Machine Learning from Data

Lecture 27: Spring 2021

Today’s Lecture

- Input Preprocessing
- Dimensionality Reduction and Feature Selection
- Principal Components Analysis (PCA)
- Hints, Data Cleaning, Validation

Learning Aides

Additional tools that can be applied to all techniques

Preprocess data to account for arbitrary choices during data collection (input normalization)

Remove irrelevant dimensions that can mislead learning (PCA)

Incorporate known properties of the target function (hints and invariances)

Remove detrimental data (deterministic and stochastic noise)

Better ways to validate (estimate E_{out}) for model selection

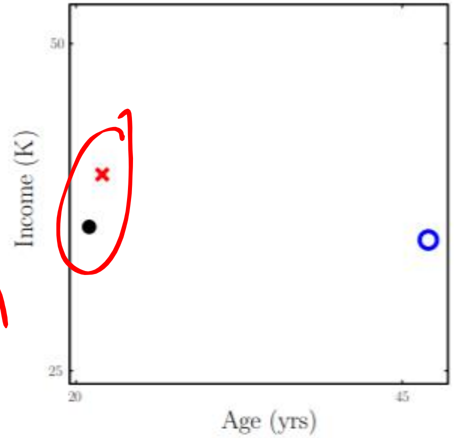
Nearest Neighbor

Mr. Good and Mr. Bad were both given credit cards by the Bank of Learning (BoL).

	Mr. Good	Mr. Bad
(Age in years, Income in $\$ \times 1,000$)	(47,35)	(22,40)

Age

Mr. Unknown who has "coordinates" (21yrs,$36K) applies for credit. Should the BoL give him credit, according to the nearest neighbor algorithm?



Decline

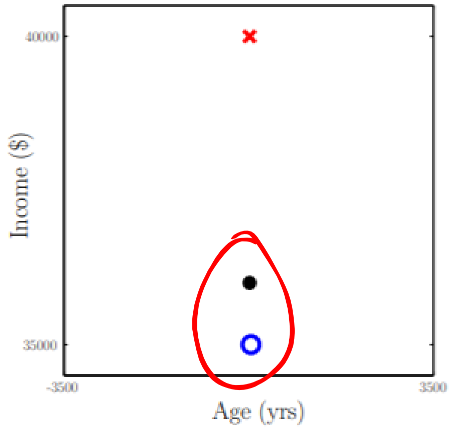
Nearest Neighbor Uses Euclidean Distance

Mr. Good and Mr. Bad were both given credit cards by the Bank of Learning (BoL).

	Mr. Good	Mr. Bad
(Age in years, Income in \$)	(47,35000)	(22,40000)

Mr. Unknown who has "coordinates" (21yrs,\$36000) applies for credit. Should the BoL give him credit, according to the nearest neighbor algorithm?

What if, income is measured in dollars instead of "K" (thousands of dollars)?



Learning →

Approve Similarity

Built-in Scale

Uniform Treatment of Dimensions

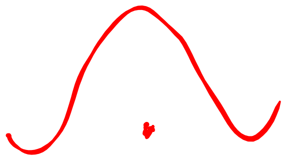
Most learning algorithms treat each dimension equally

Nearear neighbor: $d(\mathbf{x}, \mathbf{x}') = \| \mathbf{x} - \mathbf{x}' \|$

Weight Decay: $\Omega(\mathbf{w}) = \lambda \mathbf{w}^{\mathrm{T}} \mathbf{w}$

SVM: margin defined using Euclidean distance

RBF: bump function decays with Euclidean distance



Input Preprocessing

Unless you want to emphasize certain dimensions, the data should be *preprocessed* to present each dimension on an equal footing

Input Preprocessing is a Data Transform

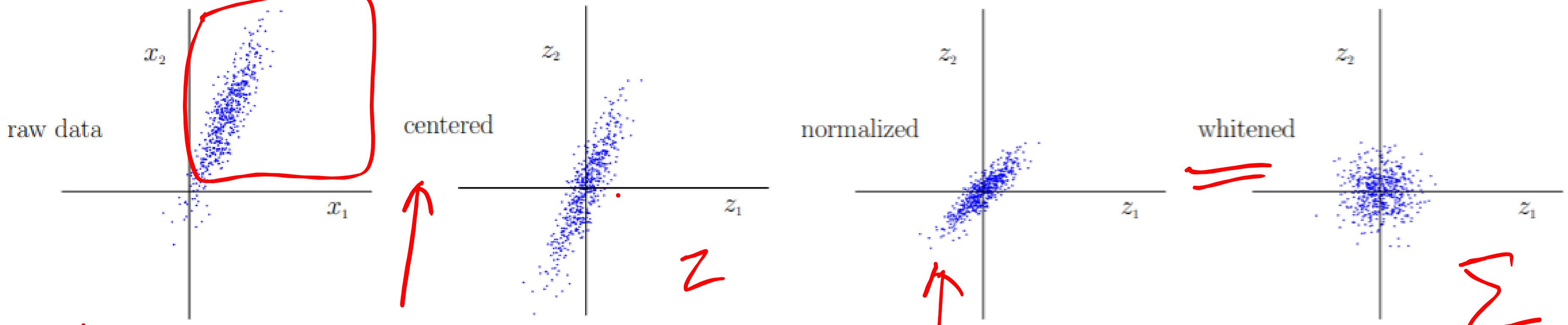
$$X = \begin{bmatrix} - & \mathbf{x}_1^{\mathrm{T}} & - \\ - & \mathbf{x}_2^{\mathrm{T}} & - \\ & \vdots & \\ - & \mathbf{x}_N^{\mathrm{T}} & - \end{bmatrix}$$

$$\mathbf{x}_n \mapsto \mathbf{z}_n$$

$$g(\mathbf{x}) = \tilde{g}(\Phi(\mathbf{x})).$$

Raw $\{\mathbf{x}_n\}$ have (for example) arbitrary scalings in each dimension, and $\{\mathbf{z}_n\}$ will not.

Bias Scale Correlation



$$\boxed{\mathbf{z}_n = \mathbf{x}_n - \bar{\mathbf{x}}}$$

average

$$\bar{\mathbf{z}} = \mathbf{0}$$

$$\mathbf{z}_n = \mathrm{D}\mathbf{x}_n$$

$$D_{ii} = \frac{1}{\sigma_i}$$

$$\tilde{\sigma}_i = 1$$

$$\mathbf{z}_n = \Sigma^{-\frac{1}{2}}\mathbf{x}_n$$

$$\Sigma = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^{\mathrm{T}} = \frac{1}{N}\mathrm{X}^{\mathrm{T}}\mathrm{X}$$

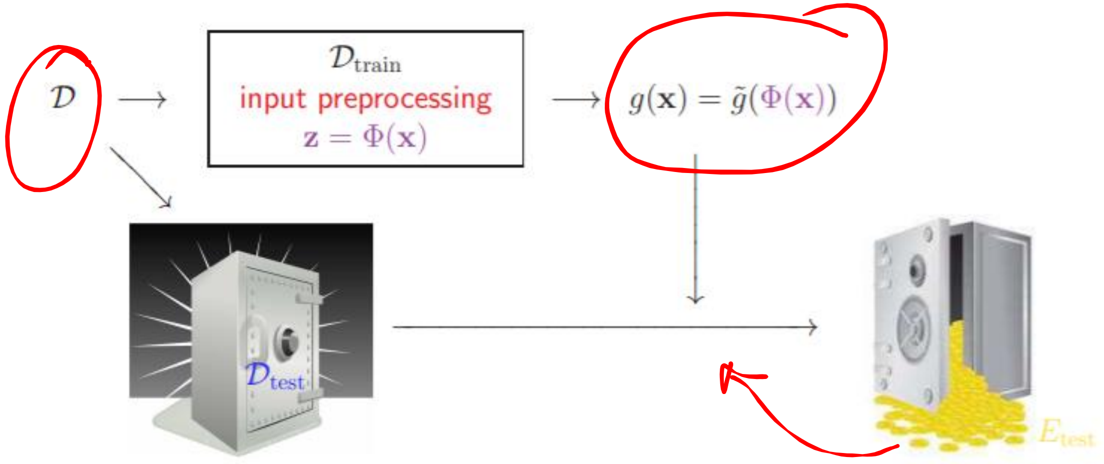
$$\widetilde{\Sigma} = \frac{1}{N}\mathrm{Z}^{\mathrm{T}}\mathrm{Z} = \mathrm{I}$$

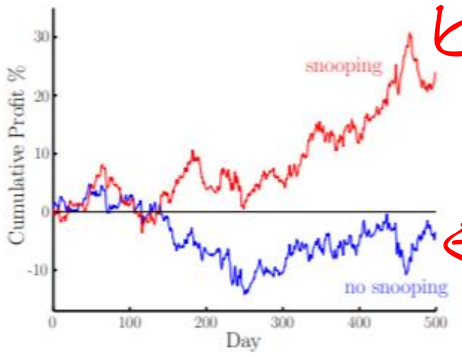
Only Use Training Data For Preprocessing

WARNING!

Transforming data into a more convenient format has a hidden trap which leads to data snooping.

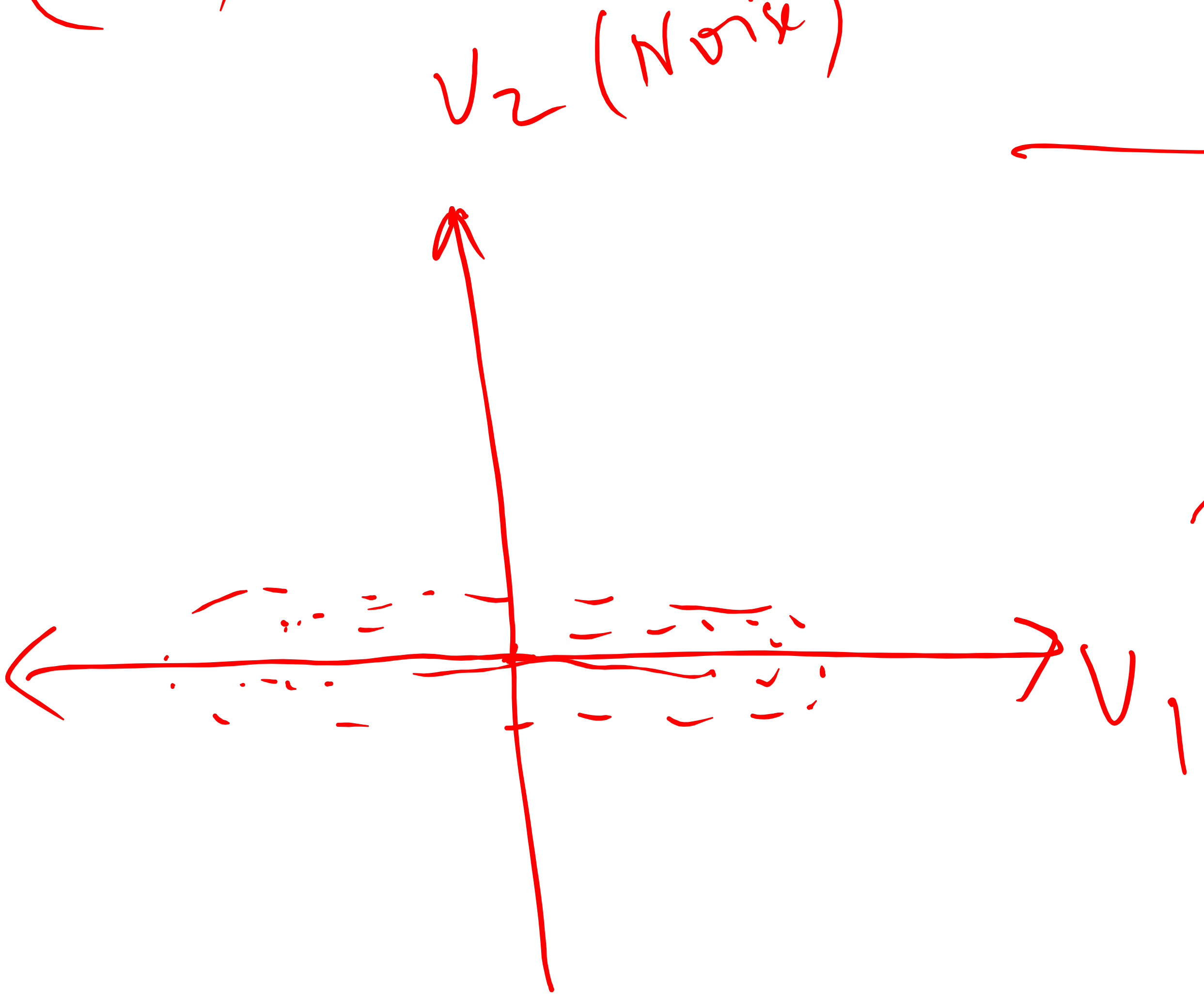
When using a test set, determine the input transformation from training data *only*.
Rule: lock away the test data until you have your final hypothesis.

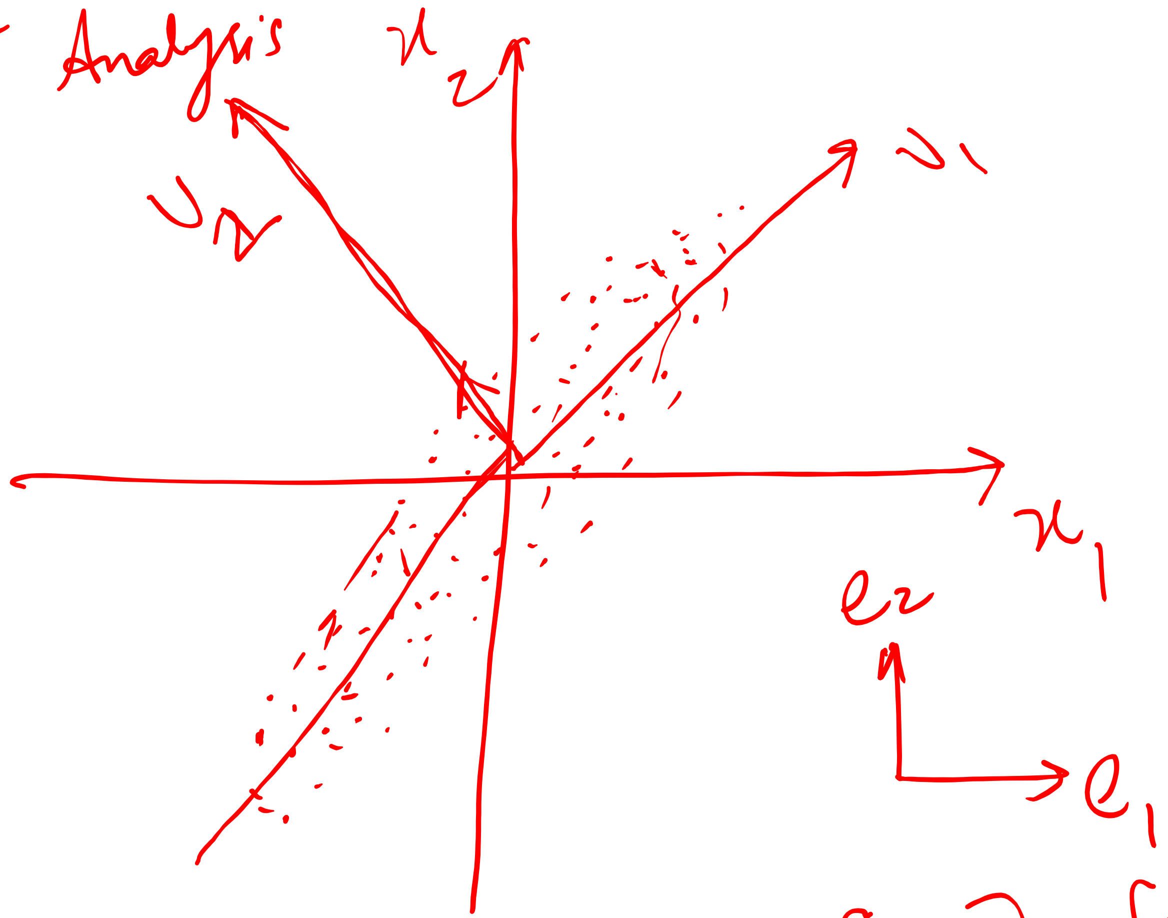




Principal Component Analysis

(PCA)





$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x^T e_1 \\ x^T e_2 \end{bmatrix}$$

Co-ordinates in natural basis:

$$x = \begin{bmatrix} x^T u_1 \\ x^T u_2 \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = z$$

ORTHONORMAL

Rotation

$$x = x_1 e_1 + x_2 e_2 = z_1 v_1 + z_2 v_2$$

NOISE

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \longrightarrow \begin{bmatrix} z_1 \\ \cancel{z_2} \end{bmatrix} \longrightarrow \begin{bmatrix} z_1 \end{bmatrix}$$

$$x \simeq z_1 v_1 \quad (z_2 v_2)$$

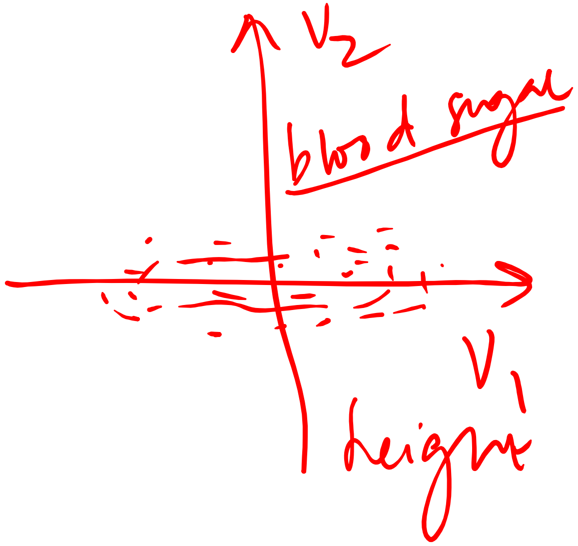
Step1 : Identify dominant directions.

Step2 : Throw away small dimensions:

i) Dimensionality Reduction (Always True)

ii) Irrevalant ~~Rela~~ Info. (Hope)

Drawback :



Dominant direction is v

$$z = x^T v$$

$$\frac{1}{N}\sum_{n=1}^{N} z_n^2 =$$

(Largest squared co-ordinate on avg.)

$$z_n^2 = (x_n^T v)^2 = \underline{\underline{v^T x_n x_n^T v}}$$

$$\frac{1}{N}\sum_{n=1}^{N} v^T x_n x_n^T v = \frac{1}{N} v^T \left(\sum x_n x_n^T\right) v$$

$$= v^T \left(\frac{1}{N}\sum x_n x_n^T\right) v$$

$$= \underline{v^T \Sigma v}$$

$$\Sigma \rightarrow v_1 \quad v_2 \; \cdots \; v_d$$

Eigenvals $\rightarrow \lambda_1 \geq \lambda_2 \geq \cdots \lambda_d \geq 0$

$$v^* = v_1$$

1) Find Σ and eigenvectors $v_1 v_2 \cdots v_d$ } orthogonal basis
$\lambda_1 \lambda_2 \cdots \lambda_d$

max

$$Z = \begin{bmatrix} x^T v_1 \\ x^T v_2 \\ \vdots \\ x^T v_d \end{bmatrix}$$

Direction of decreasing variance

max variance $\perp$ to v_1

2) K-coordinates : Top K PCA directions.

$$x \simeq z_1 v_1 + z_2 v_2 \cdots z_k v_k$$

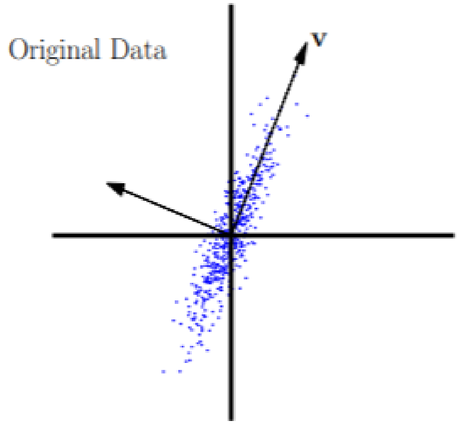
Reconstruction error

The Principal Components

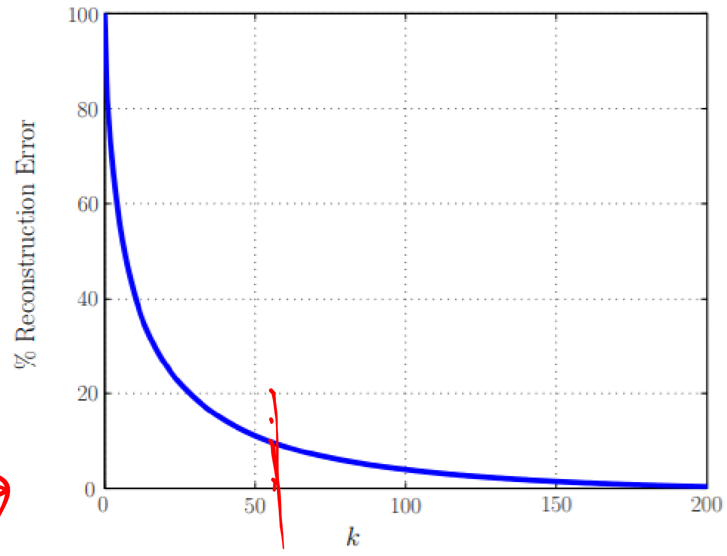
$$\begin{aligned} z_1 &= \mathbf{x}^{\mathrm{T}}\mathbf{v}_1 \\ z_2 &= \mathbf{x}^{\mathrm{T}}\mathbf{v}_2 \\ z_3 &= \mathbf{x}^{\mathrm{T}}\mathbf{v}_3 \\ &\vdots \end{aligned}$$

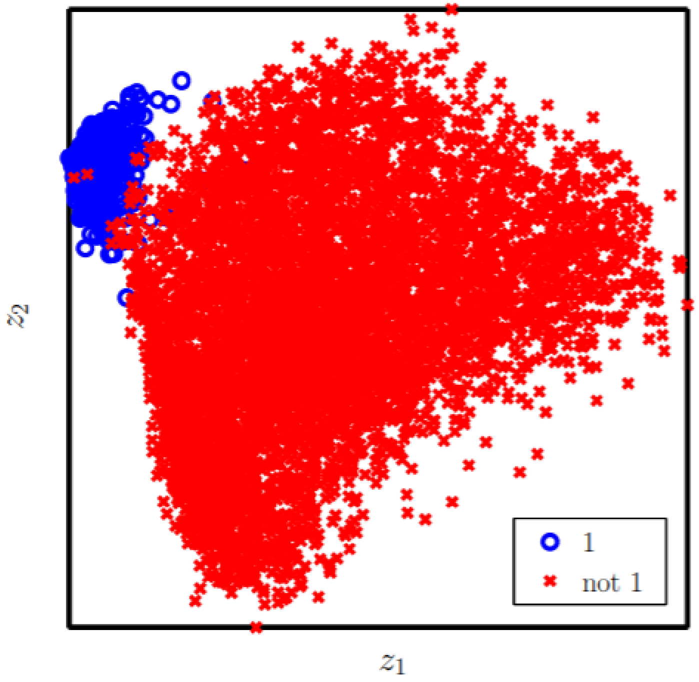
$\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_d$ are the eigenvectors of Σ with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$

Theorem [Eckart-Young]. These directions give best reconstruction of data; also capture maximum variance.



PCA Features for Digits Data





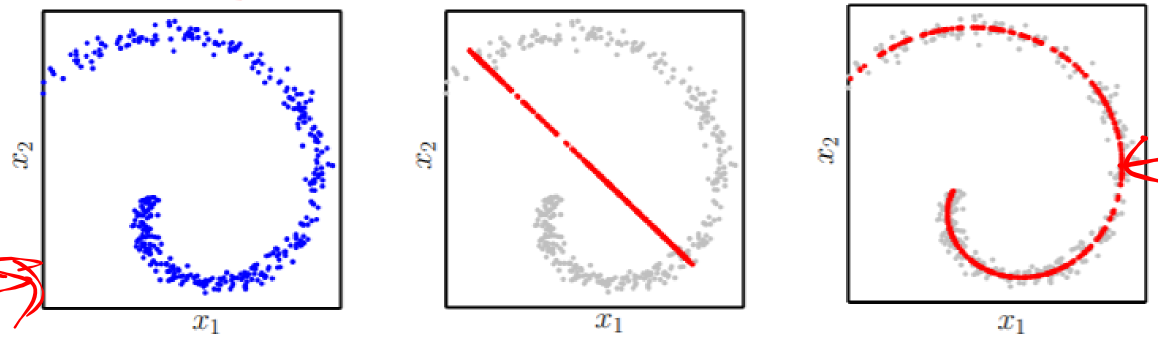
Principal components are **automated**

Captures dominant directions of the data.

May not capture dominant dimensions for f.

Other Learning Aides

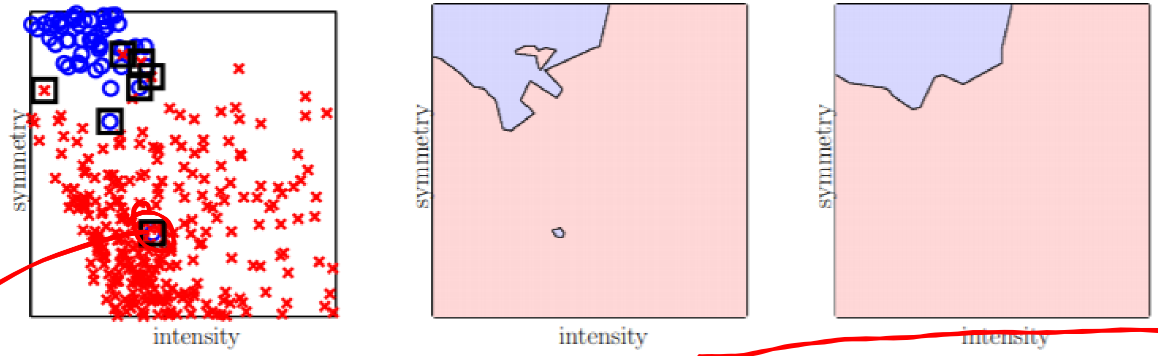
1. **Nonlinear dimension reduction:**



2. **Hints (invariances and prior information):**

   rotational invariance, monotonicity, symmetry, ...

3. **Removing noisy data:**



4. **Advanced validation techniques:** Rademacher and Permutation penalties

   More efficient than CV, more convenient and accurate than VC.

Thanks!