# Machine Learning: a Basic Toolkit

Lorenzo Rosasco,
- Universita' di Genova
- Istituto Italiano di Tecnologia

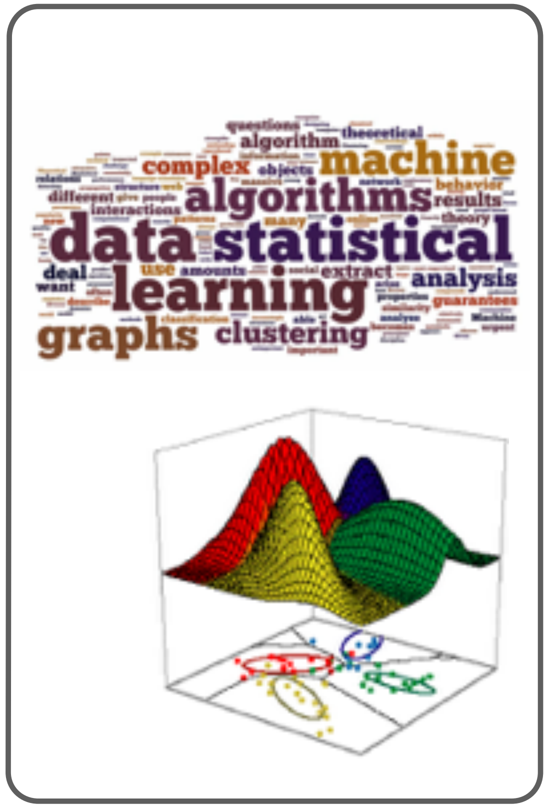
August 2015 - BMM Summer School

# Machine Learning

## Intelligent Systems

## Data Science

Artificial
intelligence
Autonomous Driving
Siri
Use your voice to send messages, set reminders, search for information, and more.



ML Desert Island Compilation

An introduction to *essential* Machine Learning:

- Concepts
- Algorithms

PART I

- Local methods
- **Bias-Variance** and Cross Validation

PART II

- **Regularization** I: Linear Least Squares
- Regularization II: Kernel Least Squares

Morning

PART III

- **Variable Selection**: OMP
- Dimensionality Reduction: PCA

PART IV

- Matlab practical session

Afternoon

# PART I

- Local methods
- Bias-Variance and Cross Validation

**GOAL:** Investigate the trade-off between stability and fitting  starting from simple machine learning approaches

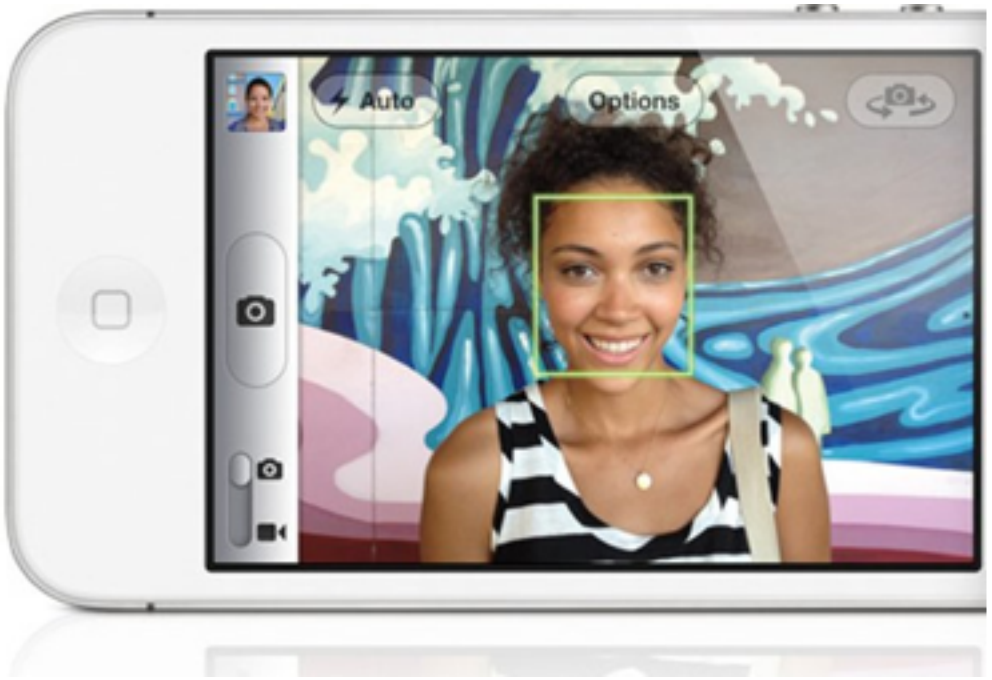
The goal of supervised learning is to find an underlying input-output relation

$$f(x_{\mathrm{new}}) \sim y,$$

given data.

The data, called *training set*, is a set of $n$ input-output pairs,

$$S = \{(x_1, y_1), \dots, (x_n, y_n)\}.$$



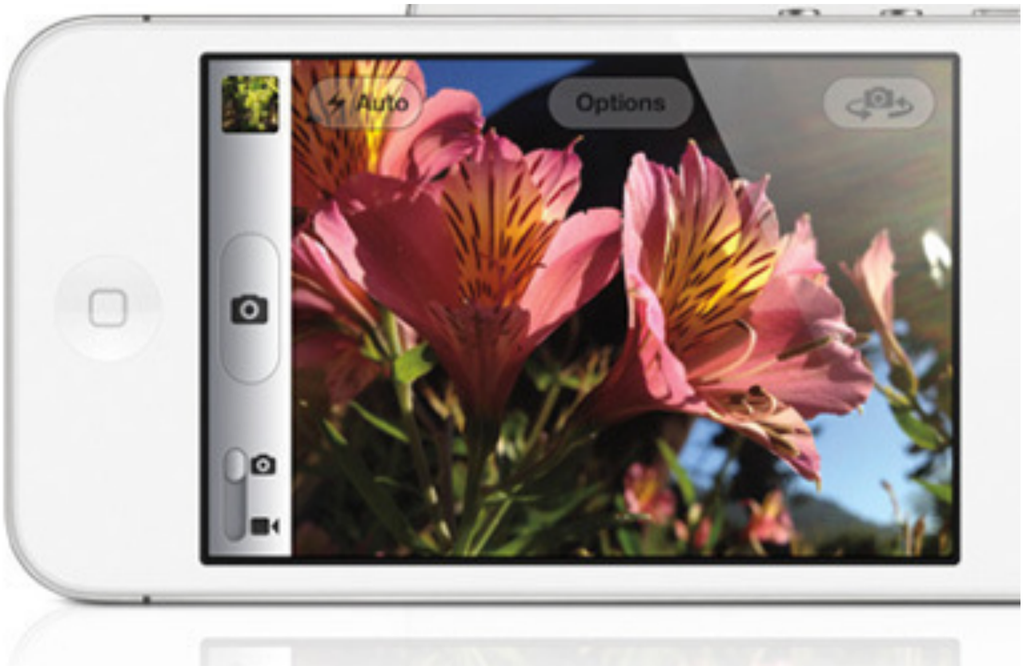
| | | | | | |
|---|---|---|---|---|---|
| 170 | 238 | 85 | 255 | 221 | 0 |
| 68 | 136 | 17 | 170 | 119 | 68 |
| 221 | 0 | 238 | 136 | 0 | 255 |
| 119 | 255 | 85 | 170 | 136 | 238 |
| 238 | 17 | 221 | 68 | 119 | 255 |
| 85 | 170 | 119 | 221 | 17 | 136 |

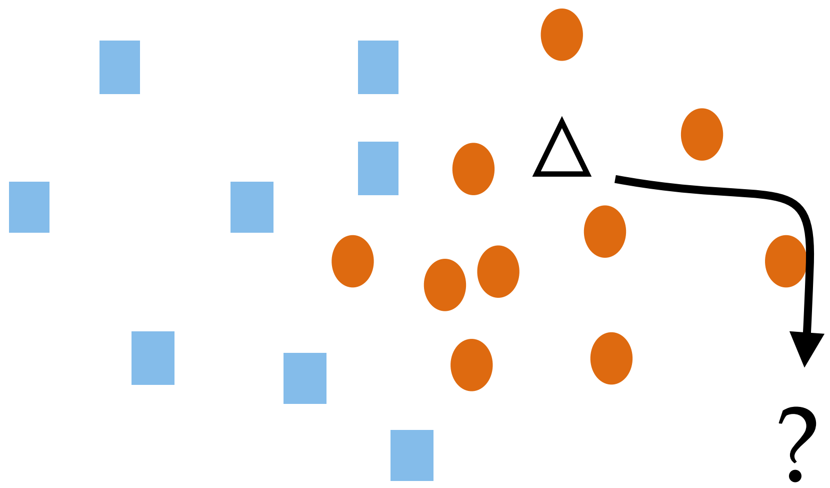
$$X_n = \begin{pmatrix} \boxed{x_1^1 \quad \dots \quad \dots \quad \dots \quad x_1^p} \\ \vdots \qquad \vdots \qquad \vdots \qquad \vdots \qquad \vdots \\ \boxed{x_n^1 \quad \dots \quad \dots \quad \dots \quad x_n^p} \end{pmatrix}$$

$$Y_n = \begin{pmatrix} \boxed{y_1} \\ \vdots \\ \boxed{y_n} \end{pmatrix}$$

$+1$

$-1$





**Local Methods**: Nearby points have similar labels

## Nearest Neighbor

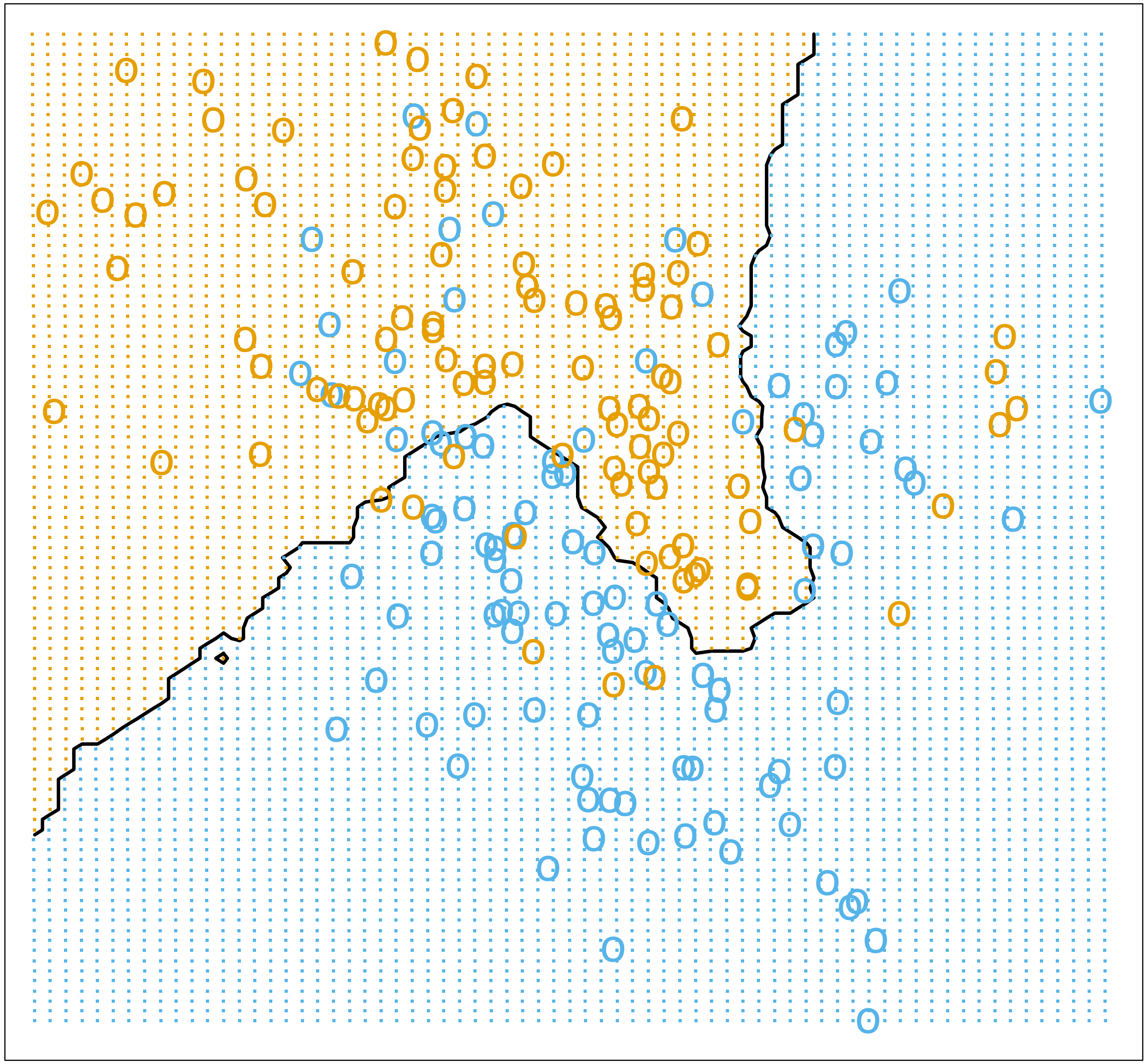
Given an input $\bar{x}$, let

$$i' = \arg\min_{i=1,\ldots,n} \|\bar{x} - x_i\|^2$$

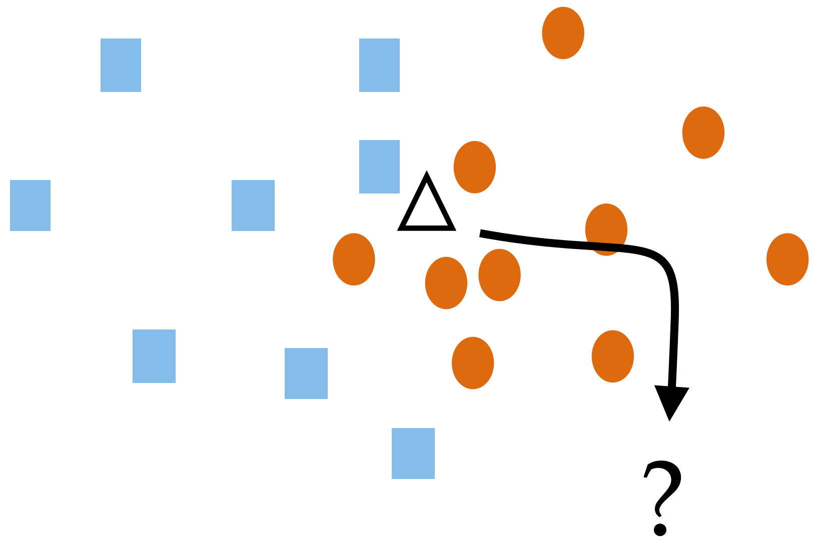
and define the nearest neighbor (NN) estimator as

$$\hat{f}(\bar{x}) = y_{i'}.$$

## How does it work?



# Plot



# K-Nearest Neighbors

Consider

$$d_{\bar{x}} = (\|\bar{x} - x_i\|^2)_{i=1}^n$$

the array of distances of a new point $\bar{x}$ to the input points in the training set. Let

$$s_{\bar{x}}$$

be the above array sorted in increasing order and

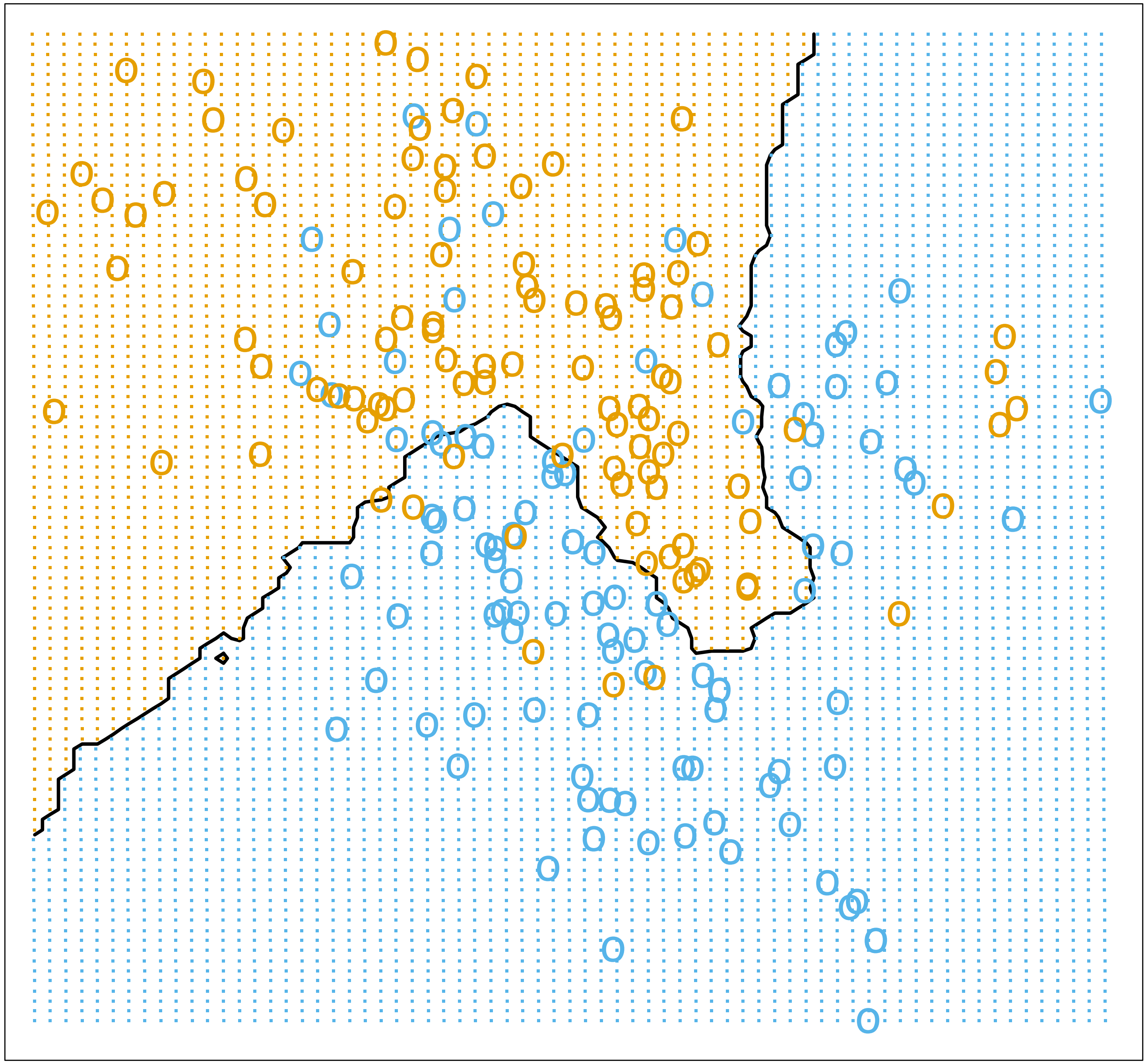
$$I_{\bar{x}}$$

the corresponding vector of indices, and

$$K_{\bar{x}} = \{I_{\bar{x}}^1, \dots, I_{\bar{x}}^K\}$$

be the array of the first $K$ entries of $I_{\bar{x}}$. The $K$-nearest neighbor estimator (KNN) is defined as,

$$\hat{f}(\bar{x}) = \sum_{i' \in K_{\bar{x}}} y_{i'},$$



Plot

# Remarks:

*Generalization I*: closer points should count more

$$\hat{f}(\bar{x}) = \frac{\sum_{i=1}^{n} y_i k(\bar{x}, x_i)}{\sum_{i=1}^{n} k(\bar{x}, x_i)}, \qquad \text{Gaussian} \quad k(x', x) = e^{-\|x-x'\|^2/2\sigma^2}.$$

**Parzen Windows**

*Generalization II*: other metric/similarities

$$X = \{0,1\}^D \qquad d_H(x, \bar{x}) = \frac{1}{D} \sum_{j=1}^{D} \mathbf{1}_{[x^j \neq \bar{x}^j]}$$

**There is one parameter controlling fit/stability**

**How do we choose it?**

**Is there an optimal value?**

**Can we compute it?**

# Is there an optimal value?

Ideally we would like to choose $K$ that minimizes the expected error

$$\mathbf{E}_S \mathbf{E}_{x,y} (y - \hat{f}_K(x))^2.$$

Next: Characterize corresponding minimization problem to uncover one of **the most fundamental aspect of machine learning.**

For the sake of simplicity we consider a regression model

$$y_i = f_*(x_i) + \delta_i, \quad \mathbf{E}\delta_I = 0, \mathbf{E}\delta_i^2 = \sigma^2 \quad i = 1, \dots, n$$

$$\mathbf{E}_S \mathbf{E}_{x,y}(y - \hat{f}_K(x))^2 = \mathbf{E}_x \underbrace{\mathbf{E}_S \mathbf{E}_{y|x}(y - \hat{f}_K(x))^2}_{\varepsilon(K)}.$$

$$\mathbf{E}_{y|x}\hat{f}_K(x) = \frac{1}{K}\sum_{\ell \in K_x} f_*(x_\ell).$$

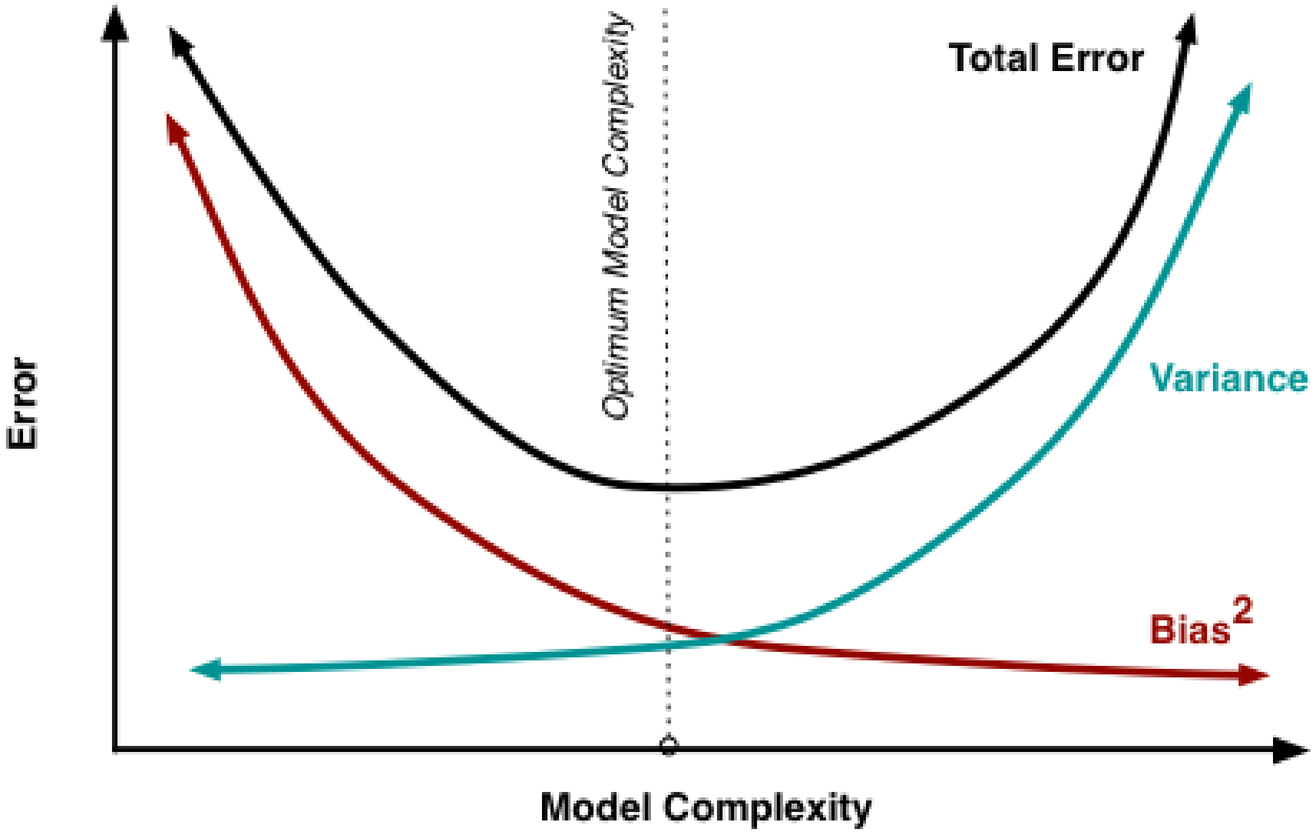
$$\mathbf{E}_S \mathbf{E}_{y|x}(f_*(x) - \hat{f}_K(x))^2 = \underbrace{(f_*(x) - \mathbf{E}_S \mathbf{E}_{y|x}\hat{f}_K(x))^2}_{Bias} + \underbrace{\mathbf{E}_S \mathbf{E}_{y|x}(\mathbf{E}_{y|x}\hat{f}_K(x) - \hat{f}_K(x))^2}_{Variance}$$

$$(f_*(x) + \frac{1}{K}\sum_{\ell \in K_x} f_*(x_\ell))^2$$

$$\frac{\sigma^2}{K}$$

# Bias Variance Trade-Off

$$\left(f_*(x) + \frac{1}{K}\sum_{\ell\in K_x} f_*(x_\ell)\right)^2 + \frac{\sigma^2}{K}$$



**Is there an optimal value? YES!** **Can we compute it?**

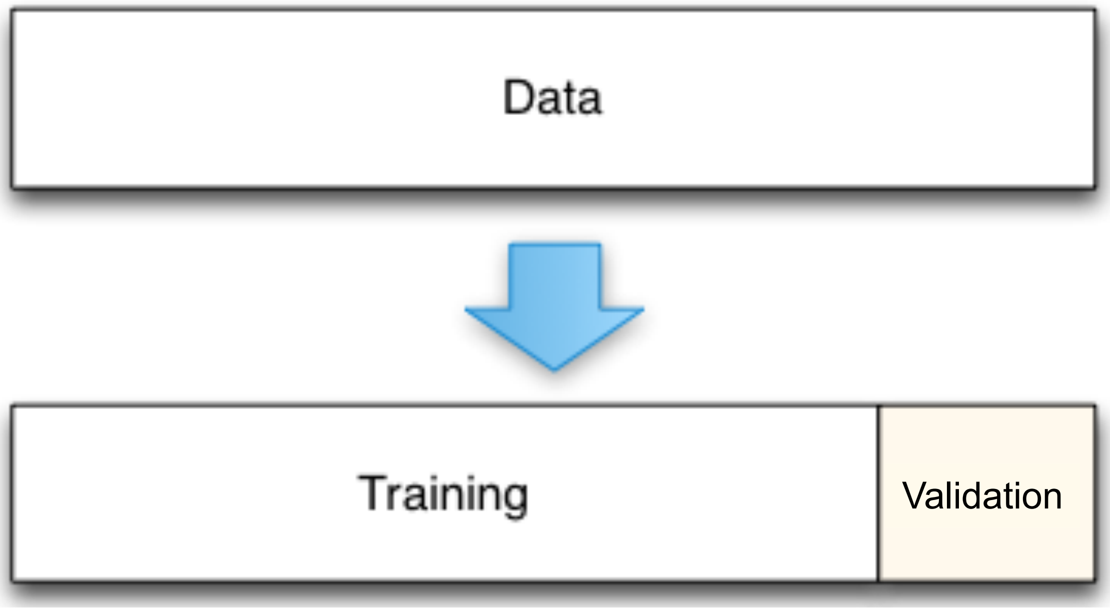
# Not quite…

$$(f_*(x) + \frac{1}{K}\sum_{\ell\in K_x} f_*(x_\ell))^2 + \frac{\sigma^2}{K}$$

## …enter Cross Validation

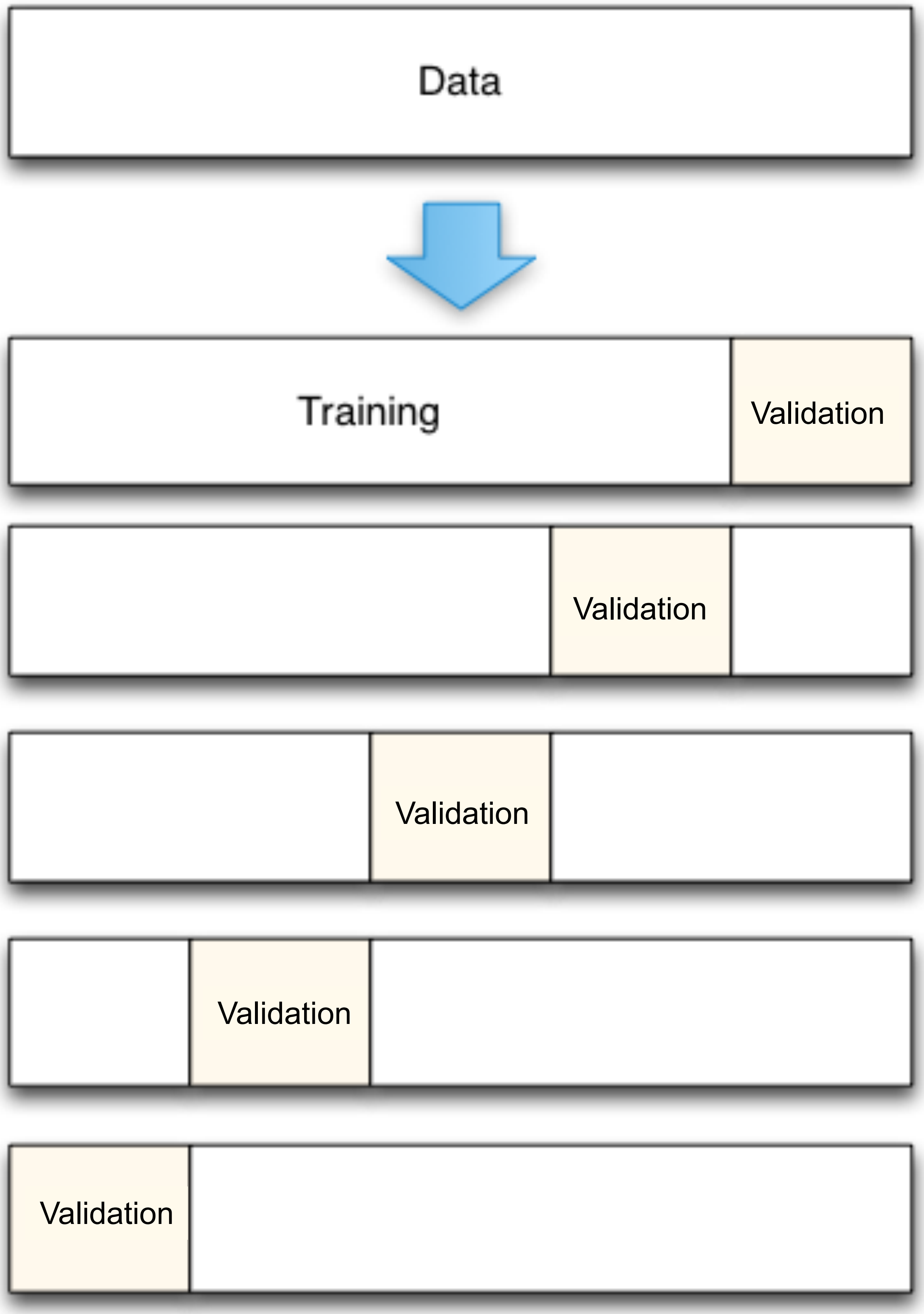
Split data: train on some, tune on some other

# Cross Validation Flavors



## Hold-Out

# Cross Validation Flavors



**V-Fold, (V=n is Leave-One-Out)**

# End of PART I

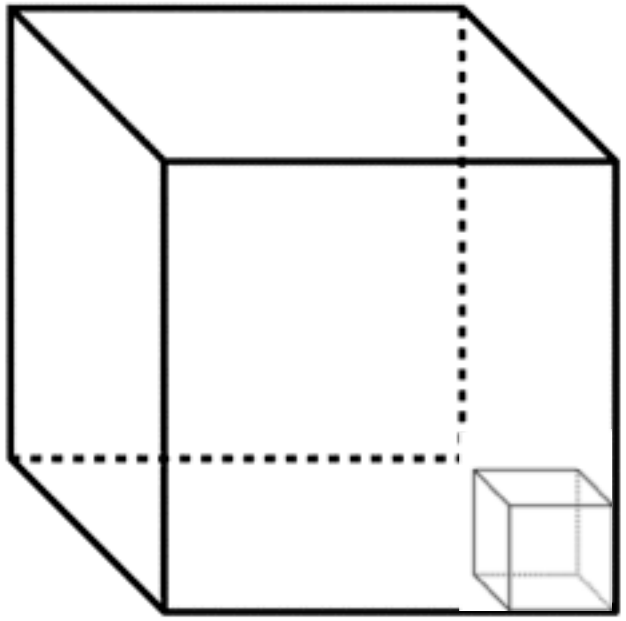
- Local methods
- Bias-Variance and Cross Validation

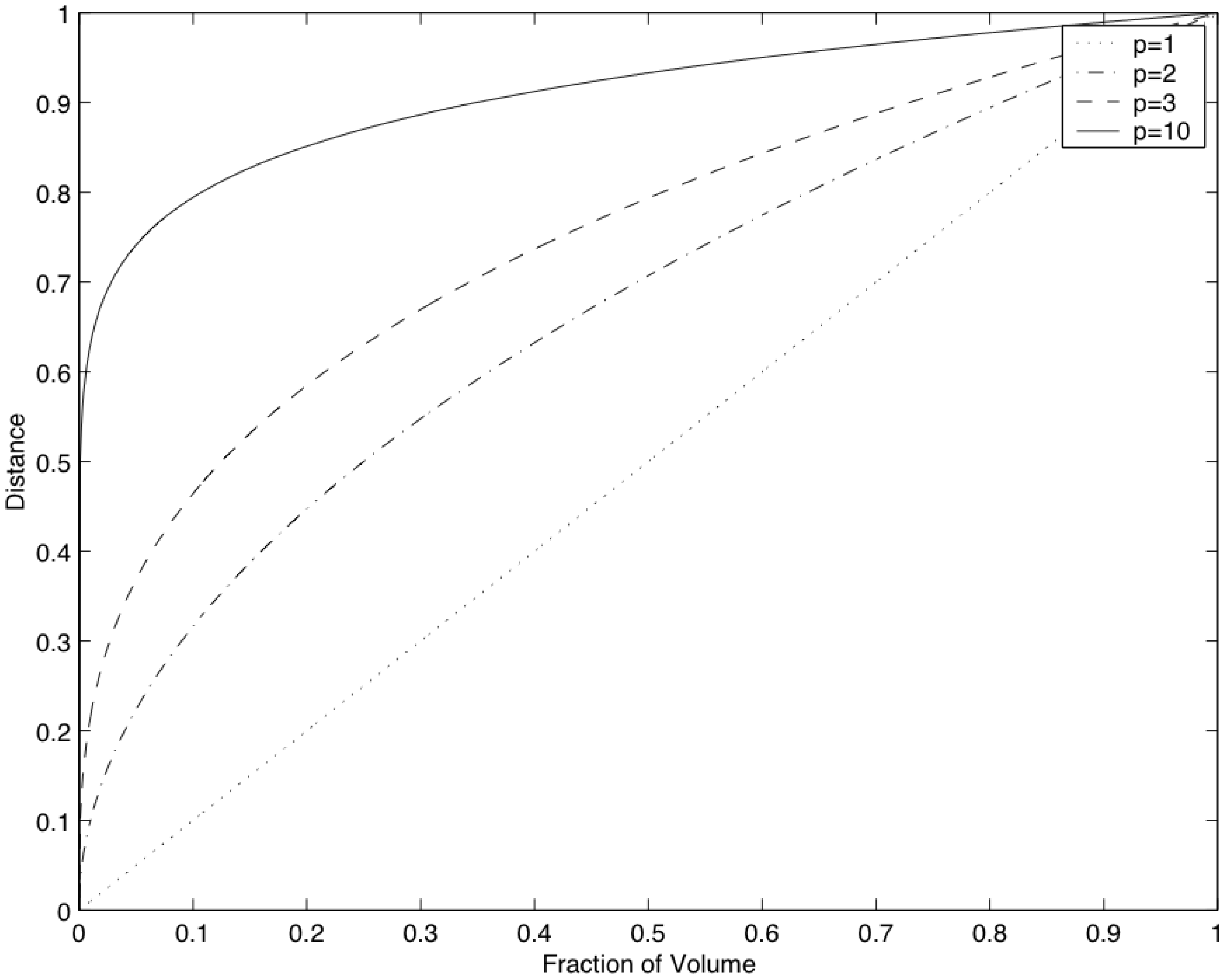
Stability - Overfitting - Bias/Variance - Cross-Validation

**End of the Story?**

# High Dimensions and Neighborhood

**tell me the length of the edge of a cube containing 1% of the volume of a cube with edge 1**





**Cubes and Dth-roots**

**Curse of dimensionality!**

# PART II

- Regularization I: Linear Least Squares
- Regularization II: Kernel Least Squares

**GOAL:** Introduce the basic (global) regularization methods with parametric and non parametric models

**Going Global + Impose Smoothness**

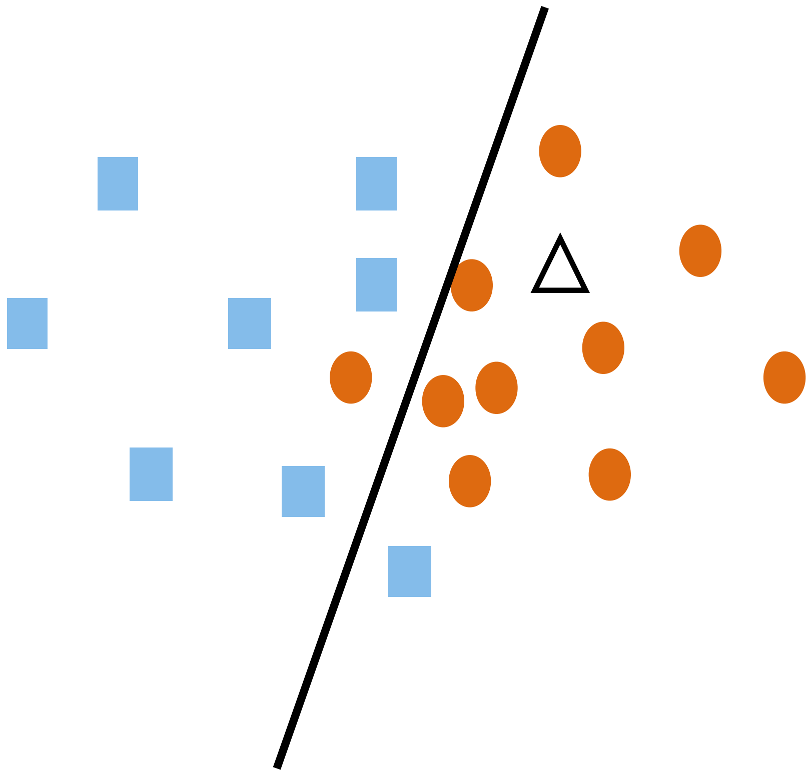
*Of all the principles which can be proposed for that purpose, I think there is none more general, more exact, and more easy of application, that of which we made use in the preceding researches, and which consists of rendering the* ***sum of squares of the errors*** *a minimum.*

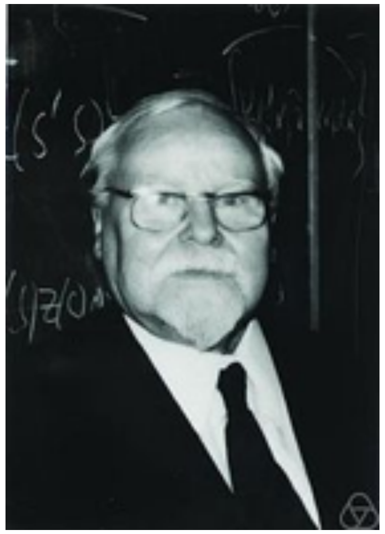
(Legendre 1805)

We consider the following algorithm

$$\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i))^2 + \lambda w^T w, \qquad \lambda \geq 0.$$

$$f(x) = w^T x = 0$$





Tikhonov '62

?

Phillips '62

?

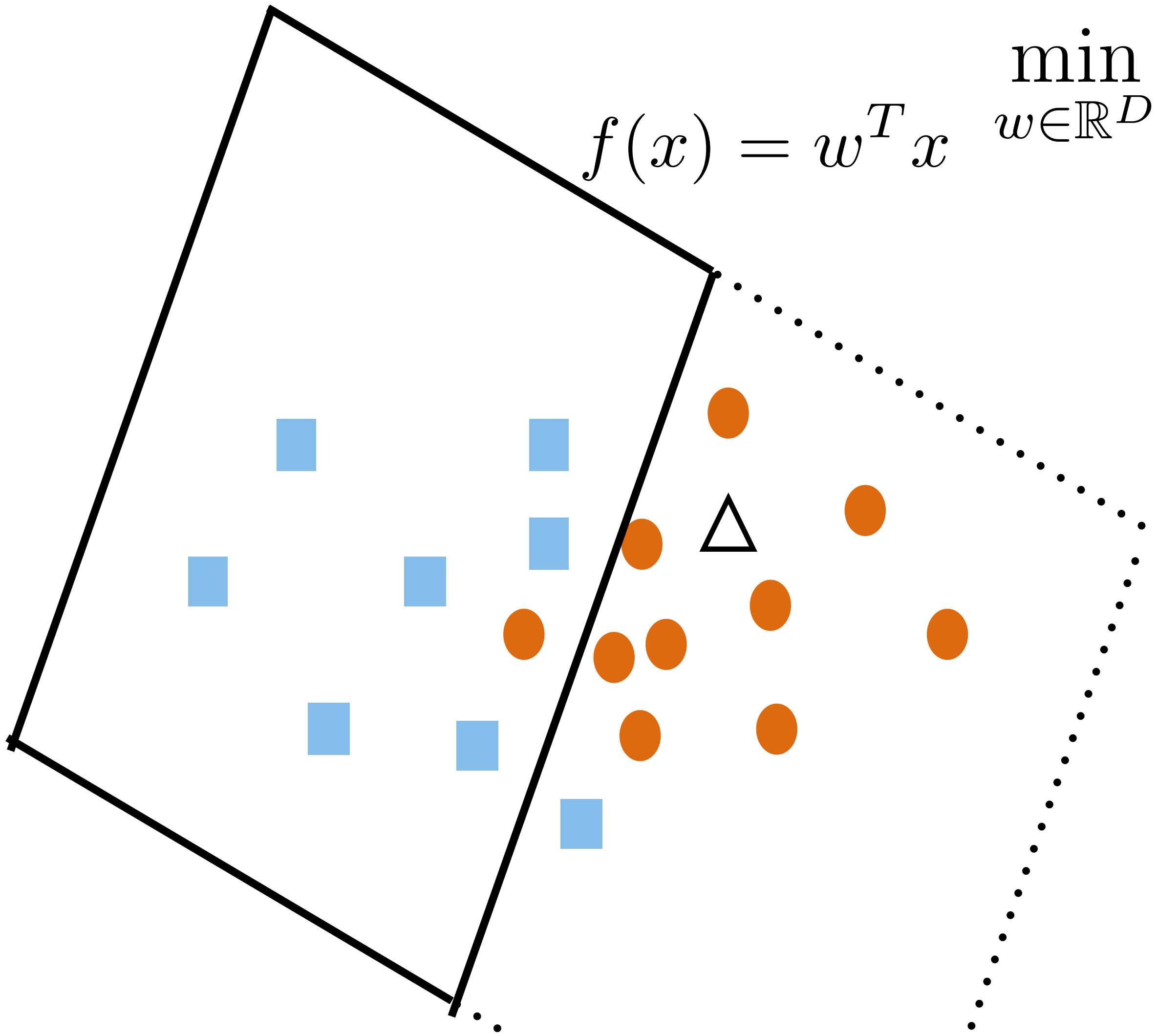
Hoerl et al. '62

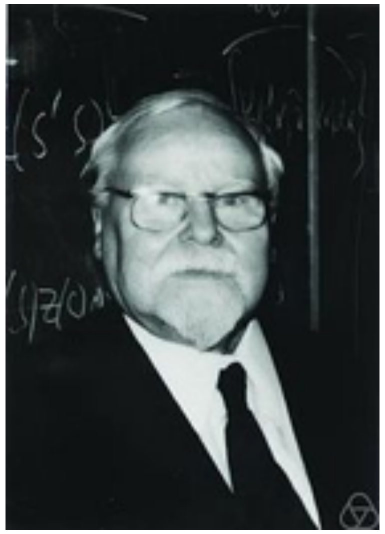
*Of all the principles which can be proposed for that purpose, I think there is none more general, more exact, and more easy of application, that of which we made use in the preceding researches, and which consists of rendering the* ***sum of squares of the errors*** *a minimum.*

(Legendre 1805)

We consider the following algorithm

$$\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i))^2 + \lambda w^T w, \quad \lambda \geq 0.$$





?

?

Tikhonov ‘62 Phillips ‘62 Hoerl et al. ‘62

$$\min_{w\in\mathbb{R}^D} \frac{1}{n}\sum_{i=1}^{n}(y_i - w^T x_i))^2 + \lambda w^T w, \quad \lambda \geq 0.$$

**Computations?**

**Statistics?**

$$\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i))^2 + \lambda w^T w, \quad \lambda \geq 0.$$

**Computations?**

**Notation** $\frac{1}{n} \sum_{i=1}^{n} (y_i - w^T x_i))^2 = \frac{1}{n} \|Y_n - X_n w\|^2$

$-\frac{2}{n} X_n^T (Y_n - X_n w), \quad \text{and,} \quad 2w$ **Setting gradients…**

**…to zero** $(X_n^T X_n + \lambda n I) w = X_n^T Y_n$

**OK, but what is this doing?**

# Interlude: Linear Systems

$$Ma = b,$$

- If $M$ is a diagonal $M = diag(\sigma_1, \dots, \sigma_D)$ where $\sigma_i \in (0, \infty)$ for all $i = 1, \dots, D$, then

$$M^{-1} = diag(1/\sigma_1, \dots, 1/\sigma_D), \quad (M + \lambda I)^{-1} = diag(1/(\sigma_1 + \lambda), \dots, 1/(\sigma_D + \lambda)$$

- If $M$ is symmetric and positive definite, then considering the eigendecomposition

$$M^{-1} = V \Sigma V^T, \quad \Sigma = diag(\sigma_1, \dots, \sigma_D),\ VV^T = I,$$

  then

$$M^{-1} = V \Sigma^{-1} V^T, \quad \Sigma^{-1} = diag(1/\sigma_1, \dots, 1/\sigma_D),$$

  and

$$(M + \lambda I)^{-1} = V \Sigma_\lambda = V^T, \quad \Sigma_\lambda = diag(1/(\sigma_1 + \lambda), \dots, 1/(\sigma_D + \lambda)$$

$$\min_{w\in\mathbb{R}^D} \frac{1}{n}\sum_{i=1}^{n}(y_i - w^T x_i))^2 + \lambda w^T w, \quad \lambda \geq 0.$$

**Statistics?**

$$(X_n^T X_n + \lambda n I)w = X_n^T Y_n$$

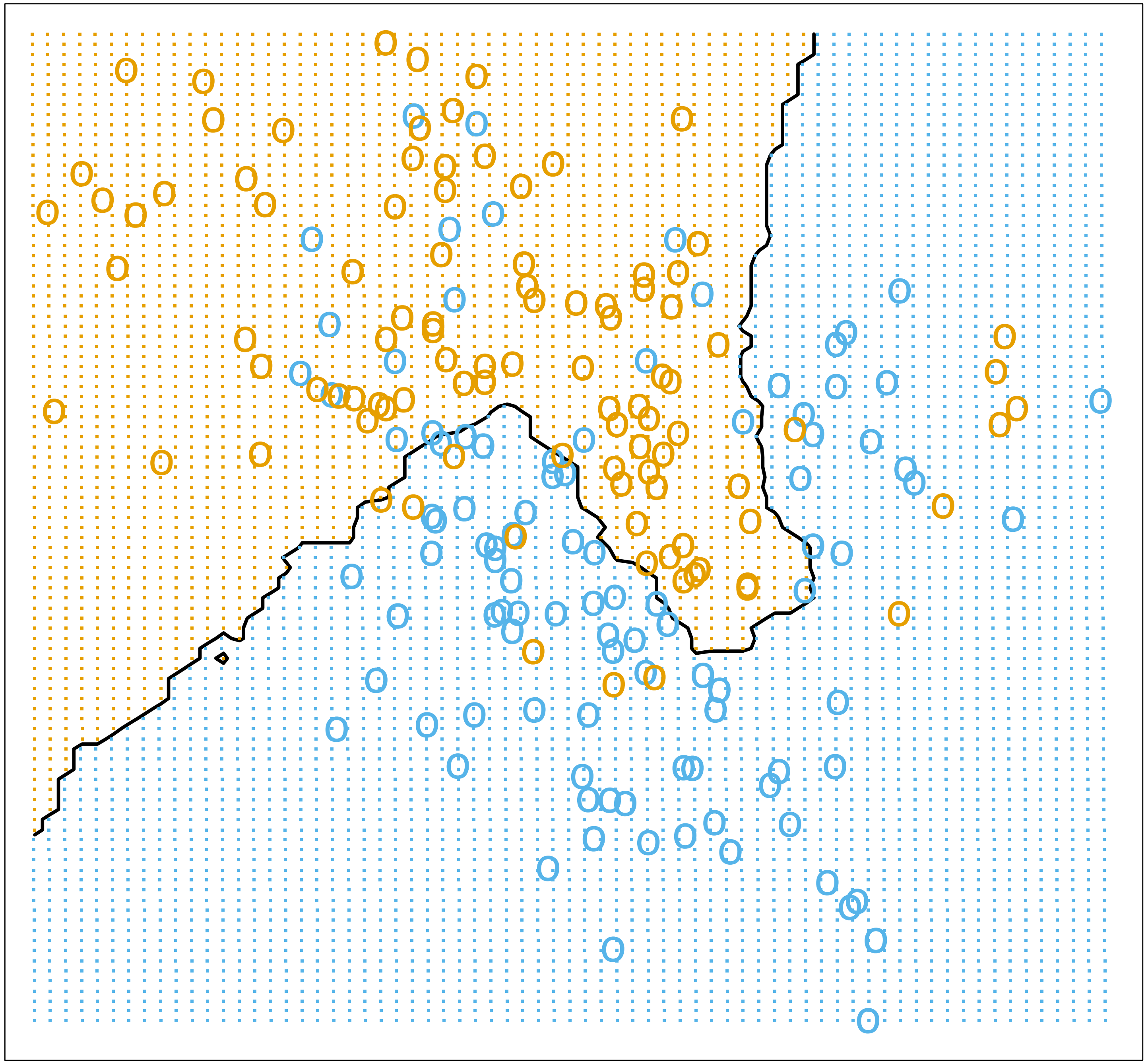
***another story that shall be told another time***

**(Stein ’56, James and Stein ’61)**

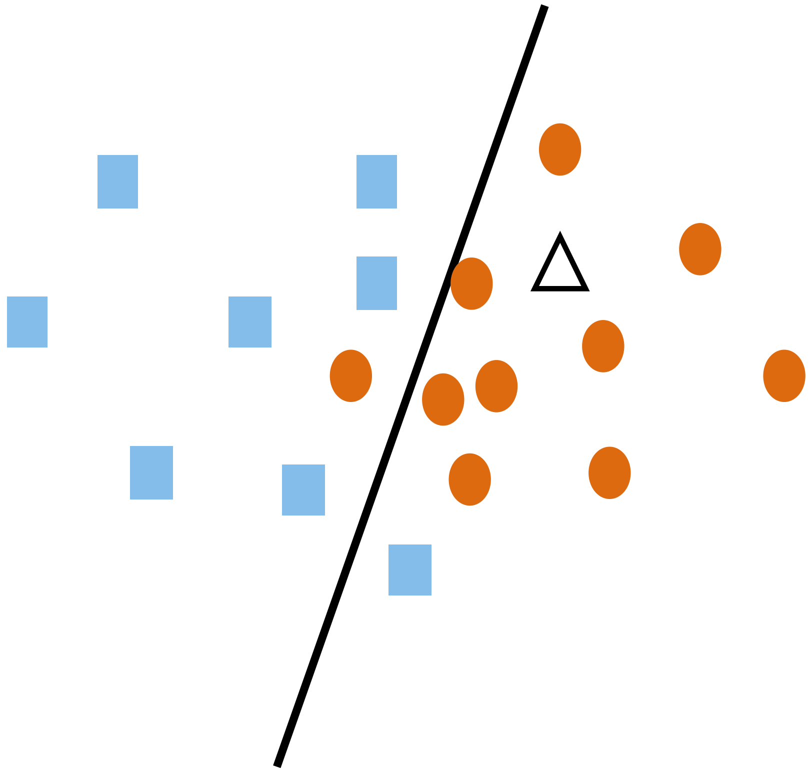
$$\min_{w\in\mathbb{R}^D} \frac{1}{n}\sum_{i=1}^{n}(y_i - w^T x_i))^2 + \lambda w^T w, \quad \lambda \geq 0.$$

$$f_w(x) = w^T x = \sum_{i=1}^{v} w^j x^j \qquad \sum_{j=1}^{D}(w^j)^2$$

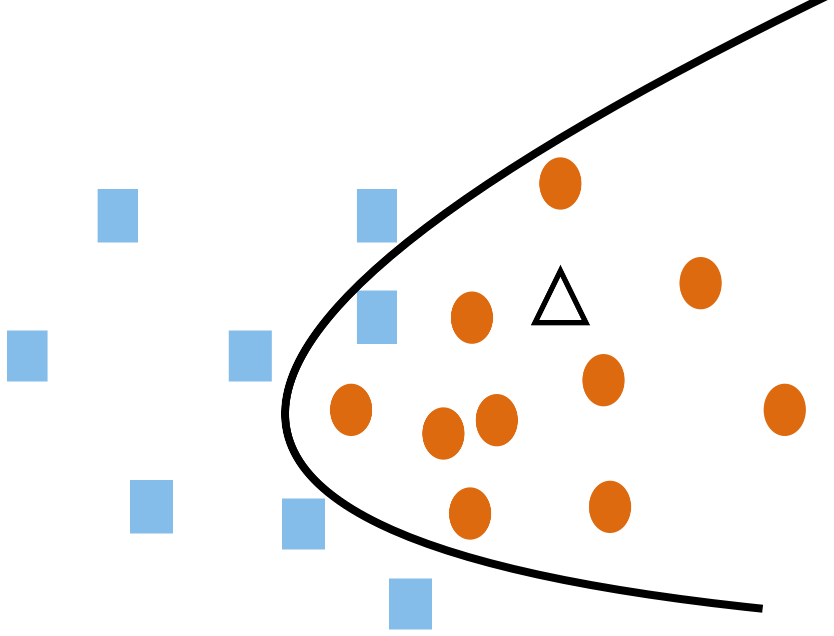
**Shrinkage - Stein Effect- Admissible Estimator**



**Plot**



# Why a linear decision rule?



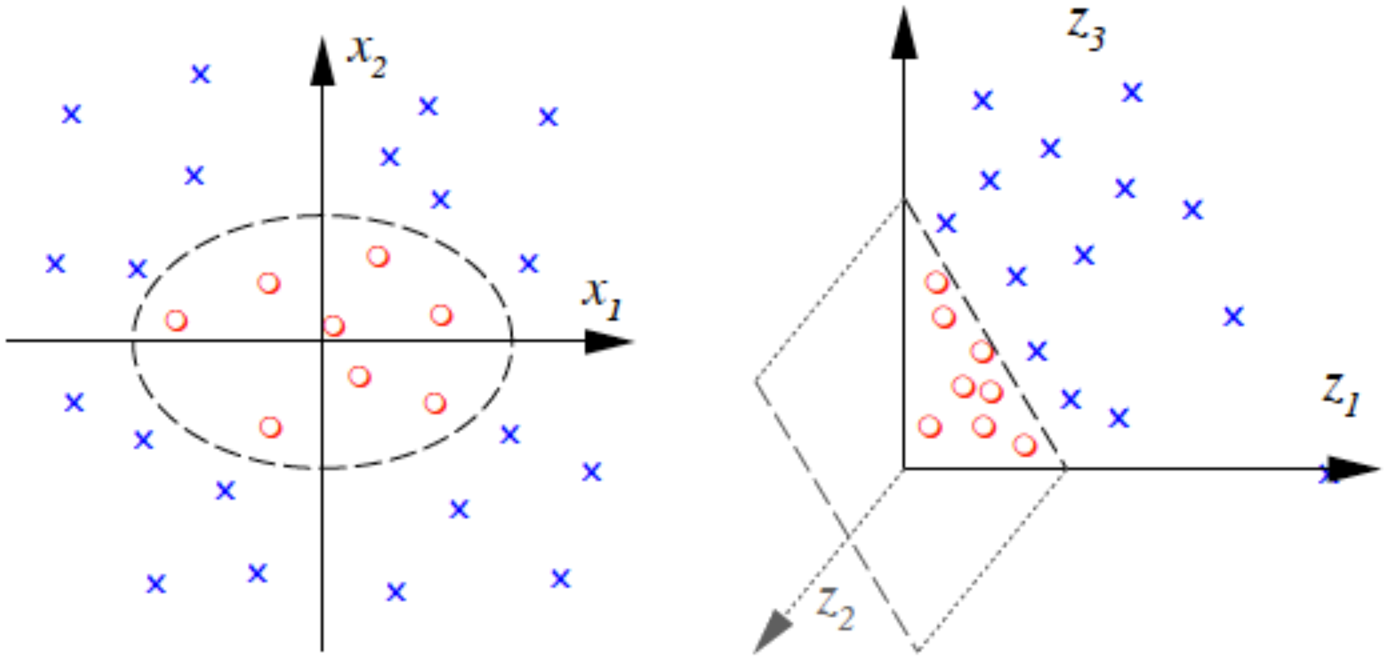
# Dictionaries

$$x \mapsto \tilde{x} = (\phi_1(x), \ldots, \phi_p(x)) \in \mathbb{R}^p$$

$$f(x) = w^T \tilde{x} = \sum_{j=1}^{p} \phi_j(x) w^j$$

$$\Phi : R^2 \to R^3$$

$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2} x_1 x_2, x_2^2)$$



$$(X_n^T X_n + \lambda n I)w = X_n^T Y_n \qquad \mapsto \qquad (\tilde{X}_n^T \tilde{X}_n + \lambda n Y)w = \tilde{X}_n^T Y_n$$

---

**What About Computational Complexity?**

# Complexity Vademecum

$M$ $n$ by $p$ matrix and $v, v'$ $p$ dimensional vectors

- $v^T v' \mapsto O(p)$
- $Mv' \mapsto O(np)$
- $MM^T \mapsto O(n^2 p)$
- $(MM^T)^{-1} \mapsto \ O(n^3)$

$$(X_n^T X_n + \lambda n I)w = X_n^T Y_n \qquad \mapsto \qquad (\tilde{X}_n^T \tilde{X}_n + \lambda n Y)w = \tilde{X}_n^T Y_n$$

## What About Computational Complexity?

$$O(p^3) + O(p^2 n)$$

## What if *p* is much larger than *n*?

$$(X_n^T X_n + \lambda n I)^{-1} X_n^T = X_n^T (X_n X_n^T + \lambda n I)^{-1}$$

$$w = X_n^T \underbrace{(X_n X_n^T + \lambda n I)^{-1} Y_n}_{c} = \sum_{i=1}^{n} x_i^T c_i$$

$$(X_n^T X_n + \lambda n I)^{-1} X_n^T = X_n^T (X_n X_n^T + \lambda n I)^{-1}$$

$$w = X_n^T \underbrace{(X_n X_n^T + \lambda n I)^{-1} Y_n}_{c} = \sum_{i=1}^{n} x_i^T c_i$$

**Computational Complexity:** $O(p^3) + O(p^2 n)$

$O(n^3) + O(pn^2)$

$$(X_n^T X_n + \lambda n I)^{-1} X_n^T = X_n^T (X_n X_n^T + \lambda n I)^{-1}$$

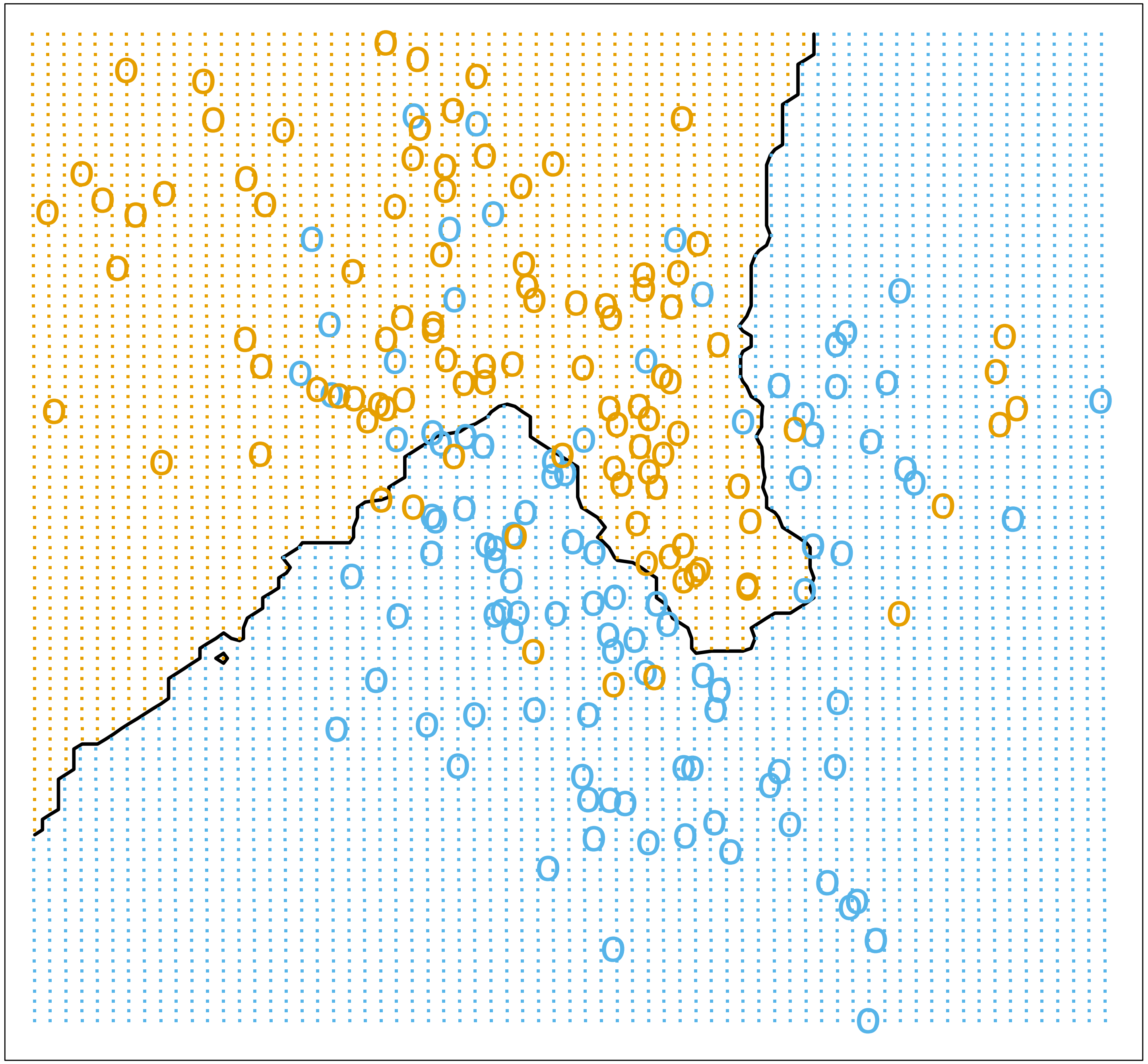
$$w = X_n^T \underbrace{(X_n X_n^T + \lambda n I)^{-1} Y_n}_{c} = \sum_{i=1}^{n} x_i^T c_i$$

**Kernels**

$$w = \sum_{j=1}^{n} x_i c_i \Rightarrow f(x) = x^T w = \sum_{j=1}^{n} \underbrace{x^T x_i}_{K(x,x_i)} c_i$$

$$(K_n + \lambda n I)^{-1} c = Y_n, \quad (K_n)_{i,j} = K(x_i, x_j)$$

- the linear kernel $K(x, x') = x^T x'$,
- the polynomial kernel $K(x, x') = (x^T x' + 1)^d$,
- the Gaussian kernel $K(x, x') = e^{-\frac{\|x - x'\|^2}{2\sigma^2}}$,



Plot

$$\hat{f}(x) = \sum_{i=1}^{n} K(x_i, x) c_i.$$

***things I won't tell you about***

- Reproducing Kernel Hilbert Spaces
- Gaussian Processes
- Integral Equations
- Sampling Theory/Inverse Problems

- Loss functions- SVM, Logistic…
- Multi - task, labels, outputs, classes

# End of PART II

- Regularization I: Linear Least Squares
- Regularization II: Kernel Least Squares

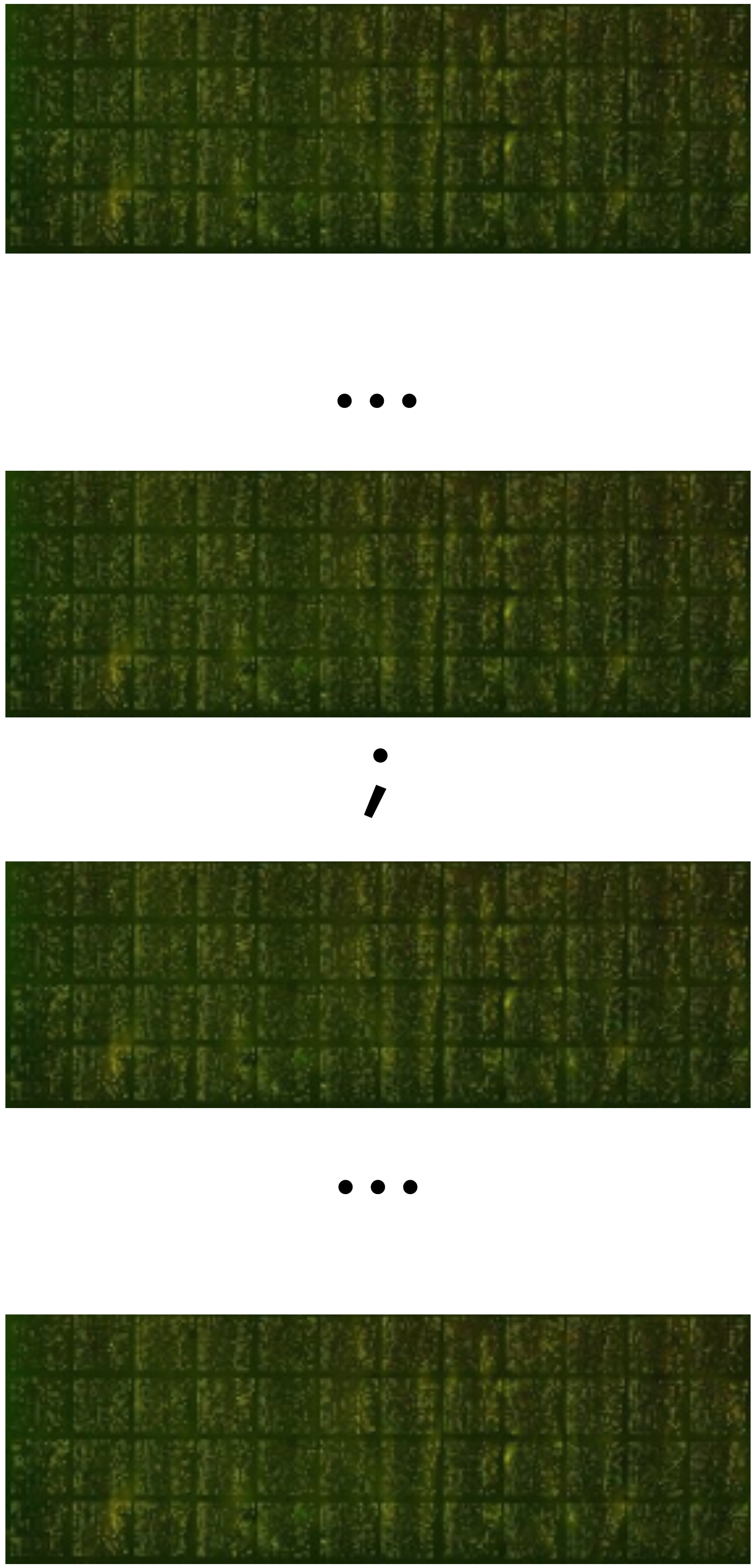
Regularized Least Squares - Dictionaries - Kernels

# PART III

- **a) Variable Selection: OMP**
- b) Dimensionality Reduction: PCA

**GOAL:** To introduce methods that allow to learn *interpretable* models from data

***n* patients *p* gene expression measurements**



$$X_n = \begin{pmatrix} x_1^1 & \dots & \dots & \dots & x_1^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^1 & \dots & \dots & \dots & x_n^p \end{pmatrix} ; Y_n = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

$$f_w(x) = w^T x = \sum_{j=1}^{D} x^j w^j$$

**Which variables are important for prediction?**

**or**

**Torture the data until they confess**

**Sparsity: only some of the coefficients are non zero**

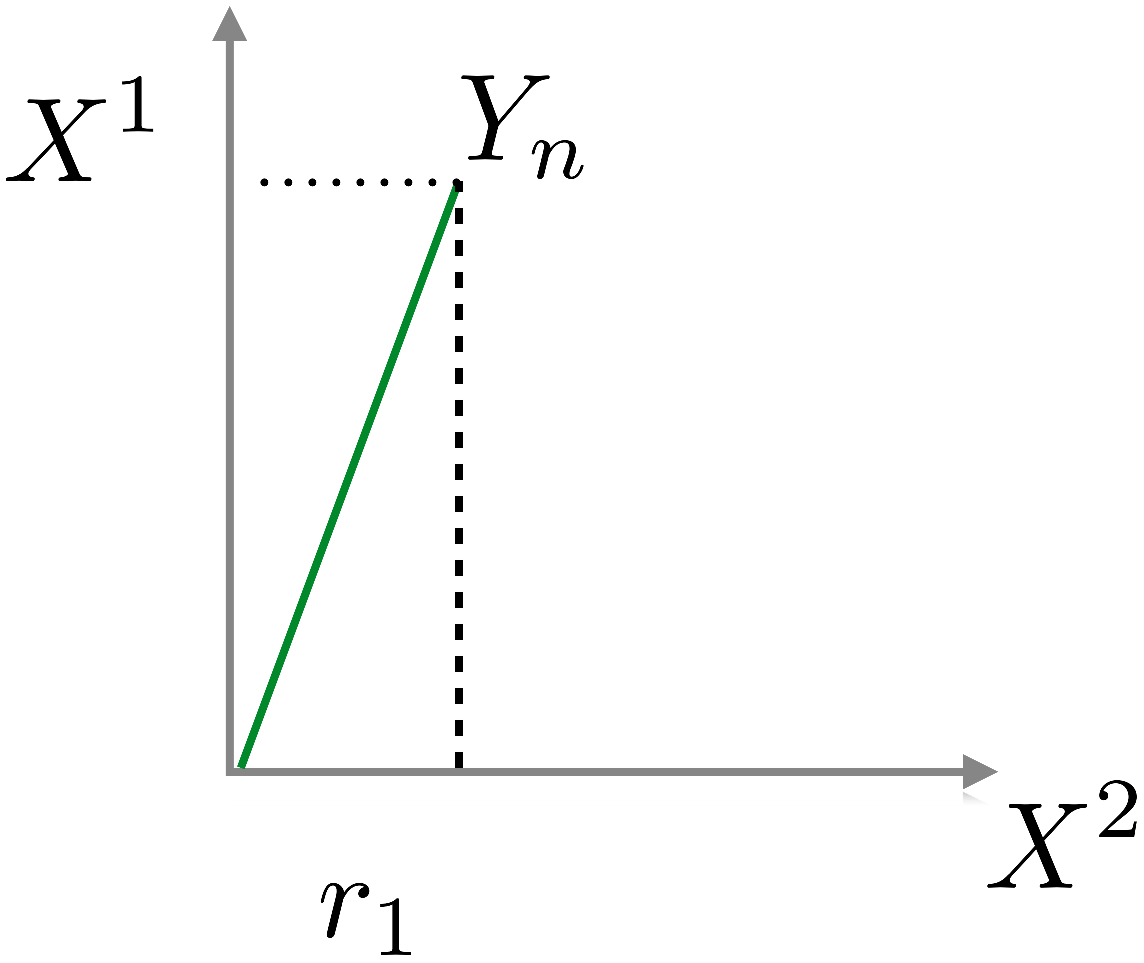
# Brute Force Approach

check all individual variables, then all couple, triplets…..

$$\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - f_w(x_i))^2 + \lambda \|w\|_0,$$

$$\|w\|_0 = |\{j \mid w^j \neq 0\}|$$

# Greedy approaches/Matching Pursuit



(1) initialize the residual, the coefficient vector, and the index set,
(2) find the variable most correlated with the residual,
(3) update the index set to include the index of such variable,
(4) update/compute coefficient vector,
(5) update residual.

# Matching Pursuit

(Mallat Zhang ’93)

```
$r_0 = Y_n, \quad , w_0 = 0, \quad I_0 = \emptyset.$

for $i \, = \, 1, \dots, T - 1$

    $k = \arg \max_{j=1,\dots,D} a_j, \quad a_j = \frac{(r_{i-1}^T X^j)^2}{\|X^j\|^2}, \circledast$

    $I_i = I_{i-1} \cup \{k\}$

    $w_i = w_{i-1} + w_k, \quad w_k k = v_k e_k$

    $r_i = r_{i-1} - X w^k.$

end
```

$$\circledast \; v^j = \frac{r_{i-1}^T X^j}{\|X^j\|^2} = \arg \min_{v \in \mathbb{R}} \|r_{i-1} - X^j v\|^2, \quad \|r_{i-1} - X^j v^j\|^2 = \|r_{i-1}\|^2 - a_j$$

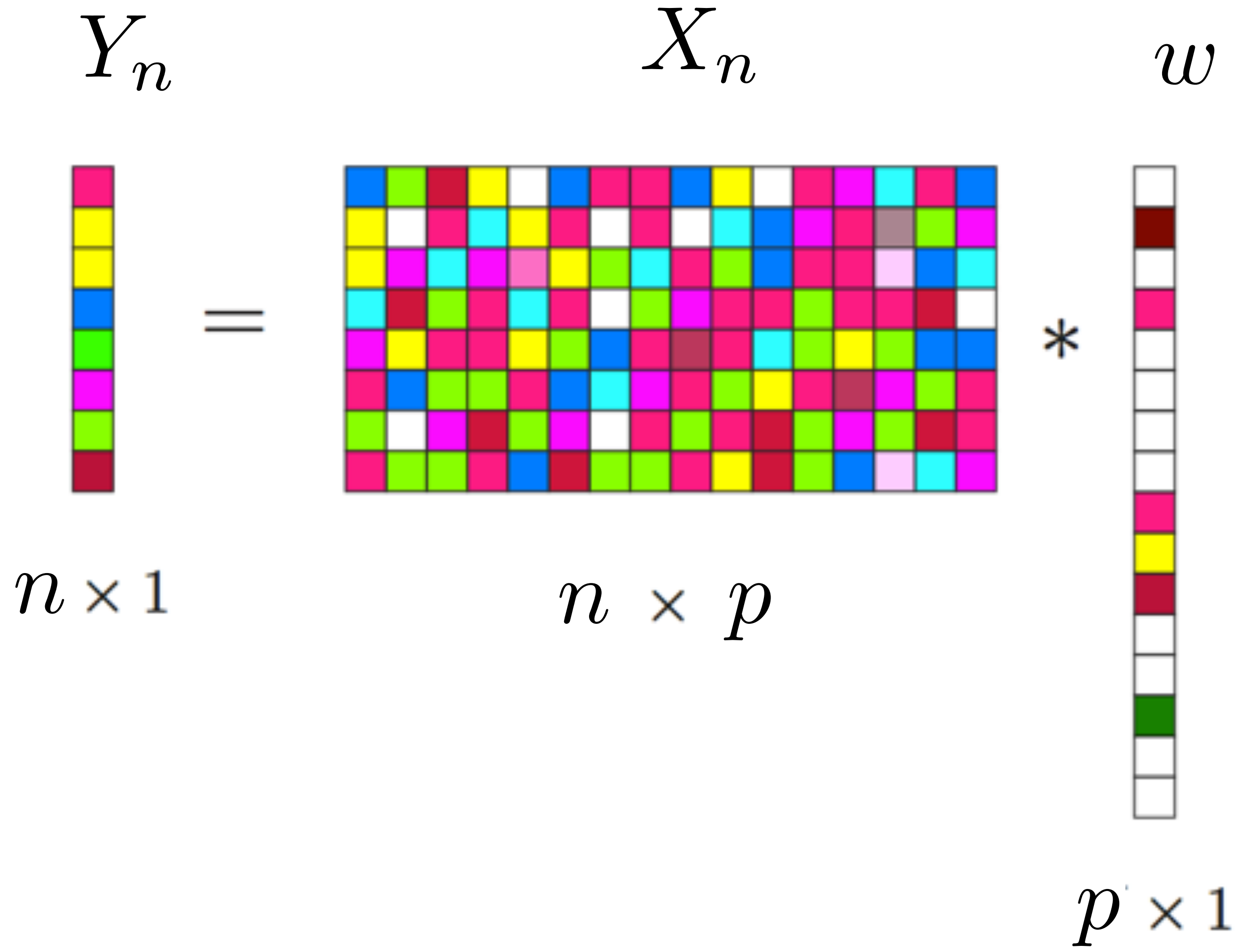
## Basis Pursuit/Lasso

(Chen Donoho Saunders ~95, Tibshirani '96)

$$\|w\|_1 = \sum_{j=1}^{D} |w^j|$$

$$\min_{w \in \mathbb{R}^D} \frac{1}{n} \sum_{i=1}^{n} (y_i - f_w(x_i))^2 + \lambda \cancel{\|w\|_0},$$

Problem is now **convex** and can be solved using convex optimization, in particular so called *proximal methods*



***things I won't tell you about***

- Solving underdetermined systems
- Sampling theory
- Compressed Sensing
- Structured Sparsity
- From vector to matrices- from sparsity to low rank

# End of PART III a)

- **a) Variable Selection: OMP**
- b) Dimensionality Reduction: PCA

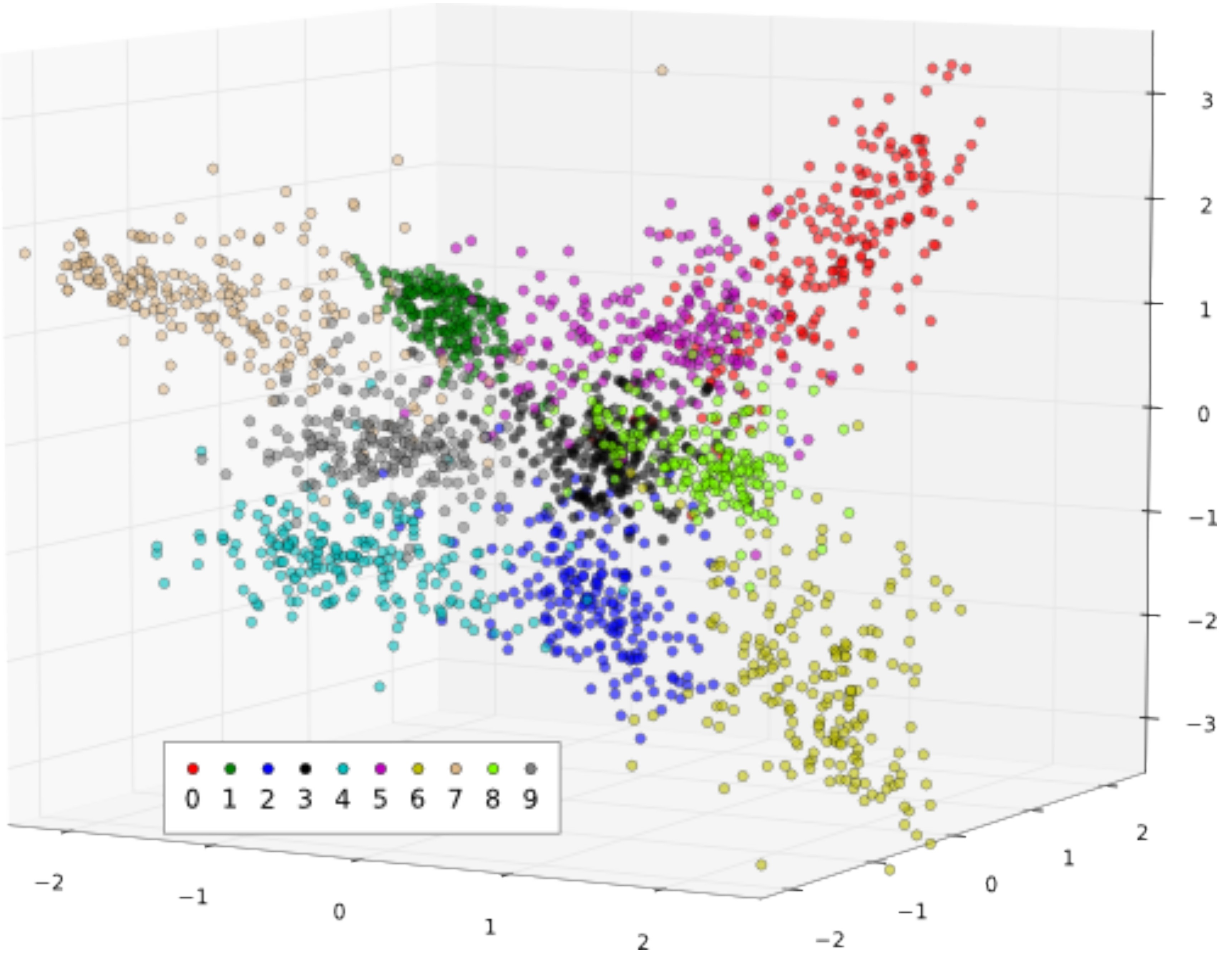
Interpretability - Sparsity - Greedy & Convex Relaxation Approaches

# PART III b)

- a) Variable Selection: OMP
- **b) Dimensionality Reduction: PCA**

**GOAL:** To introduce methods that allow to reduce data dimensionality in absence of labels, namely **unsupervised learning**

# Dimensionality Reduction for Data Visualization



# Dimensionality Reduction

$$M : X = \mathbb{R}^D \to \mathbb{R}^k, \quad k \ll D,$$

# Dimensionality Reduction

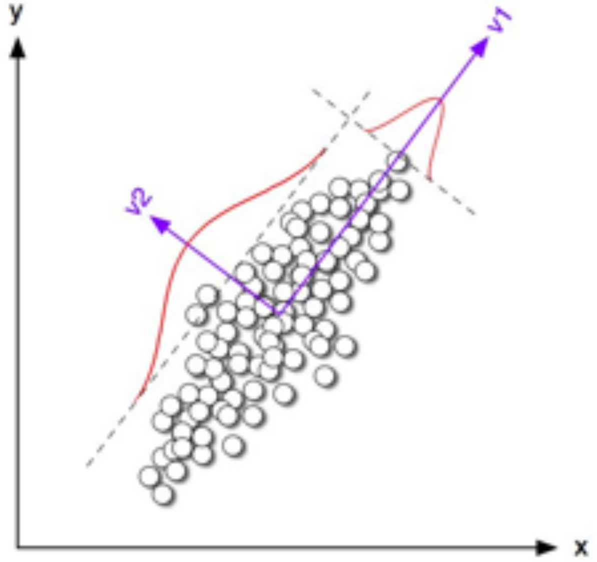
$$M : X = \mathbb{R}^D \to \mathbb{R}^k, \quad k \ll D,$$

Consider first $k = 1$

# Dimensionality Reduction

$$M : X = \mathbb{R}^D \to \mathbb{R}^k, \quad k \ll D,$$

Consider first $k = 1$



**PCA** $$\min_{w \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \|x_i - (w^T x_i) w\|^2,$$

$$w^T w = 1$$

**Computations?** **Statistics?**

$$\min_{w\in\mathbb{S}^{D-1}} \frac{1}{n}\sum_{i=1}^{n} \|x_i - (w^T x_i)w\|^2,$$

**Statistics?**

$$\min_{w \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \|x_i - (w^T x_i) w\|^2,$$

**Statistics?**

$$\|x_i - (w^T x_i) w\|^2 = \|x_i\| - (w^T x_i)^2,$$

$$\min_{w\in\mathbb{S}^{D-1}} \frac{1}{n}\sum_{i=1}^{n} \|x_i - (w^T x_i)w\|^2,$$

**Statistics?**

$$\|x_i - (w^T x_i)w\|^2 = \|x_i\| - (w^T x_i)^2,$$

$$\implies \max_{w\in\mathbb{S}^{D-1}} \frac{1}{n}\sum_{i=1}^{n}(w^T x_i)^2.$$

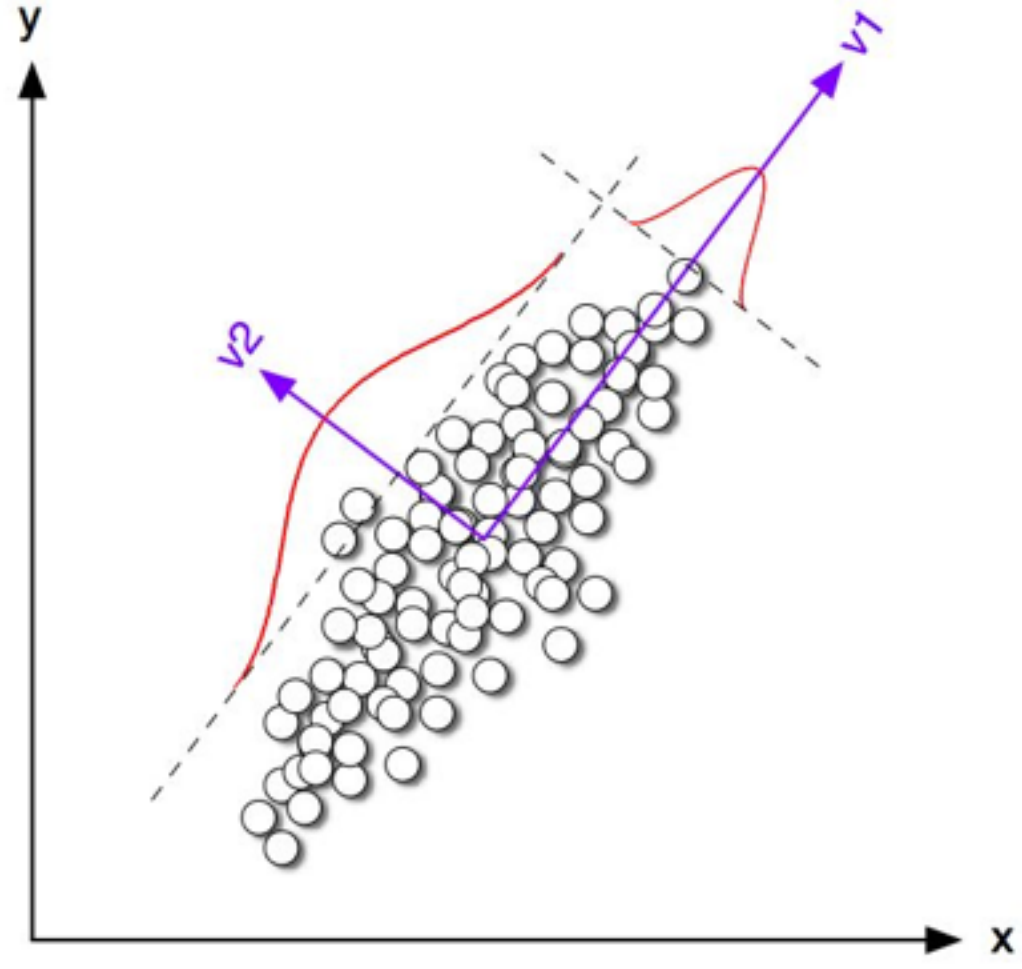
$$\min_{w\in\mathbb{S}^{D-1}} \frac{1}{n}\sum_{i=1}^{n} \|x_i-(w^Tx_i)w\|^2,$$

**Statistics?**

$$\|x_i-(w^Tx_i)w\|^2 = \|x_i\| - (w^Tx_i)^2;$$

$$\Longrightarrow \quad \max_{w\in\mathbb{S}^{D-1}} \frac{1}{n}\sum_{i=1}^{n}(w^Tx_i)^2.$$

$$\Longrightarrow \quad \max_{w\in\mathbb{S}^{D-1}} \frac{1}{n}\sum_{i=1}^{n}(w^T(x_i-\bar{x}))^2;$$



$$\min_{w \in \mathbb{S}^{D-1}} \frac{1}{n} \sum_{i=1}^{n} \|x_i - (w^T x_i) w\|^2,$$

**Computations?**

$$\min_{w\in\mathbb{S}^{D-1}} \frac{1}{n}\sum_{i=1}^{n}\|x_i-(w^Tx_i)w\|^2,$$

**Computations?**

$w_1$ **max eigenvector of** $C_n$

$$\max_{w\in\mathbb{S}^{D-1}} \frac{1}{n}\sum_{i=1}^{n}(w^Tx_i)^2. \quad \Leftrightarrow \quad \max_{w\in\mathbb{S}^{D-1}} w^TC_nw, \quad C_n=\frac{1}{n}\sum_{i=1}^{n}x_ix_i^T$$

$$\frac{1}{n}\sum_{i=1}^{n}(w^Tx_i)^2=\frac{1}{n}\sum_{i=1}^{n}w^Tx_iw^Tx_i=\frac{1}{n}\sum_{i=1}^{n}w^Tx_ix_i^Tw=w^T(\frac{1}{n}\sum_{i=1}^{n}x_ix_i^T)w$$

# Dimensionality Reduction

$$M : X = \mathbb{R}^D \to \mathbb{R}^k, \quad k \ll D,$$

What about $k = 2$?

$$\dots$$

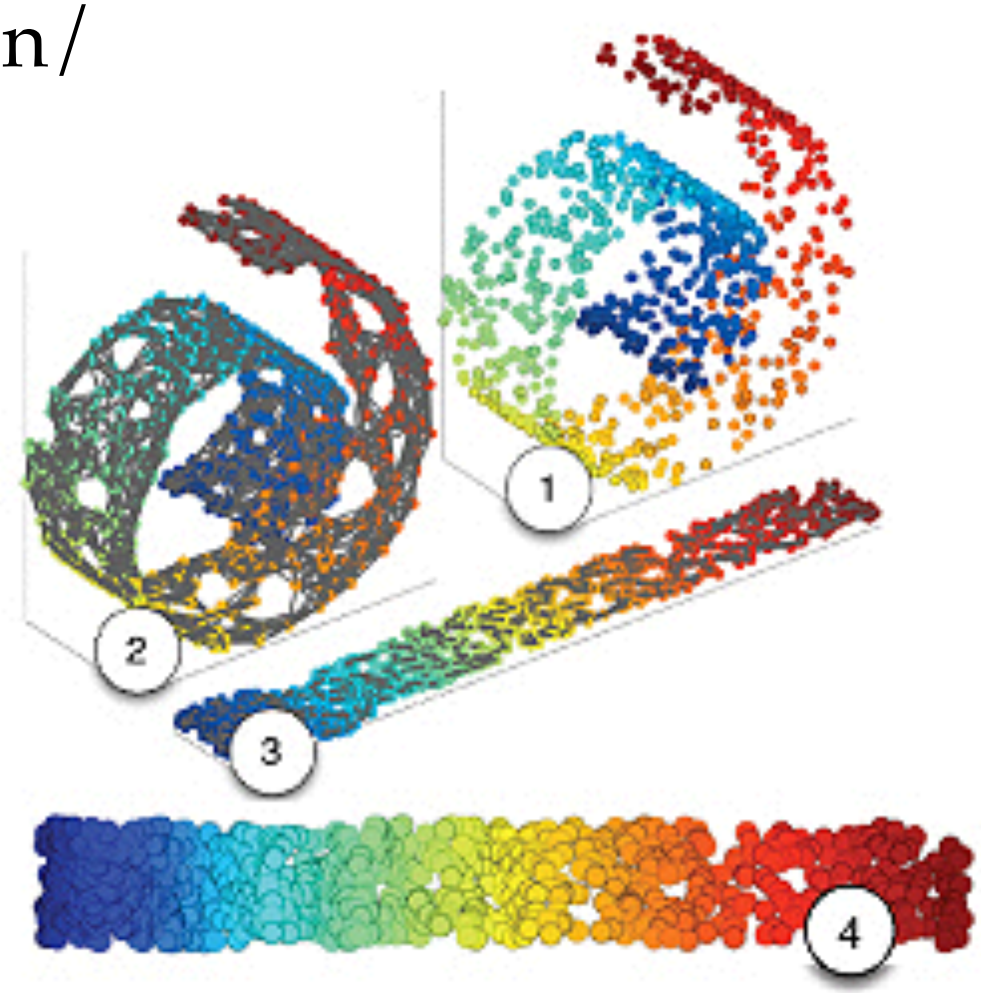
$w_2$ second eigenvector of $C_n$

$$\max_{\substack{w \in \mathbb{S}^{D-1} \\ w \perp w_1}} w^T C_n w, \quad C_n = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^T.$$

$$M : X = \mathbb{R}^D \to \mathbb{R}^k, \quad k \ll D,$$

***things I won't tell you about***

- **Random** Maps: Johnson-Linderstrauss Lemma
- **Non Linear** Maps: Kernel PCA, Laplacian/ Diffusion maps



# End of PART III b)

- a) Variable Selection: OMP
- **b) Dimensionality Reduction: PCA**

Interpretability - Sparsity - Greedy & Convex Relaxation Approaches

# The End

## PART IV

- Matlab practical session

Afternoon