. Derek Hoiem

⓪ Assuming centering data

① $\vec{u}_1, \vec{u}_2, \ldots, \vec{u}_k$ Top k Eigen Vectors.

each $\vec{u}_i$ is $p \times 1$ column vector

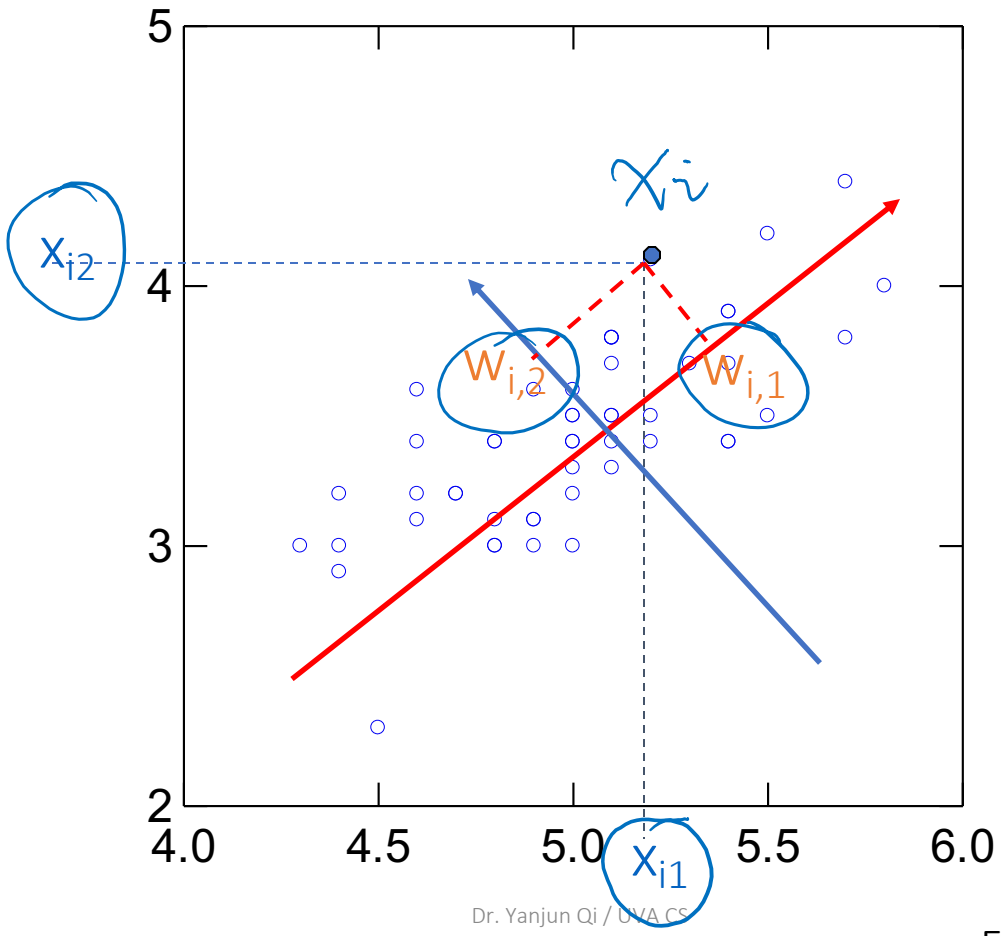
② $\vec{g}_i = \left[\vec{x}_i^T \vec{u}_1, \vec{x}_2^T \vec{u}_2, \ldots, \vec{x}_i^T \vec{u}_k \right]^T$ $k \times 1$ column vector

$$\hat{\vec{x}}_i^T = \underset{1 \times k}{\vec{g}_i^T} \begin{bmatrix} \vec{u}_1^T \\ \vec{u}_2^T \\ \vdots \\ \vec{u}_k^T \end{bmatrix}_{k \times p}$$

$$= \left[\vec{x}_i^T \vec{u}_1, \vec{x}_2^T \vec{u}_2, \ldots, \vec{x}_i^T \vec{u}_k \right] \begin{bmatrix} \vec{u}_1^T \\ \vec{u}_2^T \\ \vdots \\ \vec{u}_k^T \end{bmatrix}$$

$$= \vec{x}_i^T \sum_{i=1}^{k} \vec{u}_i \vec{u}_i^T$$

New representation in the lower-dim PC space



original

$x_i \rightarrow [x_{i1}, x_{i2}]$

$\Downarrow$

$g_i \rightarrow [w_{i1}, w_{i2}]$

projected

$[x_i^T u_1, x_i^T u_2]$

From Prof. Derek Hoiem

Key Property of Eigenspace Representation

Given

- 2 images $\hat{x}_1, \hat{x}_2$ that are used to construct the Eigenspace
- $\hat{g}_1$ is the eigenspace projection of image $\hat{x}_1$
- $\hat{g}_2$ is the eigenspace projection of image $\hat{x}_2$

Then,

$$\| \hat{g}_2 - \hat{g}_1 \| \approx \| \hat{x}_2 - \hat{x}_1 \|$$

That is, distance in Eigenspace is approximately equal to the distance between two original images.

Classify / Recognition with eigenfaces

Step I: Process labeled training images

- Find mean μ and covariance matrix
- Find k principal components (i.e. eigenvectors of Σ) ➔ $u_1,\ldots u_k$
- Project each training image x_i onto subspace spanned by the top principal components:
  $(w_{i1},\ldots,w_{ik}) = (u_1^T(x_i - \mu), \ldots, u_k^T(x_i - \mu))$

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

Classify / Recognition with eigenfaces

Step 2: Nearest neighbor based face classification

Given a novel image x

- Project onto k PC's subspace:
  $(w_1,\ldots,w_k) = (u_1^T(x-\mu), \ldots, u_k^T(x-\mu))$
- Optional: check reconstruction error $x - \hat{x}$ to determine whether the image is really a face
- Classify as closest training face(s) in the lower k-dimensional subspace

M. Turk and A. Pentland, Face Recognition using Eigenfaces, CVPR 1991

Is this a face or not?

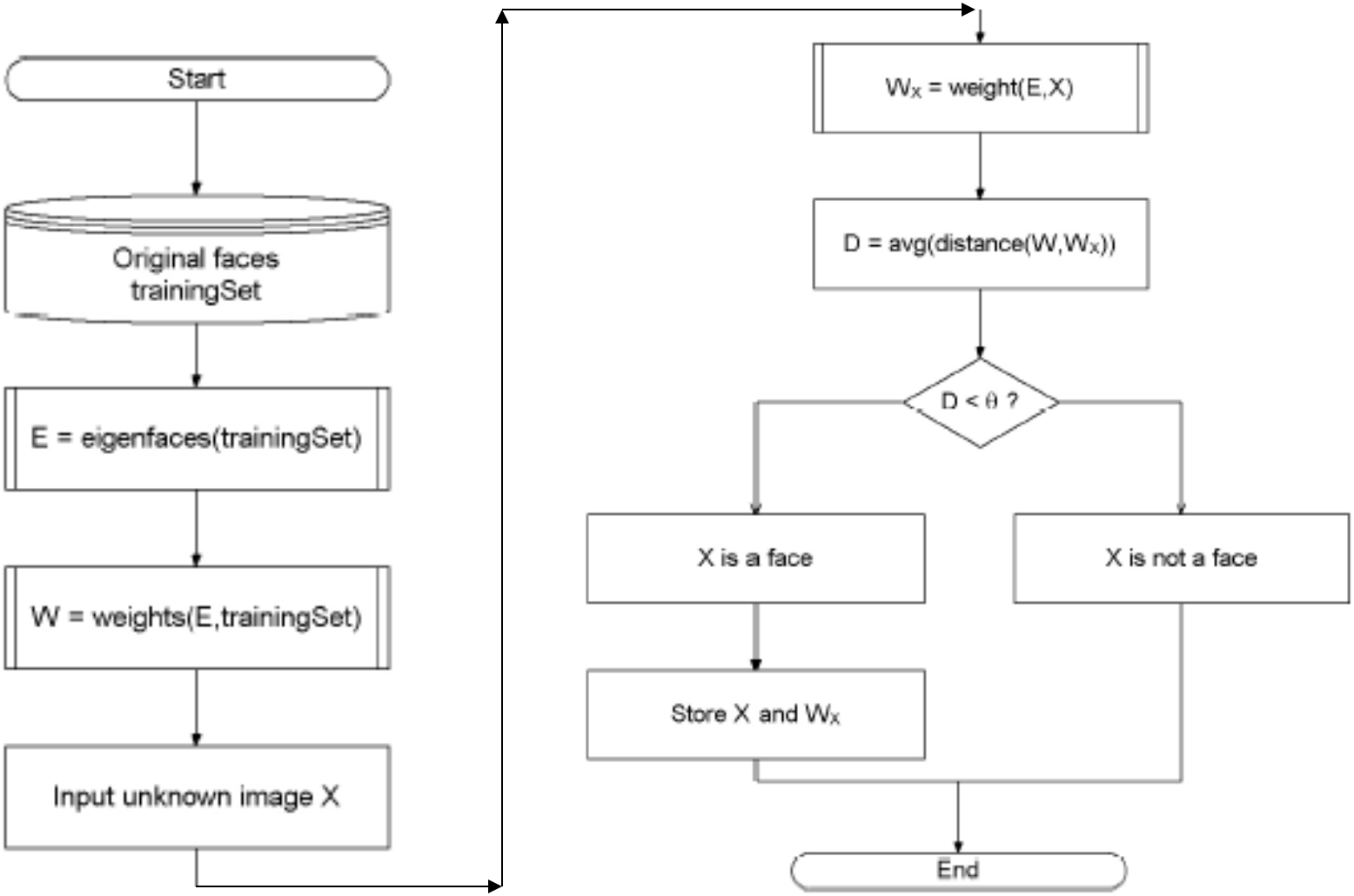


Figure 1: High-level functioning principle of the eigenface-based facial recognition algorithm

Example 2: e.g. Handwritten Digits

- 16 x 16 gray scale
- Total 658 such 3's
- 130 is shown
- Image $x_i : R^{256}$
- Compute principal components

$p=256$

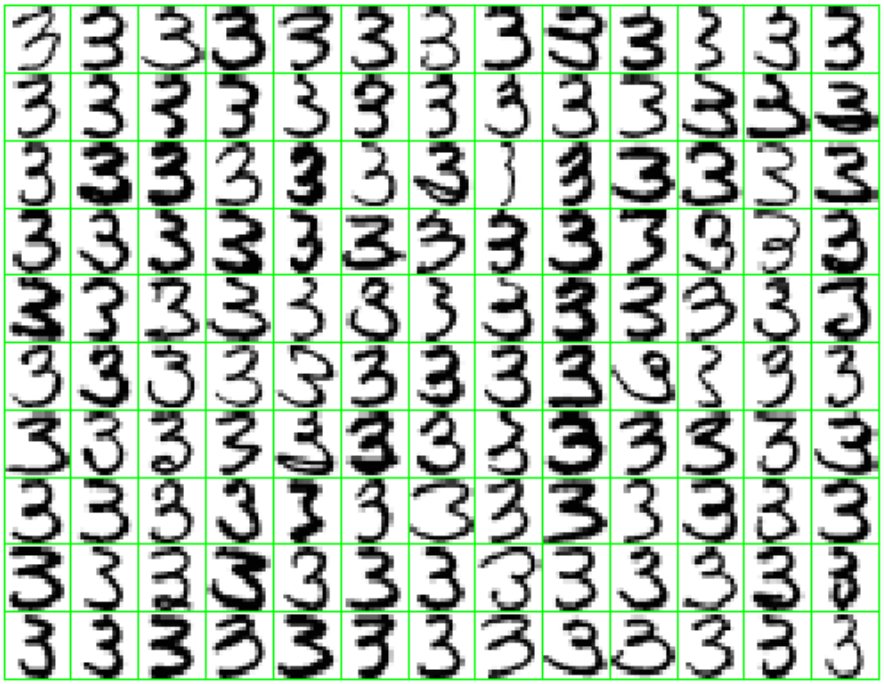
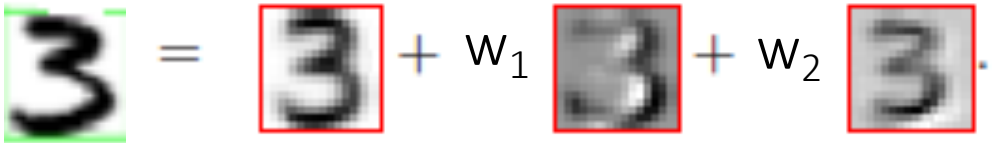


FIGURE 14.22. *A sample of 130 handwritten 3's shows a variety of writing styles.*

$\hat{x} = \mu + v_1 + v_2$ $K=2$



e.g. From ESL book

$$\mathbf{x} \rightarrow [\mathbf{u}_1^{\mathrm{T}}(\mathbf{x}-\mu), \ldots, \mathbf{u}_k^{\mathrm{T}}(\mathbf{x}-\mu)]$$

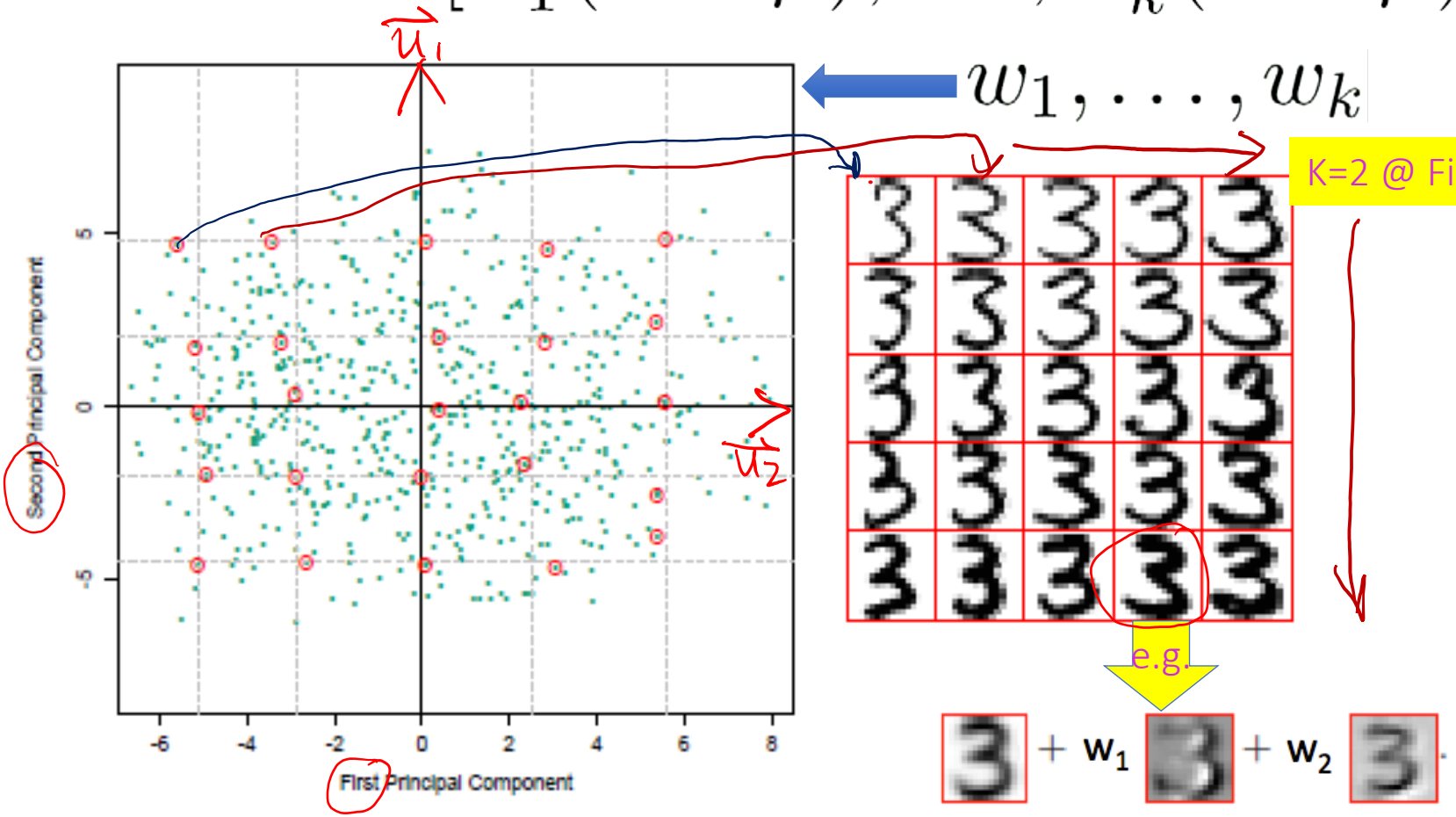
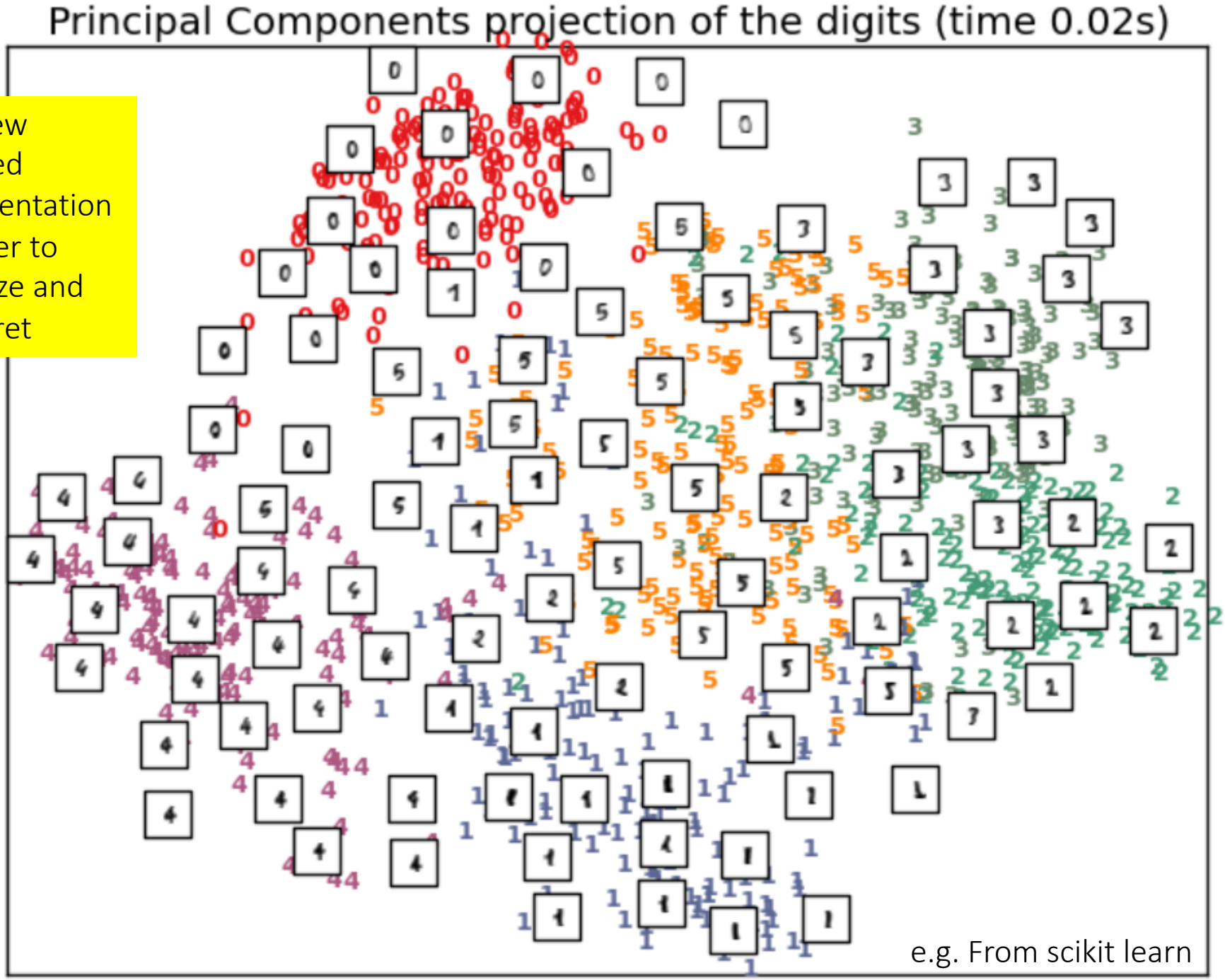


FIGURE 14.23. *(Left panel:) the first two principal components of the hand-written threes. The circled points are the closest projected images to the vertices of a grid, defined by the marginal quantiles of the principal components. (Right panel:) The images corresponding to the circled points. These show the nature of the first two principal components.*

e.g. From ESL book


Principal Components projection of the digits (time 0.02s)
The new reduced representation is easier to visualize and interpret
e.g. From scikit learn

Extra: A 2D Numerical Example

From Dr. S. Narasimhan

PCA Example –STEP 1

- Subtract the mean from each of the data dimensions.

- 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.

From Dr. S. Narasimhan

PCA Example –STEP 1

DATA: (p=2)

x1	x2
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

ZERO MEAN DATA:

x1	x2
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01

From Dr. S. Narasimhan

PCA Example –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 the x1 and x2 variable increase together.

From Dr. S. Narasimhan

PCA Example –STEP 3

- Calculate the eigenvectors and eigenvalues of the covariance matrix

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

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

From Dr. S. Narasimhan

PCA Example –STEP 3

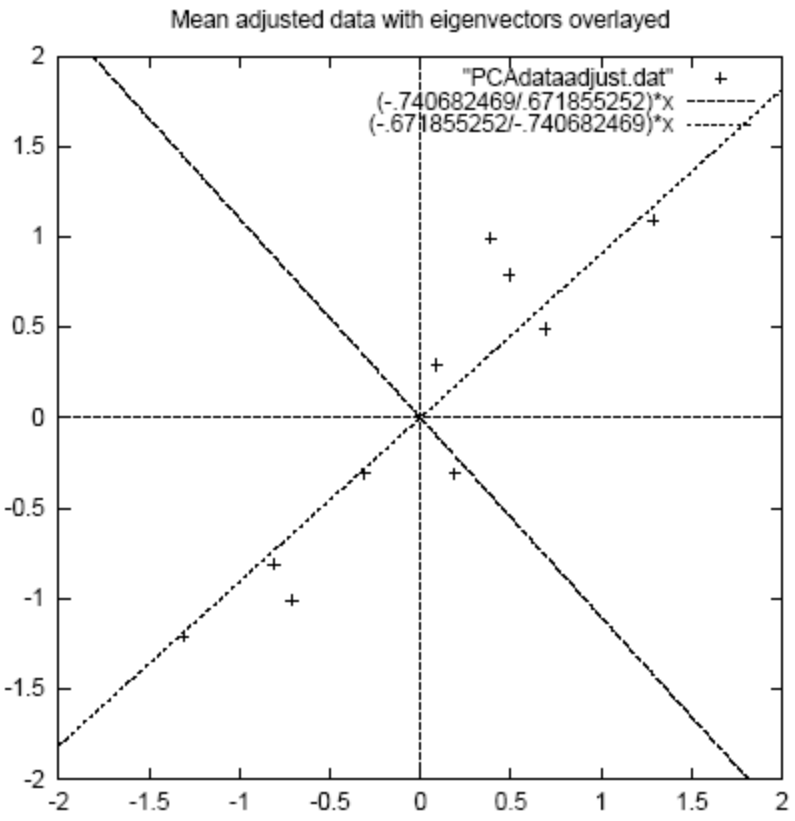


Figure 3.2: A plot of the normalised data (mean subtracted) with the eigenvectors of the covariance matrix overlayed on top.

- 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.

From Dr. S. Narasimhan

PCA Example –STEP 4

- Reduce dimensionality and form *feature vector*

  the eigenvector with the *highest* eigenvalue is the *principle component* of the data set.

  In our example, the eigenvector with the largest 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.

From Dr. S. Narasimhan

PCA Example –STEP 4

- Feature Vector

FeatureVector = (eig_1 eig_2 eig_3 … 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:

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

Now, if you like, you can decide to ignore the components of lesser significance.

You do lose some information, but if the eigenvalues are small, you don't lose much

PCA Example –STEP 5

- Deriving the new data

  FinalData = RowFeatureVector x RowZeroMeanData

RowFeatureVector is the matrix with the eigenvectors in the columns *transposed* so that the eigenvectors are now in the rows, with the most significant eigenvector at the top

RowZeroMeanData is the mean-adjusted data *transposed*, ie. the data items are in each column, with each row holding a separate dimension.

From Dr. S. Narasimhan

PCA Example –STEP 5

FinalData transpose: dimensions along columns

w1	w2
-.827970186	-.175115307
1.77758033	.142857227
-.992197494	.384374989
-.274210416	.130417207
-1.67580142	-.209498461
-.912949103	.175282444
.0991094375	-.349824698
1.14457216	.0464172582
.438046137	.0177646297
1.22382056	-.162675287

From Dr. S. Narasimhan

PCA Example –STEP 5

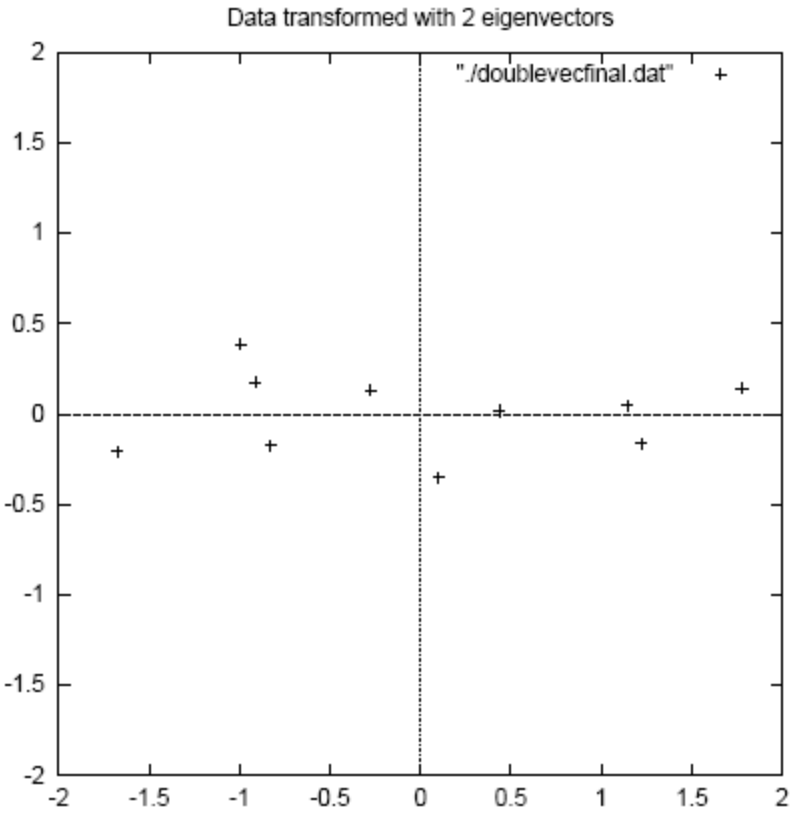


Figure 3.3: The table of data by applying the PCA analysis using both eigenvectors, and a plot of the new data points.

From Dr. S. Narasimhan

Reconstruction of original Data

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

- In our example let us assume that we considered only the w1 dimension…

From Dr. S. Narasimhan

Reconstruction of original Data

w1

-.827970186

1.77758033

-.992197494

-.274210416

-1.67580142

-.912949103

.0991094375

1.14457216

.438046137

1.22382056

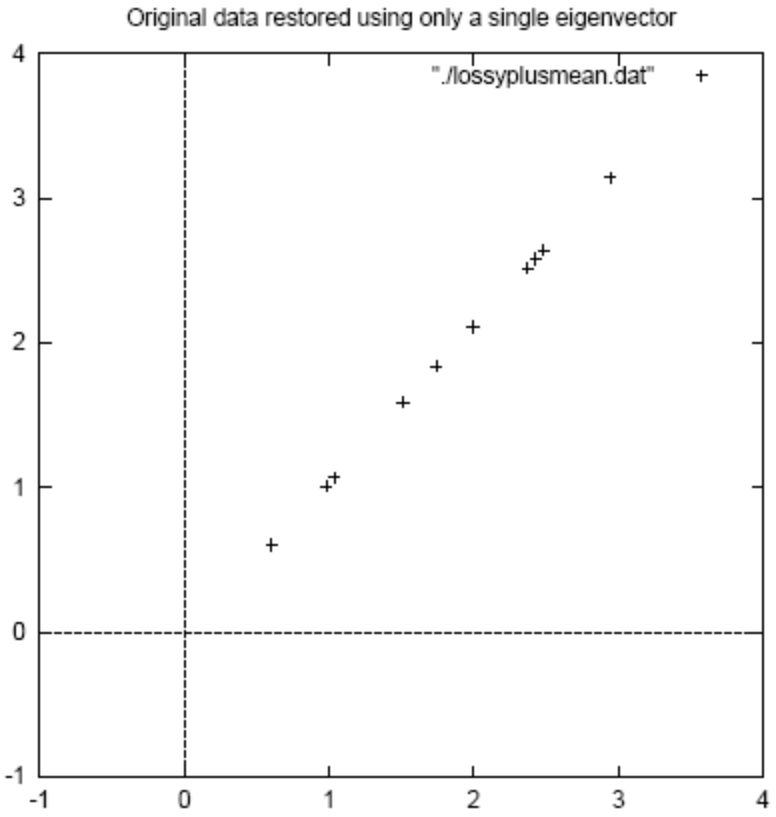


Figure 3.5: The reconstruction from the data that was derived using only a single eigenvector

From Dr. S. Narasimhan