S1133 2019

Introduction to Machine Learning for Bioinformatics

Chloé-Agathe Azencott
Centre for Computational Biology, Mines ParisTech
chloe-agathe.azencott@mines-paristech.fr

Inserm
Institut national de la santé et de la recherche médicale

institutCurie

PSL
RESEARCH UNIVERSITY PARIS

MINES ParisTech

Overview

- What kind of **problems** can machine learning solve, particularly in bioinformatics?
- Some popular **supervised** ML **algorithms:**
  - **Linear** models
  - **Support vector machines**
  - **Random forests**
  - **Neural networks**
- How do we **select** a machine learning algorithm?
- What is **overfitting**, and how can we avoid it?
- How do we deal with **high-dimensional** data?

What is (Machine) Learning?

Why Learn?

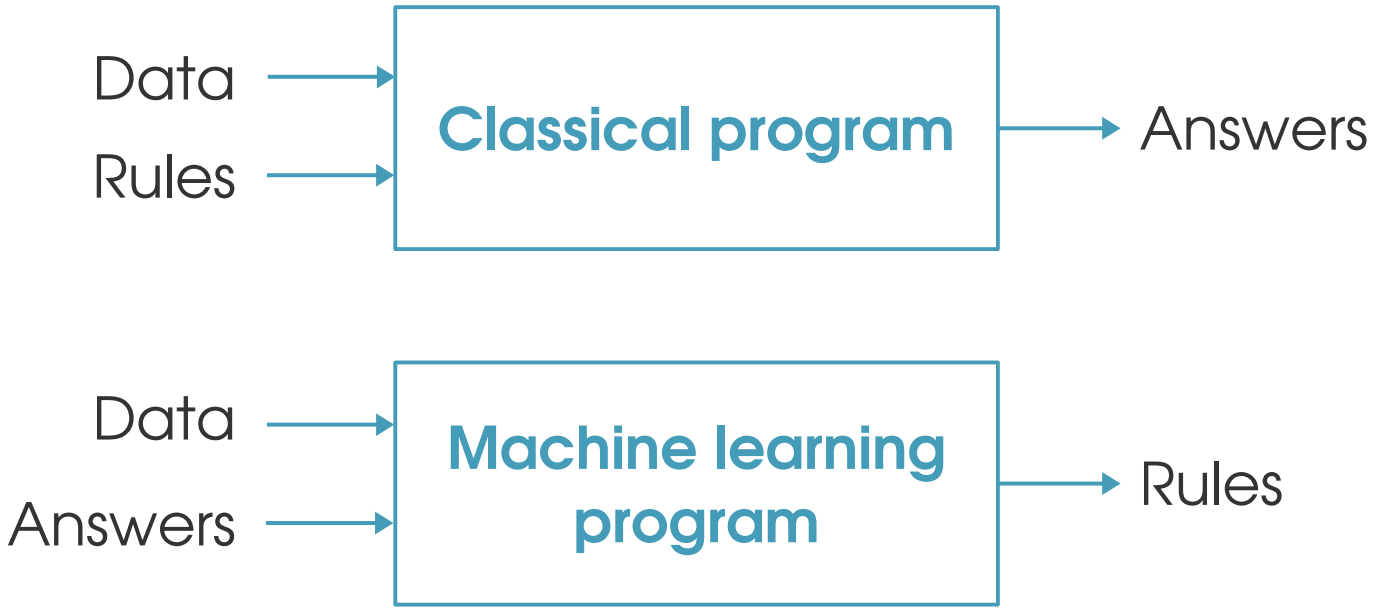
- **Learning**:

  Aquire a **skill** from **experience, practice.**

- **Machine learning**: Programming computers to
  - **model phenomena**
  - by means of **optimizing** an **objective function**
  - using **example** data.

Why Learn?

- There is no need to "learn" to calculate payroll.
- Learning is used when
  - Human expertise does not exist (bioinformatics);
  - Humans are unable to explain their expertise (speech recognition, computer vision).



What about AI?

Artificial Intelligence

- **Weak AI:** a software or machine focused on **one narrow task**

  E.g. Siri (Apple), DeepBlue (IBM), AlphaGo (Alphabet), Alexa (Amazon).

- **Strong AI:** a machine
  - able to address **any problem**
  - possessing **consciousness**, **sentience**, a **mind** of its own
  - **always active** and interacting with the world

  E.g. Hal (Clarke), Skynet (Cameron), Rachael (Dick), Samantha (Jonze).

Artificial Intelligence

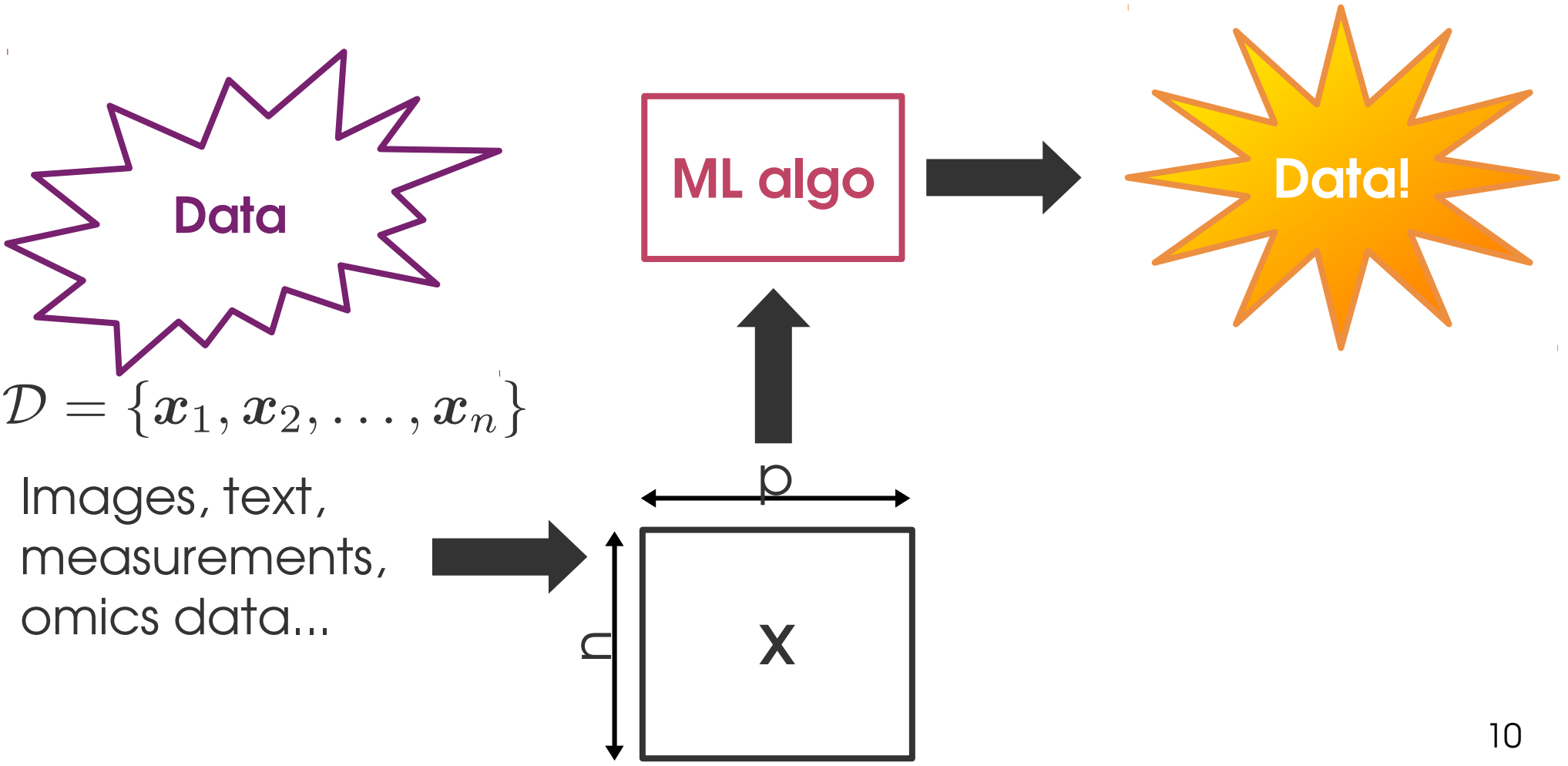
ML is a **component** of AI.

So are robotics, data bases, language processing...

Zoo of ML Problems

Unsupervised learning

Learn a **new representation** of the data



Clustering

Group **similar** data points together

$\mathcal{D} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$

- Understand **general characteristics** of the data;
- **Infer some properties** of an object based on how it relates to other objects.

Clustering: applications

- **Customer segmentation**

  Find groups of customers with similar buying behavior.

- **Topic modeling**

  Find groups of documents with similar content (and thus, hopefully, topics).

- **Gene expression clustering**

- **Disease subtyping**

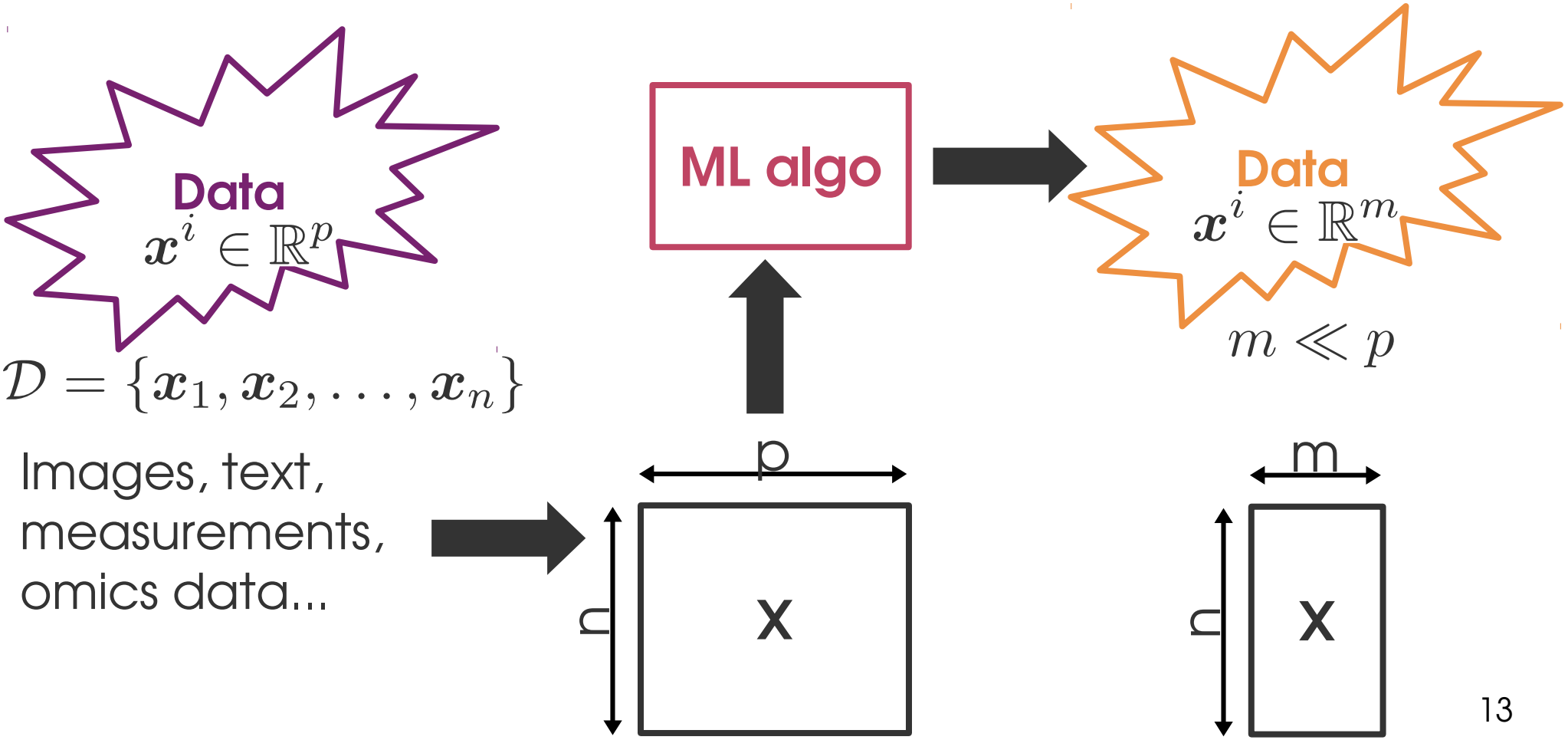
  Find groups of patients with similar molecular basis for the disease / similar symptomes.

- **Tumor cell clustering**

  Untangle tumor heterogeneity.

Dimensionality reduction

Find a **lower-dimensional** representation



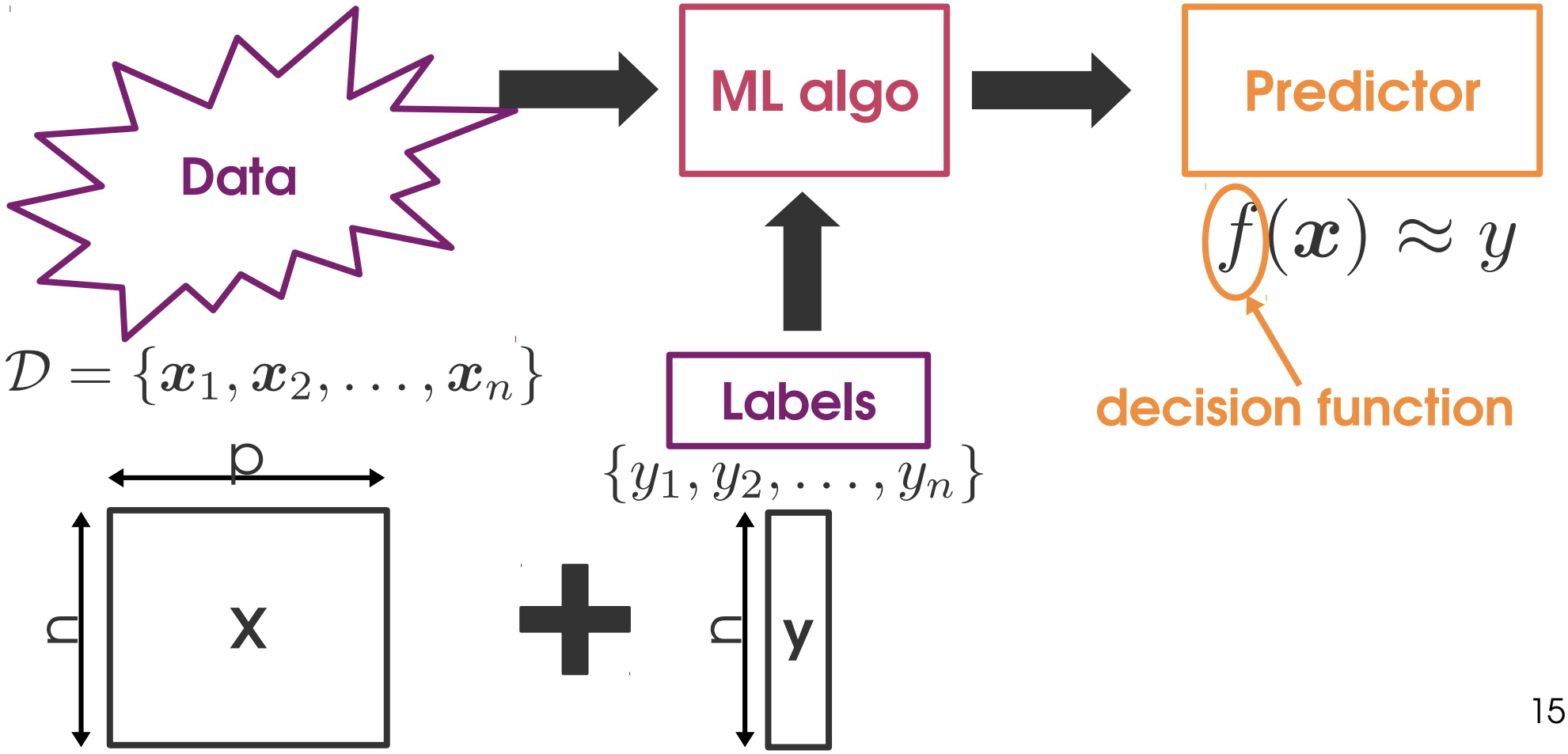
Dimensionality reduction

Find a **lower-dimensional** representation

- Reduce storage **space** & computational **time**
- Remove **redundances**
- **Visualization** (in 2 or 3 dimensions) and **interpretability**.

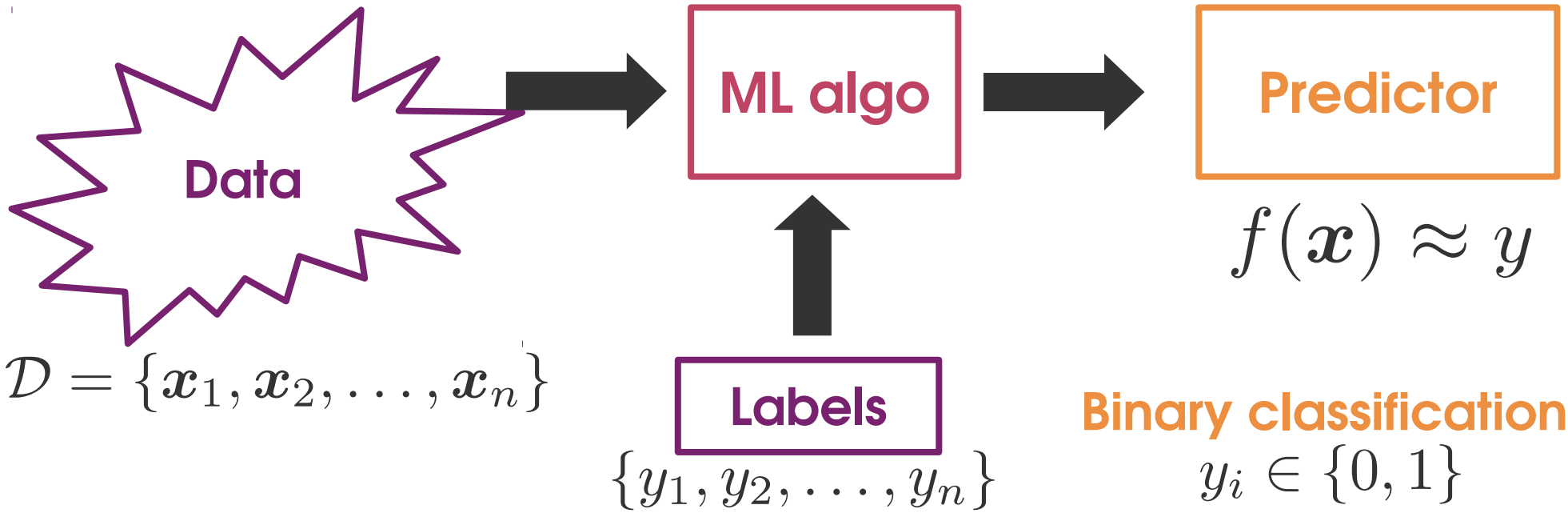
Supervised learning

Make **predictions**



Classification

Make **discrete** predictions



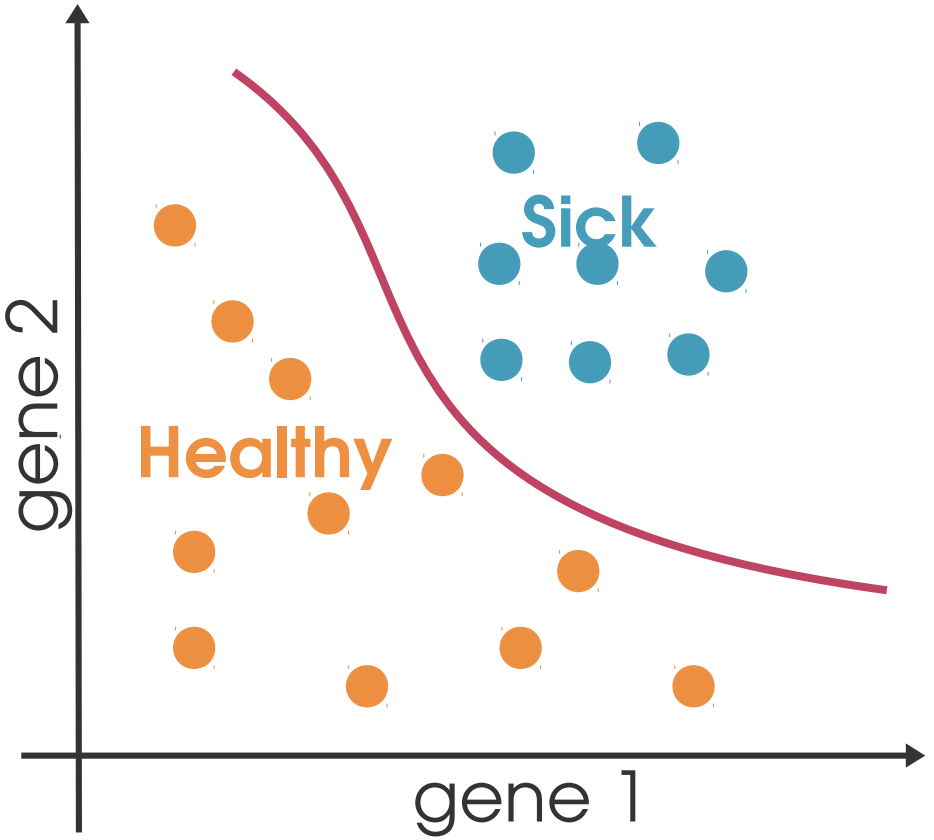
Binary classification

$y_i \in \{0, 1\}$

Multi-class classification

$y_i \in \{0, 1, \ldots, k\}$

Classification



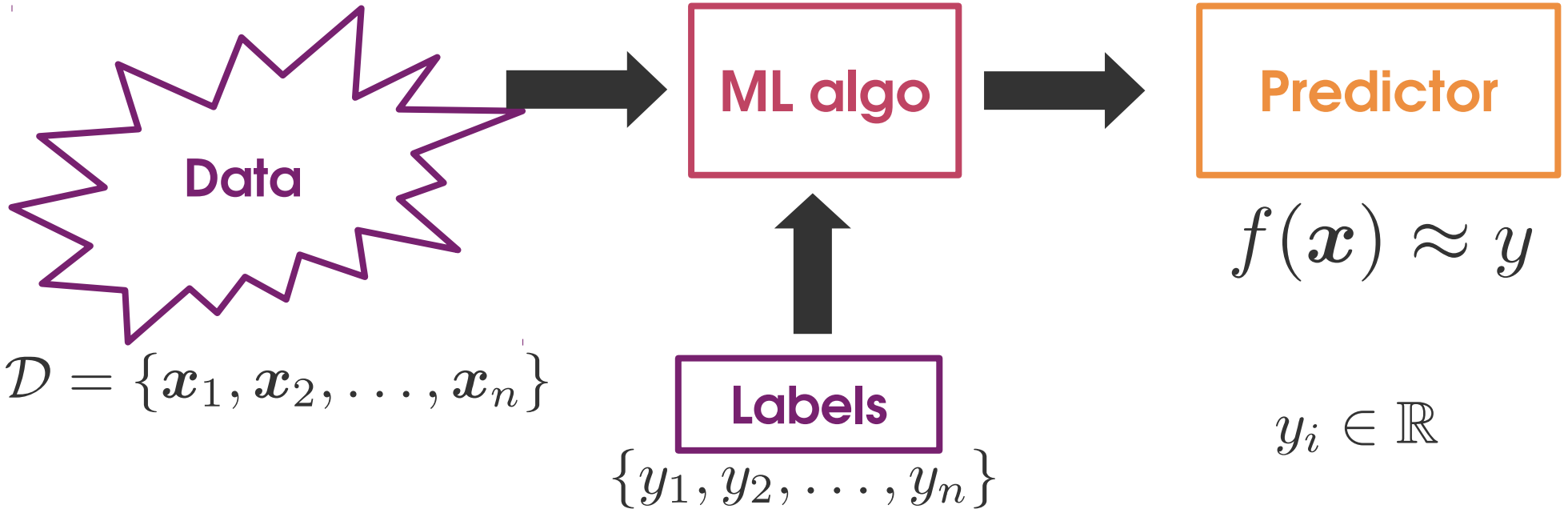
Classification: Applications

- **Face recognition**
- **Handwritten character recognition**

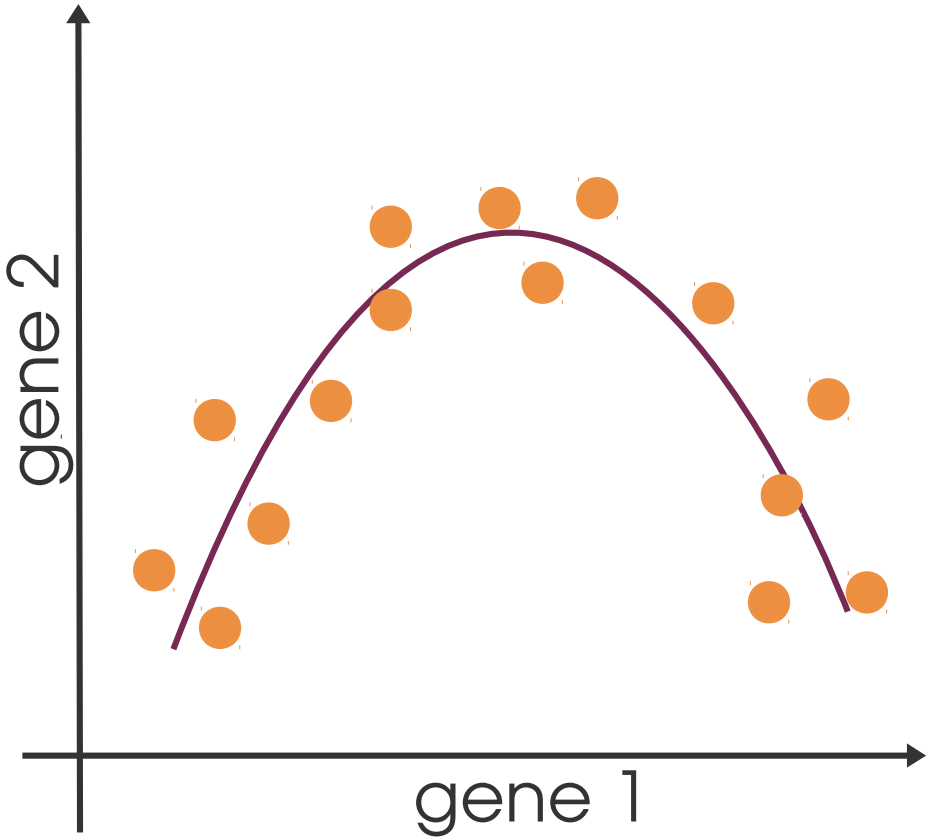
- What is the **function of this gene?**
- Is this **DNA sequence** a **micro-RNA?**
- Does this blood sample come from a **diseased or a healthy individual?**
- Is this **drug** appropriate for this patient?
- What **side effects** could this new drug have?

Regression

Make **continuous** predictions



Regression



Regression: Applications

- **Click prediction**

  How many people will click on this ad? Comment on this post? Share this article on social media?

- **Load prediction**

  How many users will my service have at a given time?

- **Gene expression regulation:** how much of this gene will be expressed?
- When will this **patient relapse?**
- **Drug efficacy:** how well does this drug work on this tumor?
- What is the **binding affinity** between these two molecules?
- How **soluble** is this chemical in water?

Parametric models

- Decision function has a **set form**
- **Model complexity** ≈ number of parameters

$$f(x) = \alpha_1 x_1 + \alpha_2 (x_1 x_2)^{\beta} + \alpha_3 \log(x_3)$$

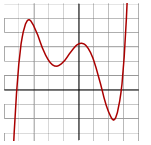
Non-parametric models

- Decision function can have **"arbitrary" form**
- **Model complexity** grows with the number of samples.

$$f(x) = \frac{1}{K} \sum_{i : x_i \in \mathcal{N}_K(x)} y_i$$

Training a supervised model

- **Ingredients:**
  - **Data**
    $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$
  - **Hypotheses class**
    Type of decision function f
  - **Cost function**
    Error of f on a sample
- **Recipe: empirical risk minimzation (ERM)**
  Find, among all functions of the hypotheses class, one that minimizes the average error of f on all samples.

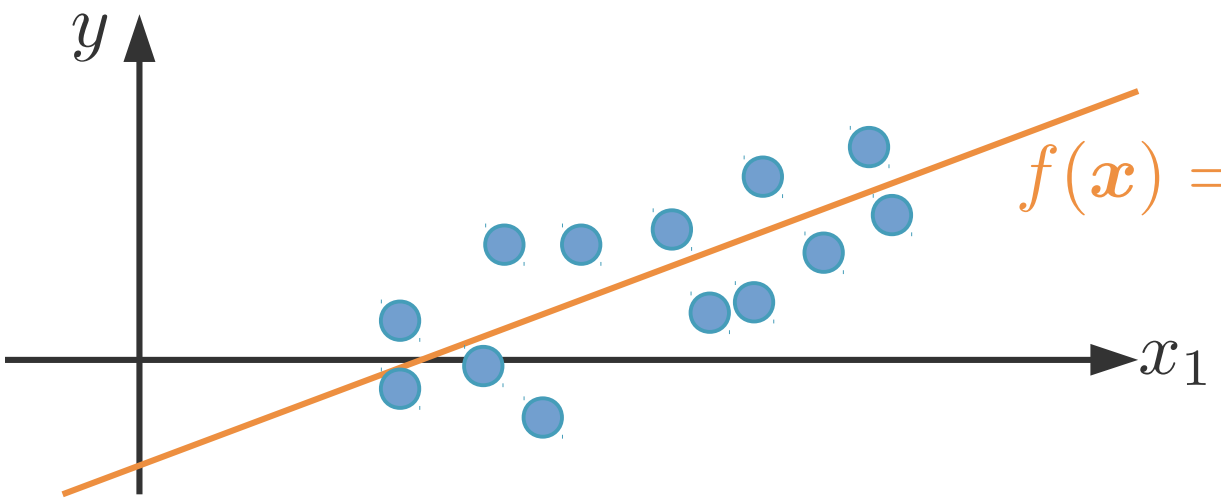


Linear models

Linear regression

$\boldsymbol{x} \in \mathbb{R}^p, y \in \mathbb{R}$

$\mathcal{D} = \{\boldsymbol{x}^i, y^i\}_{i=1,\ldots,n}$

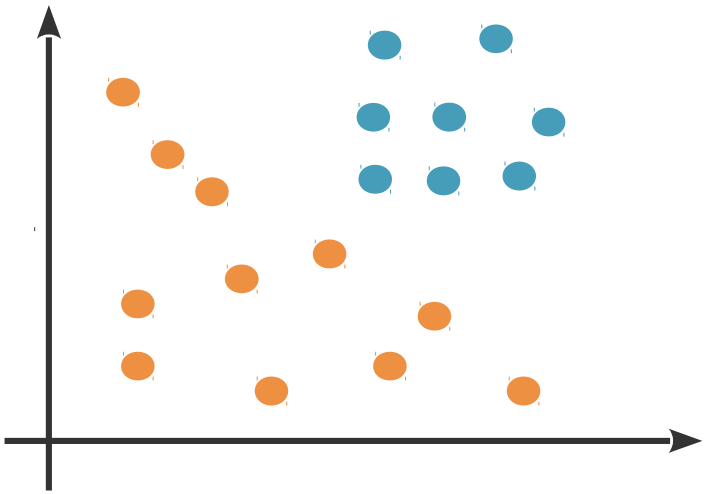


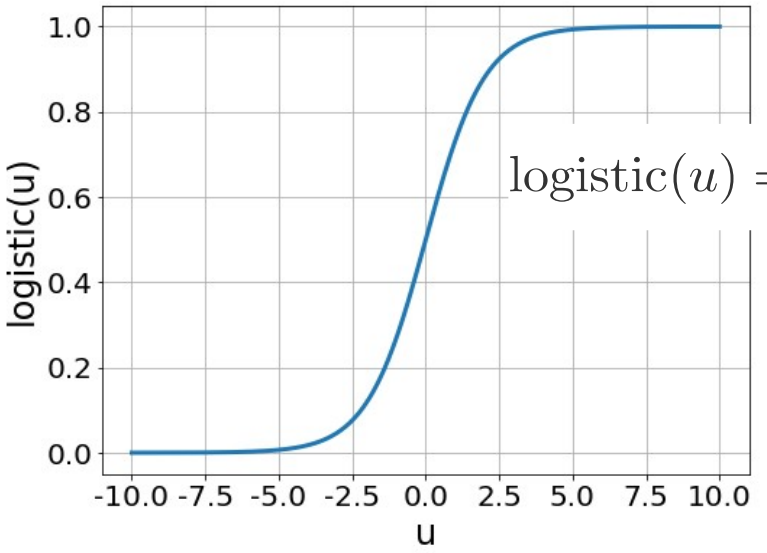
$$f(\boldsymbol{x}) = \sum_{j=1}^{p} \beta_j x_j + \beta_0$$

- **Least-squares fit:** $\boldsymbol{\beta} = \arg\min ||X\boldsymbol{\beta} - \boldsymbol{y}||_2^2$
- Equivalent to **maximizing the likelihood** $p(\mathcal{D}|\boldsymbol{\beta})$ under the assumption of Gaussian noise
- **Exact solution** $\hat{\boldsymbol{\beta}} = (X^\top X)^{-1} X^\top \boldsymbol{y}$
  if X has full column rank

Classification: logistic regression

Linear function → probability





$$\text{logistic}(u) = \frac{1}{1+e^{-u}}$$

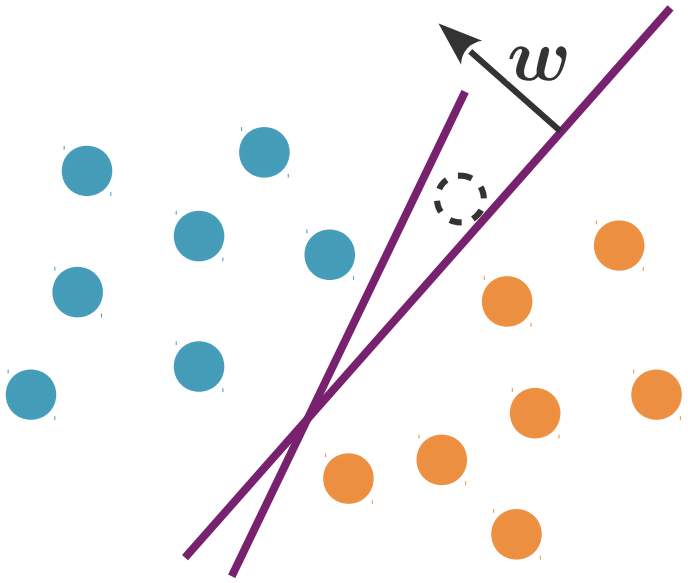
$$p(y=1|\boldsymbol{x}) = \text{logistic}(\sum_{j=1} \beta_j x_j + \beta_0)$$

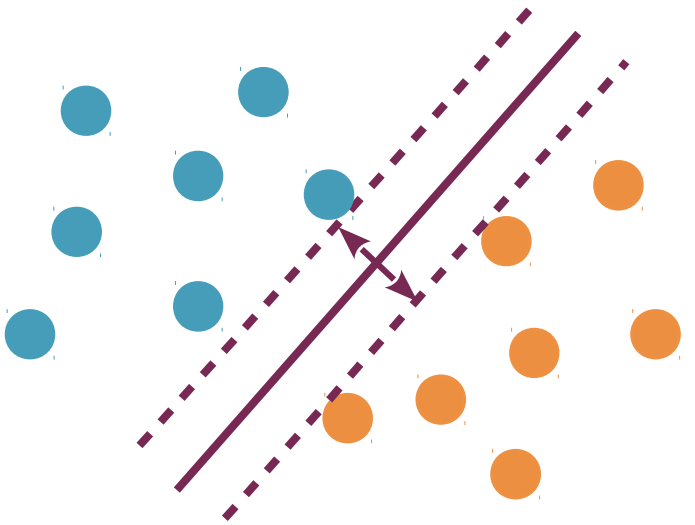
- Solve by **maximizing the likelihood**
- No analytical solution
- Use **gradient descent.**

Support Vector Machines

Large margin classifier

- Find the separating hyperplane with the **largest margin.**





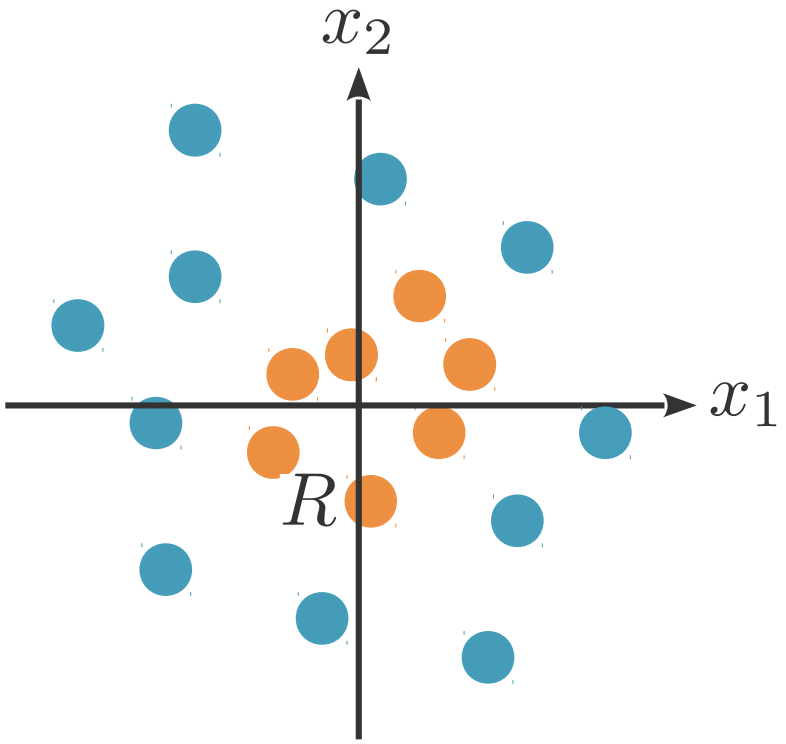
$$\arg\min_{w,b} \left(\frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \max(0, 1 - y^i (\langle w, x^i \rangle + b)) \right)$$

f(x)

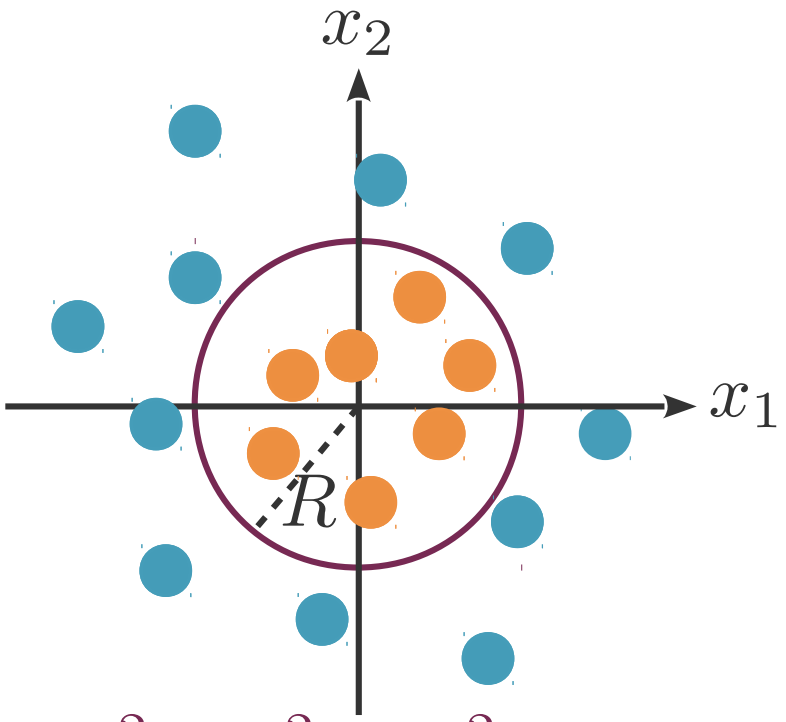
inverse of the margin

prediction error

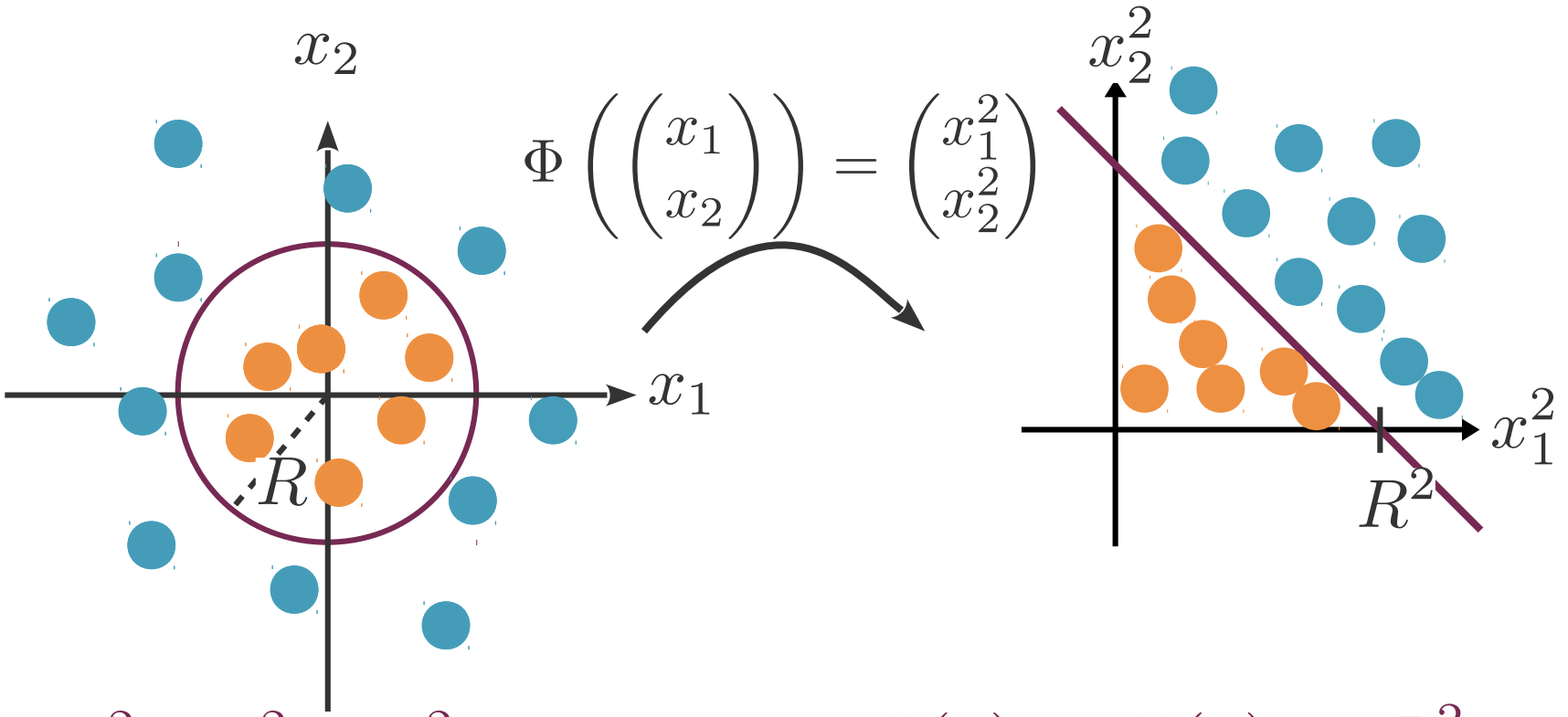
When lines are not enough



When lines are not enough



When lines are not enough



- **Non-linear mapping** to a **feature space**

When lines are not enough

- Non-linear mapping to a space of **higher dimension.**
- `https://www.youtube.com/watch?v=3liCbRZPrZA`
- Example:

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in \mathbb{R}^2 \qquad \Phi(\boldsymbol{x}) = \begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix} \in \mathbb{R}^3$$

$$\begin{aligned} \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}') \rangle &= x_1^2 {x'_1}^2 + 2x_1x_2x'_1x'_2 + x_2^2 {x'_2}^2 \\ &= \langle \boldsymbol{x}, \boldsymbol{x}' \rangle^2 \end{aligned}$$

The kernel trick

- The solution & SVM-solving algorithm can be expressed using **only** $k(\boldsymbol{x}, \boldsymbol{x}') = \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}') \rangle$
- Never need to **explicitly compute** $\Phi(\boldsymbol{x})$
- k: **kernel**
  - must be **positive semi-definite**
  - can be interpreted as a **similarity function.**

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i^* y^i k(\boldsymbol{x}^i, \boldsymbol{x}) + b^*$$

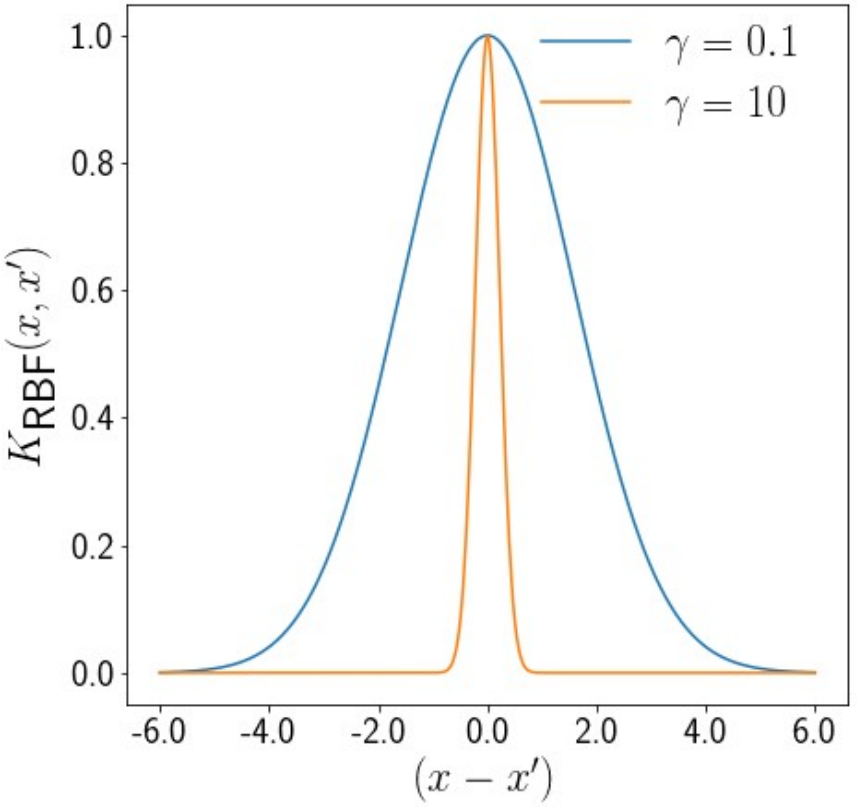
- **Support vectors:** training points for which $\alpha \neq 0$.

RBF kernel

- **Radial Basis Function** kernel, or **Gaussian** kernel

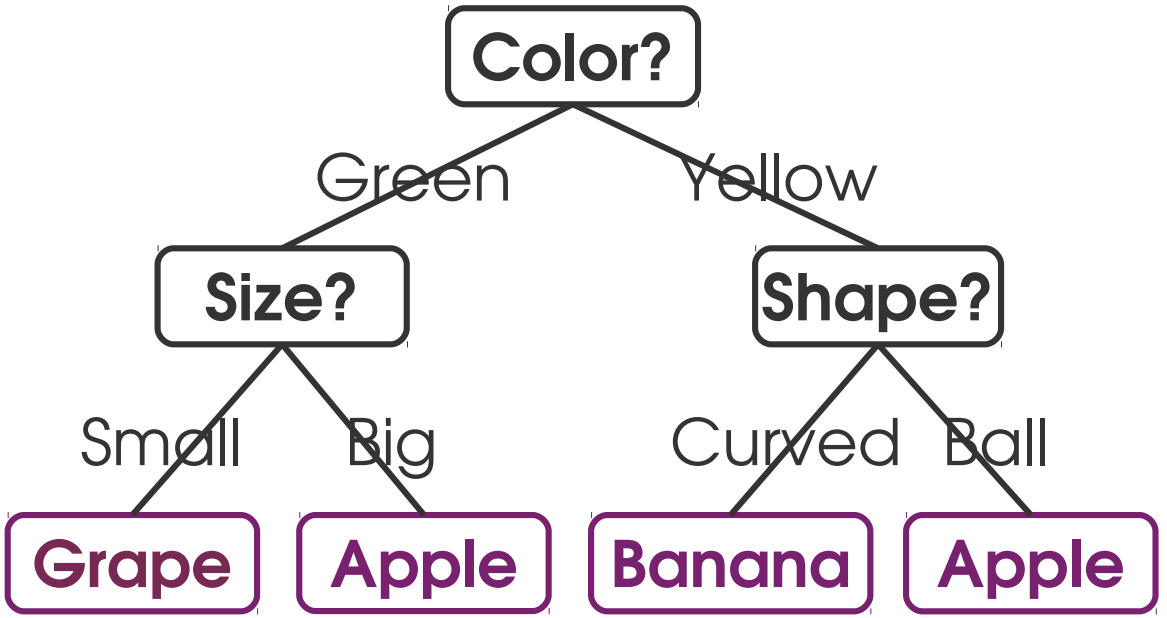
$$K_{\text{RBF}}(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\gamma||\boldsymbol{x} - \boldsymbol{x}'||^2)$$

- The feature space is **infinite-dimensional.**



Random Forests

Decision trees



- Well suited to **categorical features**
- Naturally handle **multi-class classification**
- **Interpretable**
- **Perform poorly.**

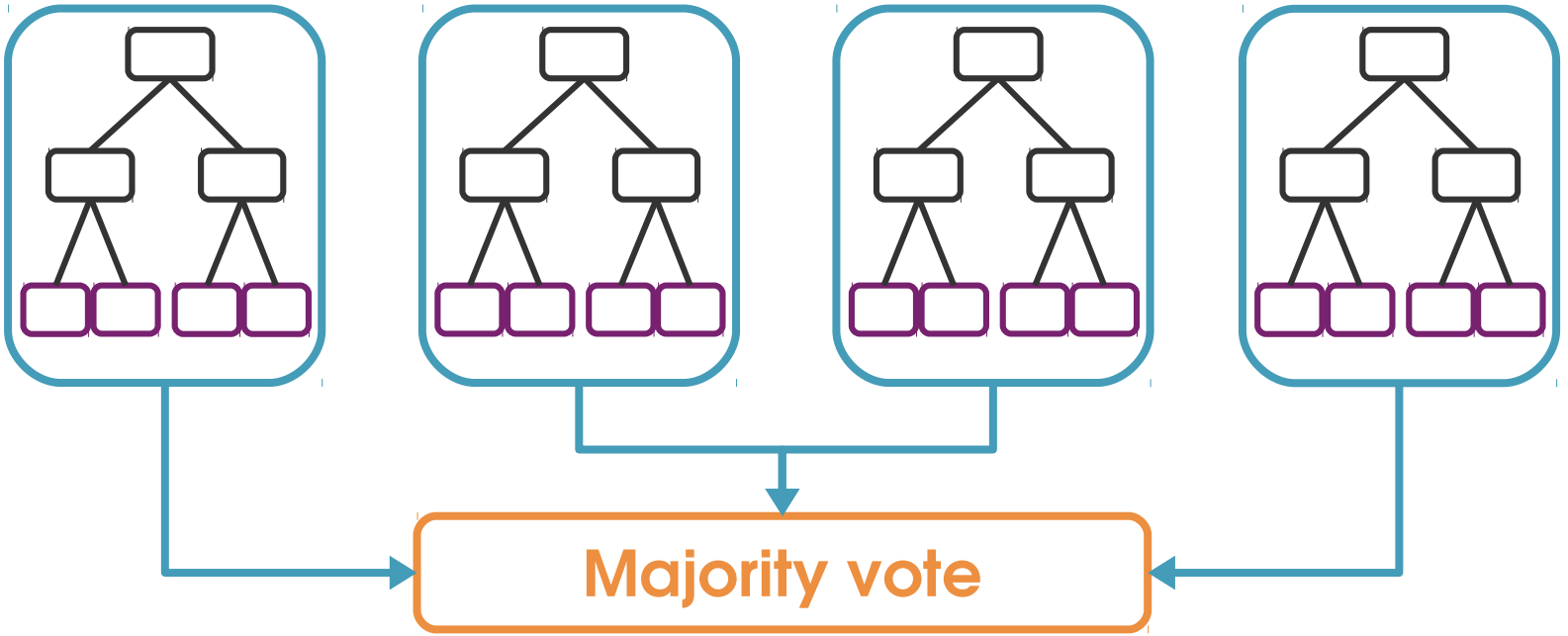
Ensemble learning

- Combining **weak learners** averages out their individual errors (**wisdom of crowds**)
- Final prediction:
  - Classification: **majority vote**
  - Regression: **average.**
- **Bagging:** weak learners are trained on **bootstraped samples** of the data (Breiman, 1996).

  bootstrap: sample n, with replacement.
- **Boosting:** weak learners are built iteratively, based on performance (Shapire, 1990).

Random forests

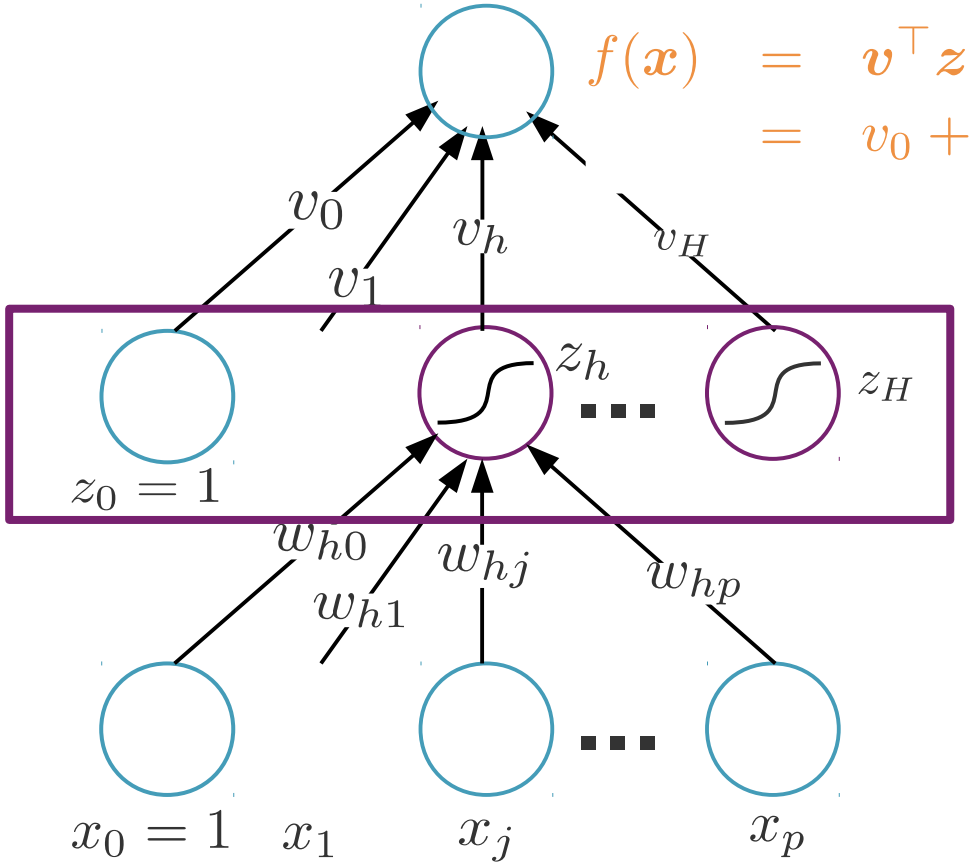
- Combine many decision trees



- Each tree is trained on a data set created using
  - A **bootstrap sample** (sample with replacement) of the **data**
  - A **random sample** of the **features.**
- Very **powerful** in practice.

(Deep) Neural Networks

(Deep) neural networks

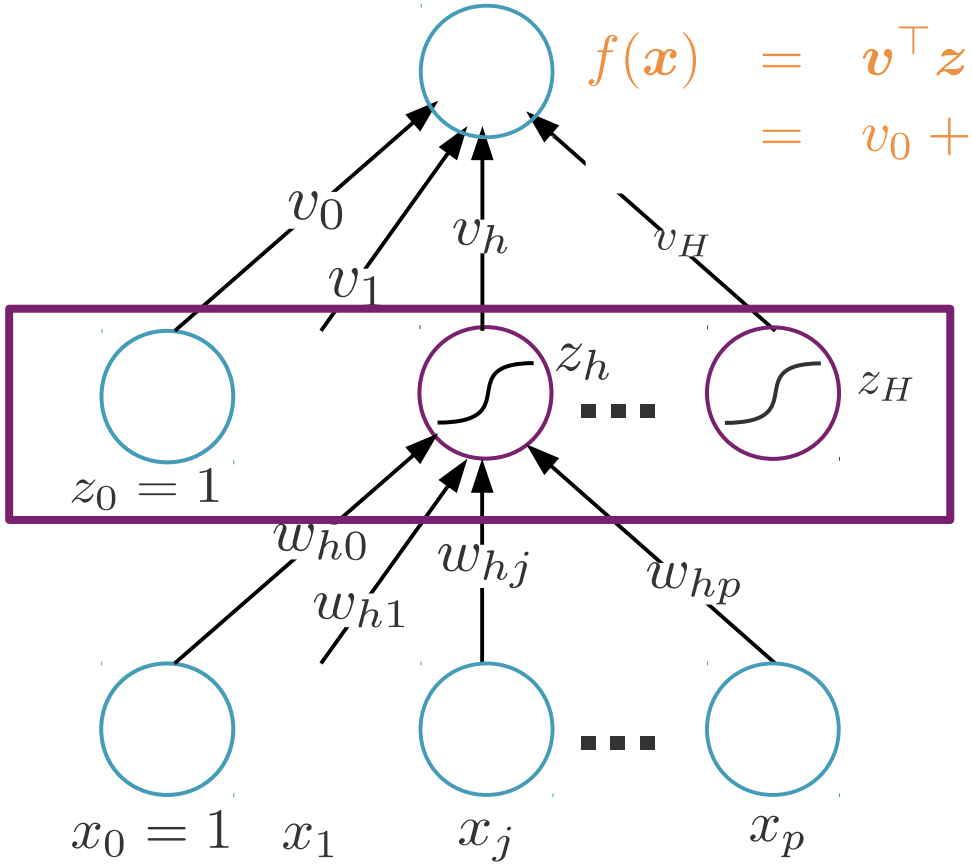


hidden layer

$$z_h = \frac{1}{1 + e^{-\boldsymbol{w}_h^\top \boldsymbol{x}}}$$

- Nothing more than a (possibly complicated) **parametric model**

(Deep) neural networks

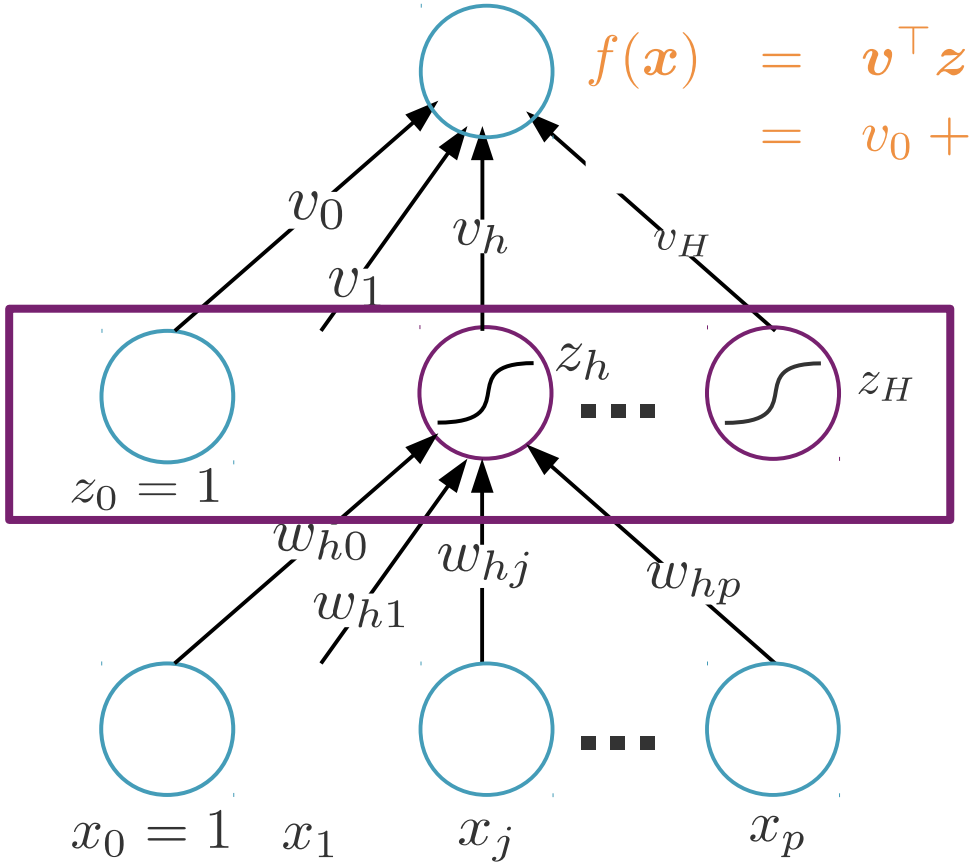


hidden layer

$$z_h = \frac{1}{1 + e^{-w_h^\top x}}$$

- **Fitting weights:**
  - **Non-convex** optimization problem
  - Solved with gradient descent
  - Can be difficult to tune.

(Deep) neural networks



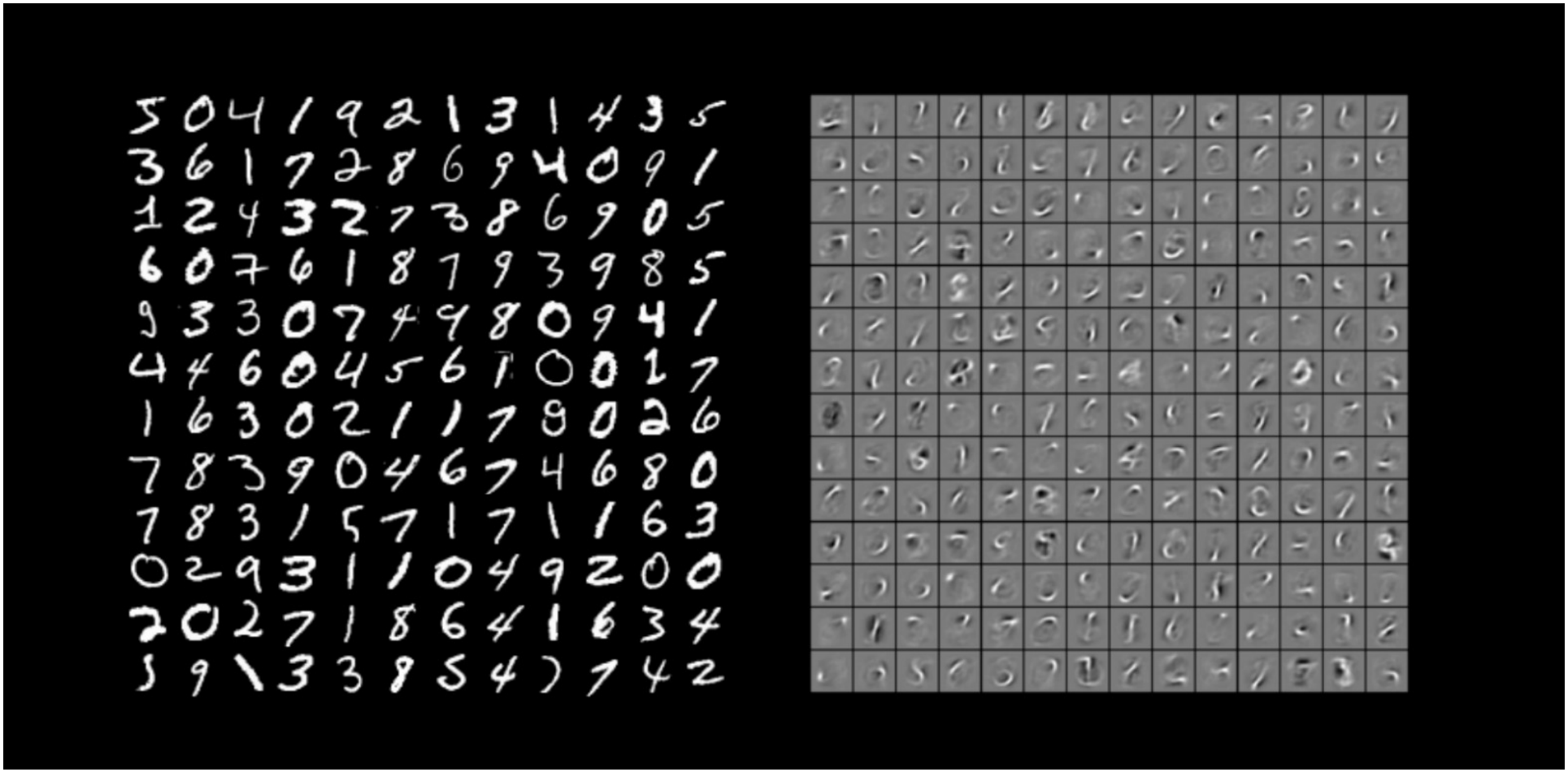
hidden layer

$$z_h = \frac{1}{1 + e^{-\boldsymbol{w}_h^\top \boldsymbol{x}}}$$

- Learn an **internal representation** of the data on which a **linear model** works well.
- Currently one of the **most powerful** supervised learning algorithms **for large training sets.**

(Deep) neural networks

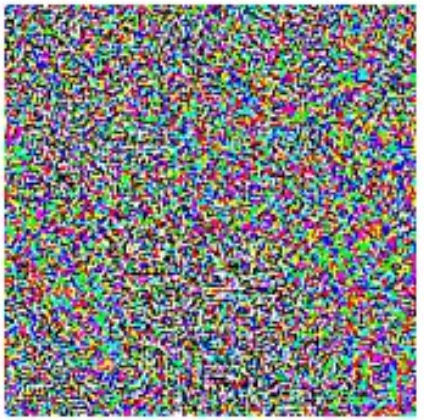
Internal representation of the digits data



Yann Le Cun et al. (1990)

Adversarial examples

$+ .007 \times$



$=$

panda

gibbon

Goodfellow et al. ICLR 2015
https://arxiv.org/pdf/1412.6572v3.pdf

October 14

Deep learning for image recognition

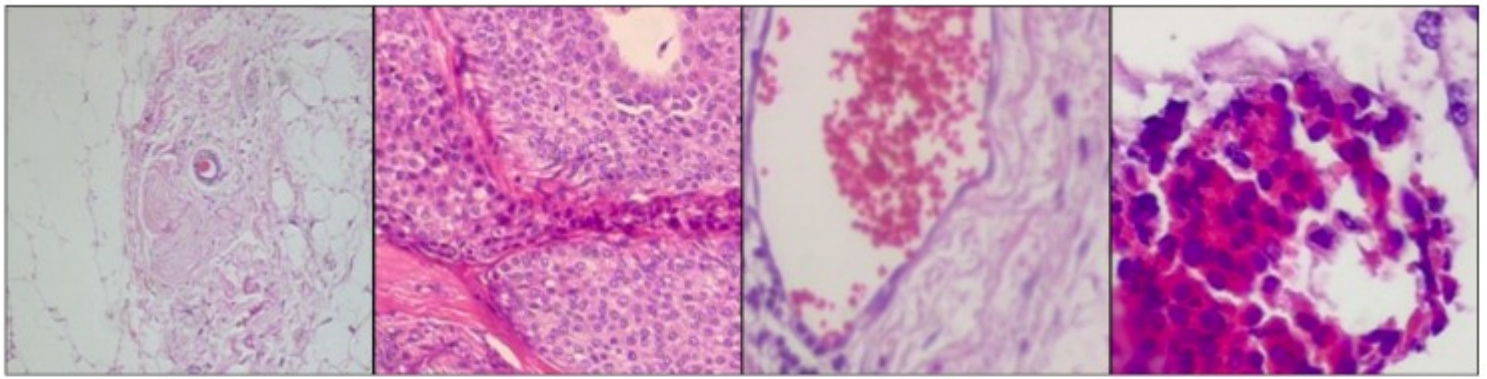


Fig. 2. A malignant breast tumor acquired from a single slide seen in different magnification factors: $40\times$, $100\times$, $200\times$, and $400\times$.

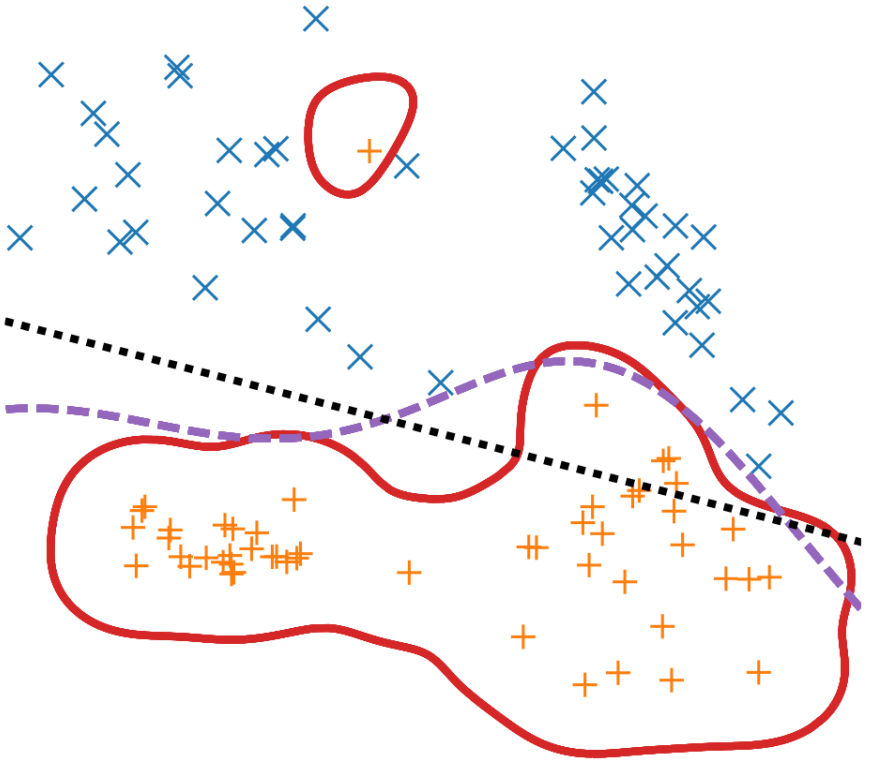
Source: DOI: 10.1109/ICPR.2016.7900002

Generalization & overfitting

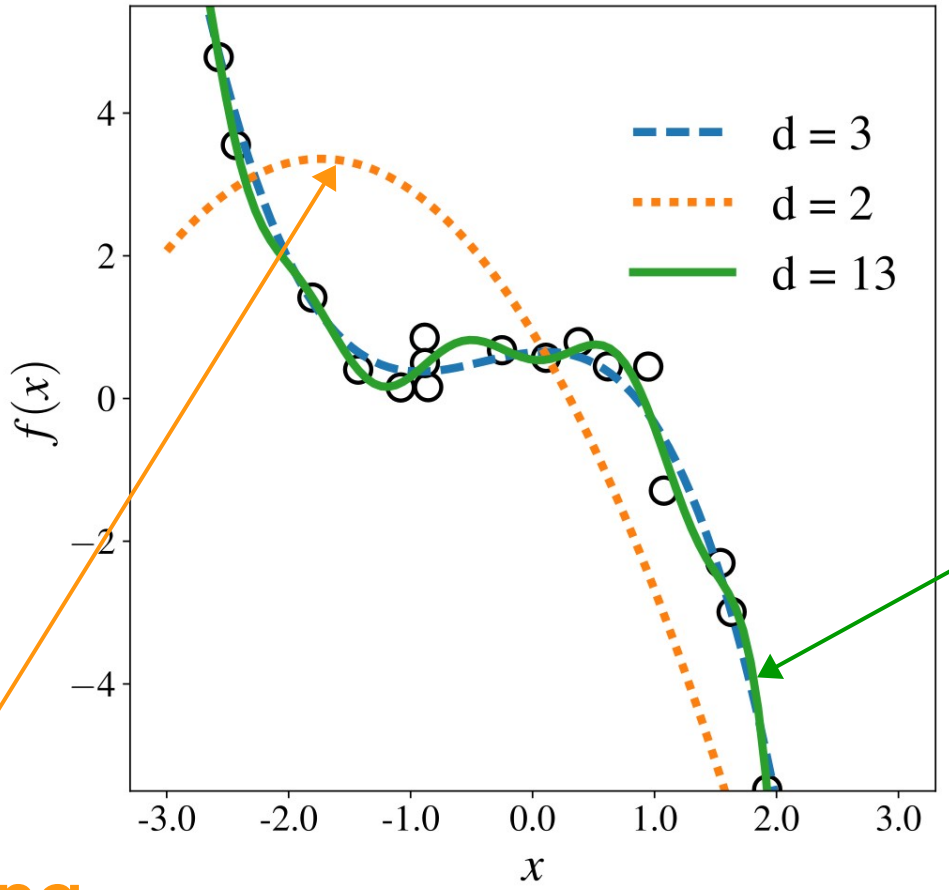
Generalization

- **Goal:** build models that make good predictions on **new data.**
- Models that work "too well" on the data we learn on tend to model noise as well as the underlying phenomenon: **overfitting.**

Overfitting (Classification)



Overfitting (Regression)



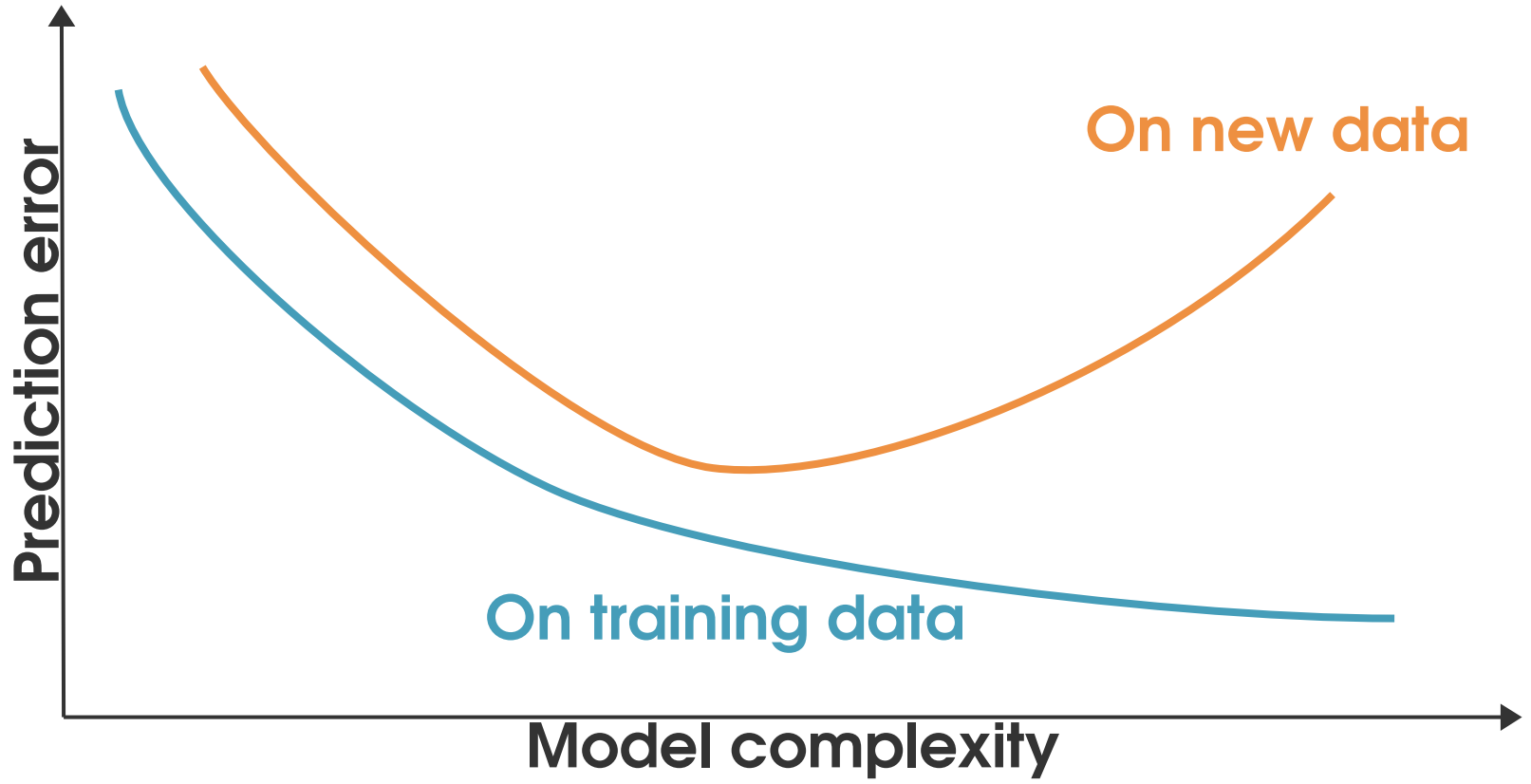
Underfitting

Overfitting

Model complexity

Simple models are

- more **plausible** (Occam's Razor)
- easier to **train, use,** and **interpret.**



Regularization

- Prevent overfitting by designing an **objective function** that accounts not only for the **prediction error** but also for **model complexity.**

```
min (empirical_error + λ*model_complexity)
```

- Rember the SVM

$$\arg\min_{\boldsymbol{w},b} \left(\frac{1}{2}||\boldsymbol{w}||^2 + C \sum_{i=1}^{n} \max(0, 1 - y^i (\langle \boldsymbol{w}, x^i \rangle + b)) \right)$$

inverse of the margin

f(x)

prediction error

Ridge regression

$$\hat{\boldsymbol{\beta}}_{\text{ridge}} = \arg\min_{\boldsymbol{\beta}} \underbrace{||\boldsymbol{y} - X\boldsymbol{\beta}||_2^2}_{\text{prediction error}} + \underbrace{\lambda}_{\text{hyperparameter}} \underbrace{||\boldsymbol{\beta}||_2^2}_{\text{regularizer}}$$

- **Unique solution, always exists**

$$\hat{\boldsymbol{\beta}}_{\text{ridge}} = (X^\top X + \lambda I)^{-1} X^\top \boldsymbol{y}$$

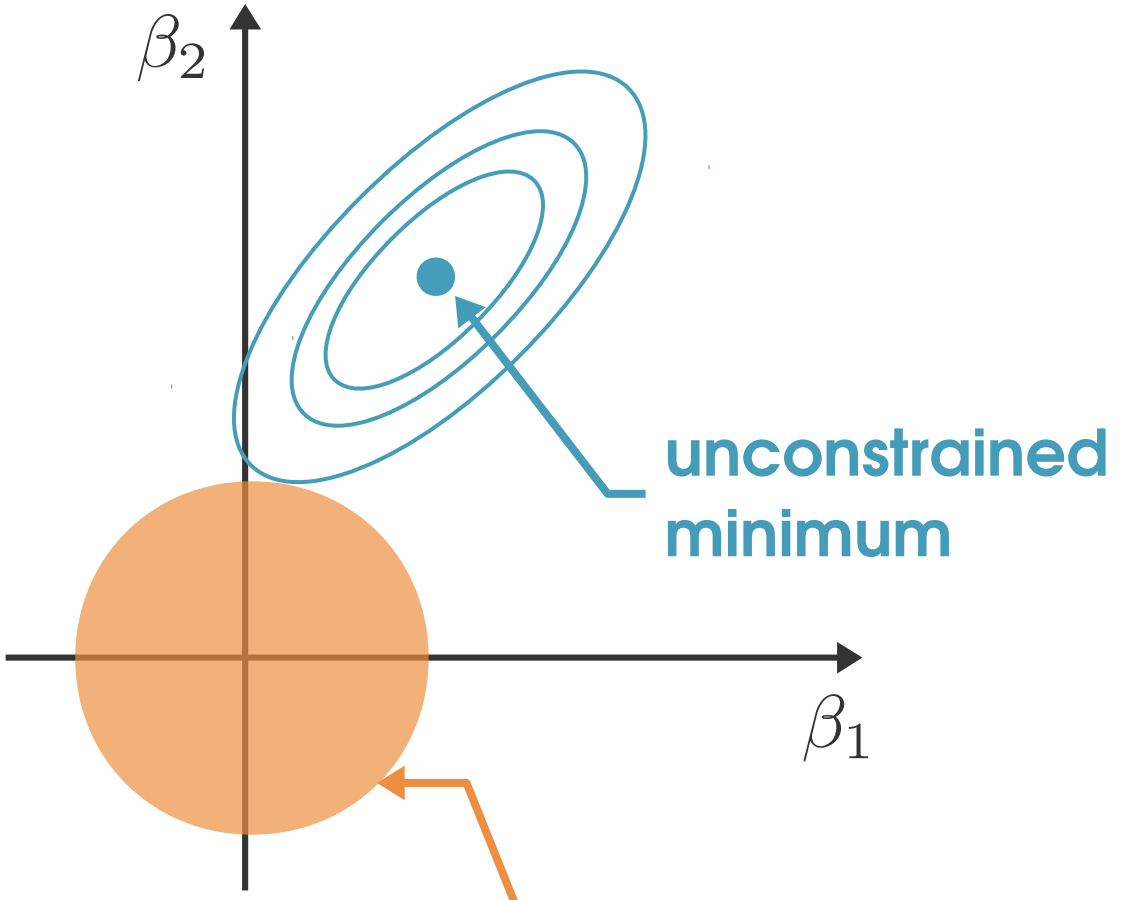
- **Grouped selection:**

  Correlated variables get similar weights.

- **Shrinkage**

  Coefficients shrink towards 0.

Geometry of ridge regression



Model selection & evaluation

No free lunch theorem

- Any two optimization algorithms are equivalent when their performance is averaged over all possible problems.
- For any learner, **there is a learning task that it will not learn well.**

Wolpert & Macready, 1997

Evaluation on held-out data

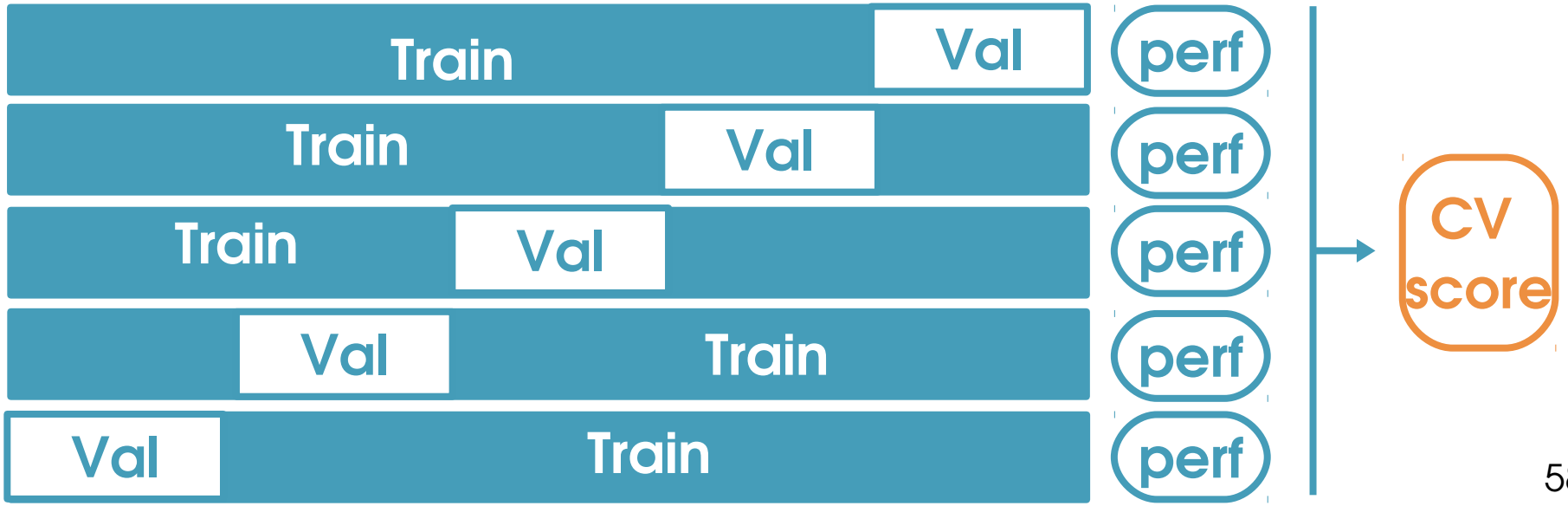
- If we evaluate the model on the data we've used to train it, we risk **over-estimating performance.**
- **Proper procedure:**
  - Separate the data in **train/test** sets

Evaluation on held-out data

- **Proper procedure:**
  - Separate the data in **train/test** sets

Evaluation on held-out data

- **Proper procedure:**
  - Separate the data in **train/test** sets
  - Use a **cross-validation** on the train set to find the best algorithm + hyperparameter(s)



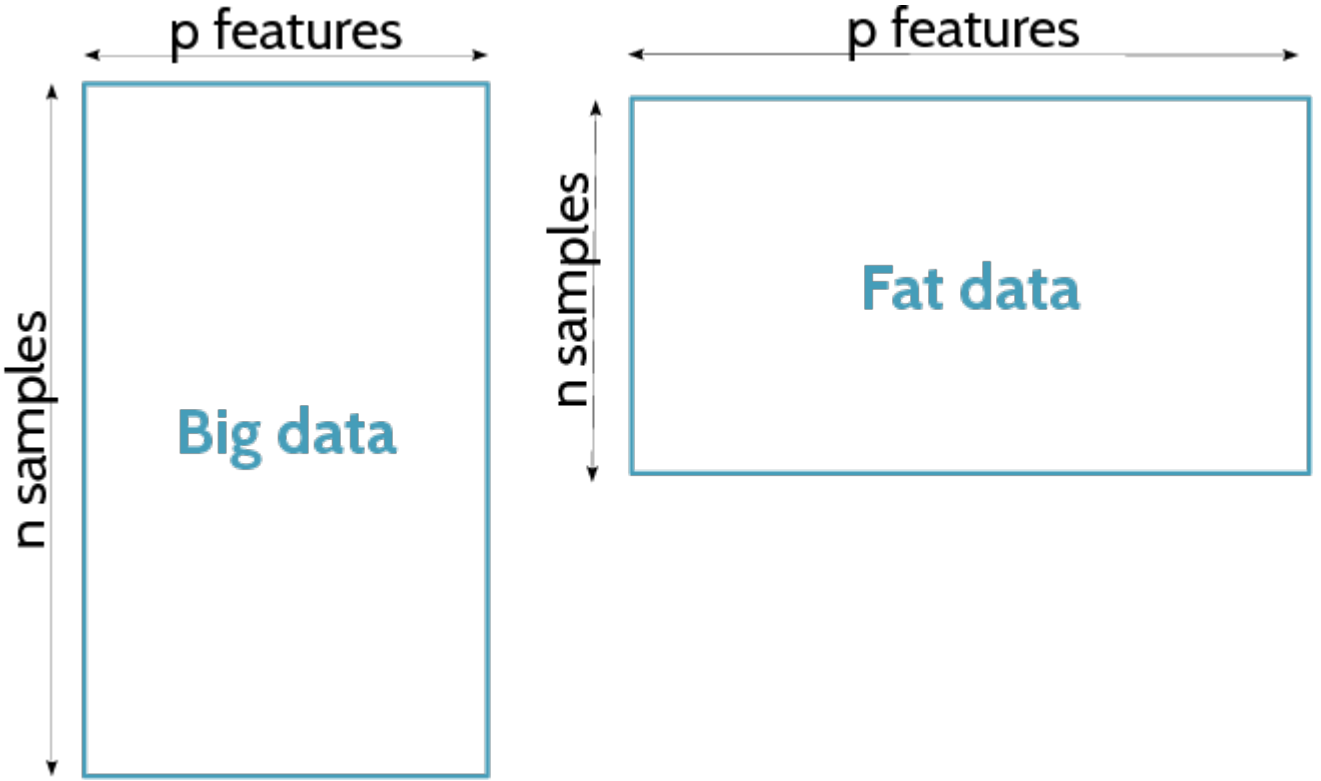
Evaluation on held-out data

- **Proper procedure:**
  - Separate the data in **train/test** sets
  - Use a **cross-validation** on the train set to find the best algorithm + hyperparameter(s)
  - Train this best algorithm + hyperparameter(s) on the entire train set
  - The performance on the test set estimates **generalization performance.**

Practical

ML in high dimension

“Big data” vs “fat data”



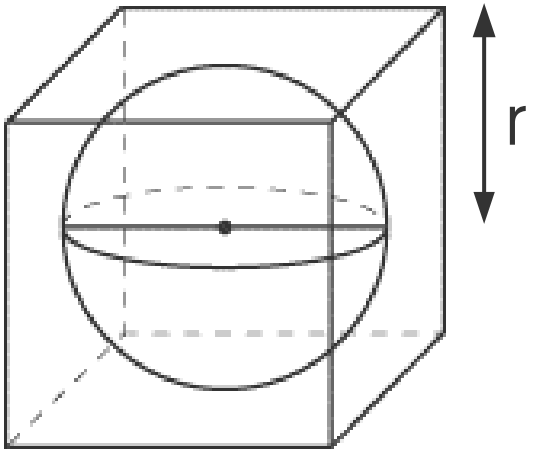
Challenges when p>>n:

- **Computational** challenges (e.g. linear regression)
- **Curse of dimensionality**
- Higher risk of **overfitting.**

Curse of dimensionality

- Methods/intuitions that work in low dimension **do not necessarily apply in high dimension.**
- Hyperspace is very big and **everything is far apart**

  Most points inside a cube are outside of the sphere inscribed in this cube.



- **Dimensionality reduction:**
  - Feature extraction
  - Feature selection.

Feature extraction

- Project the data onto a lower-dimensional space
- Creates **new features** → **interpretability?**
- **Matrix factorization** techniques:

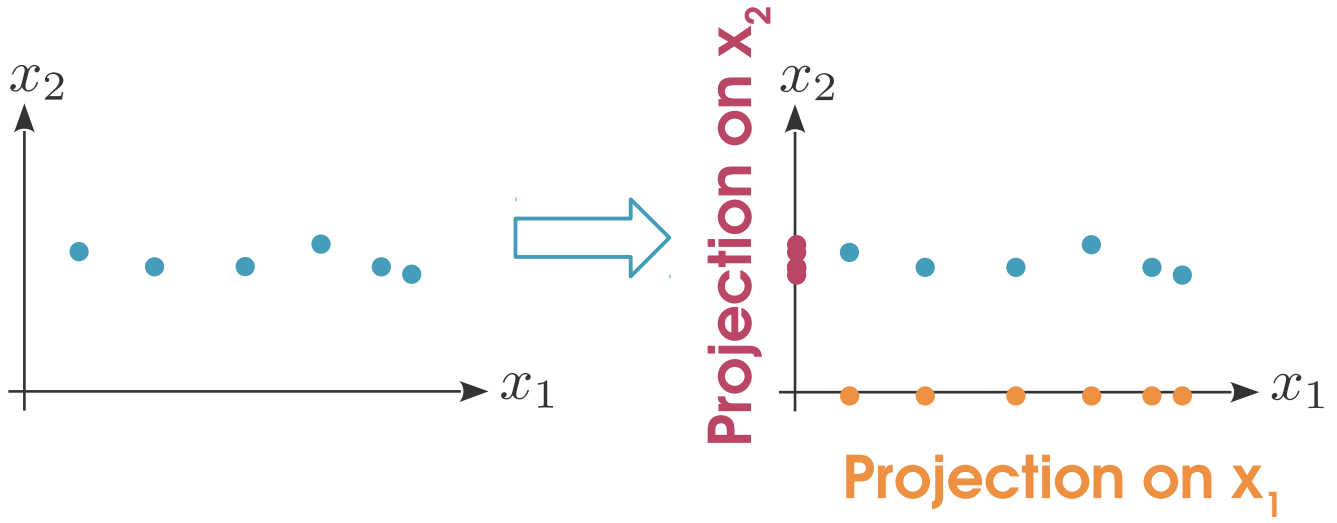
  PCA & kPCA, factorial analysis, non-negative matrix factorization.
- **Manifold learning** techniques

  MDS, tSNE
- **Autoencoders:** neural networks

  Goal: recover the input as well as possible.

Principal Components Analysis

- Find a space of dimension *m* s.t. the **variance** of the data projected onto that space is **maximal.**



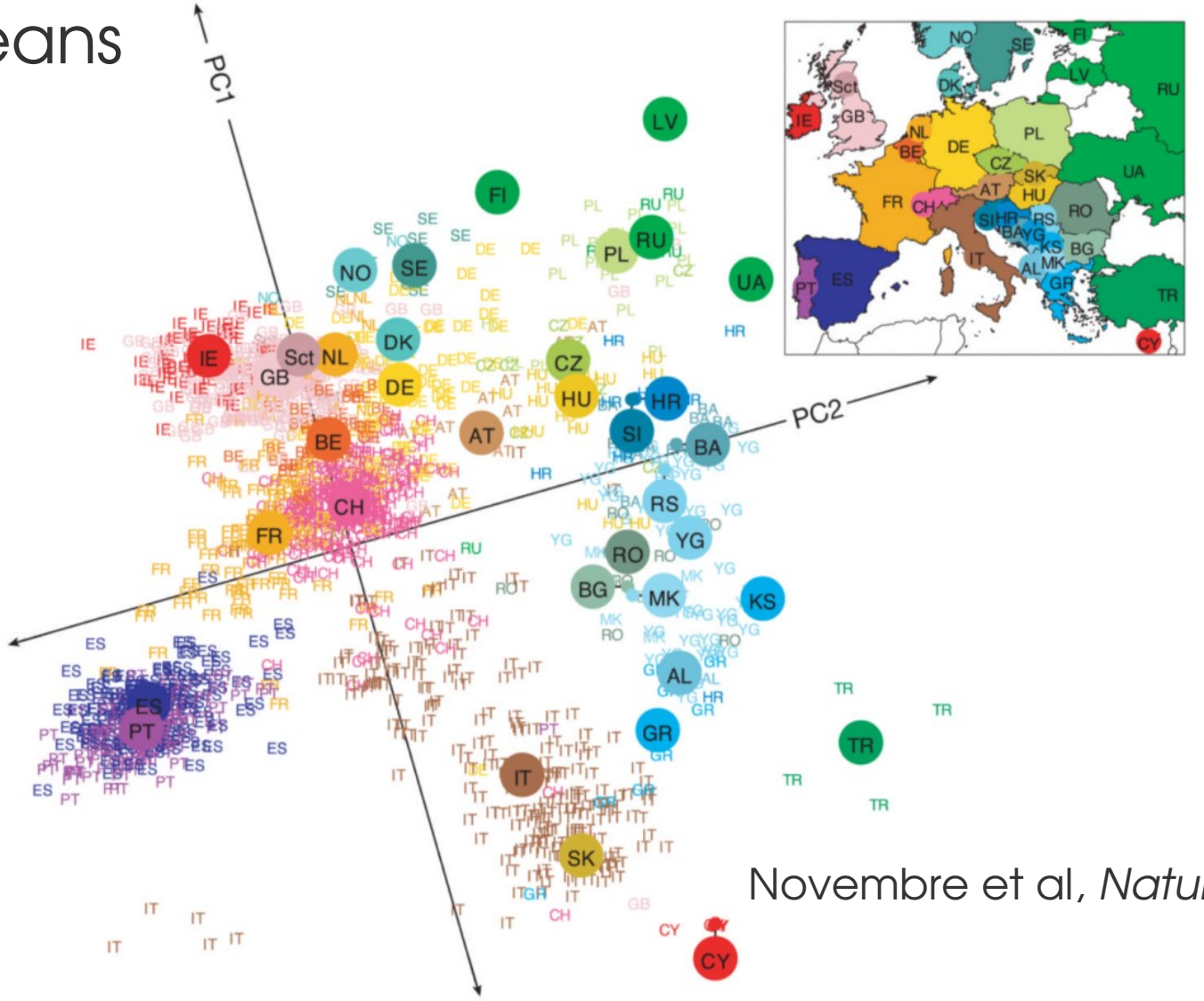
Principal Components Analysis

- Find a space of dimension *m* s.t. the **variance** of the data projected onto that space is **maximal.**
- Find orthonormal directions $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_m\}$ s.t.
  - $\boldsymbol{w}_j = \arg\max_{\boldsymbol{w} \in \mathbb{R}^p} \mathrm{Var}(\boldsymbol{w}^\top X)$
  - $\boldsymbol{w}_j^\top \boldsymbol{w}_k = 0$
  - $||\boldsymbol{w}_j|| = 1.$

- These directions, called **principal components** are the **eigenvectors** of $X^\top X$, sorted by decreasing eigenvalue.

Principal Components Analysis

SNP data (Affymetrix 500K) for
1387 Europeans



Novembre et al, *Nature* 2008

Supervised feature selection

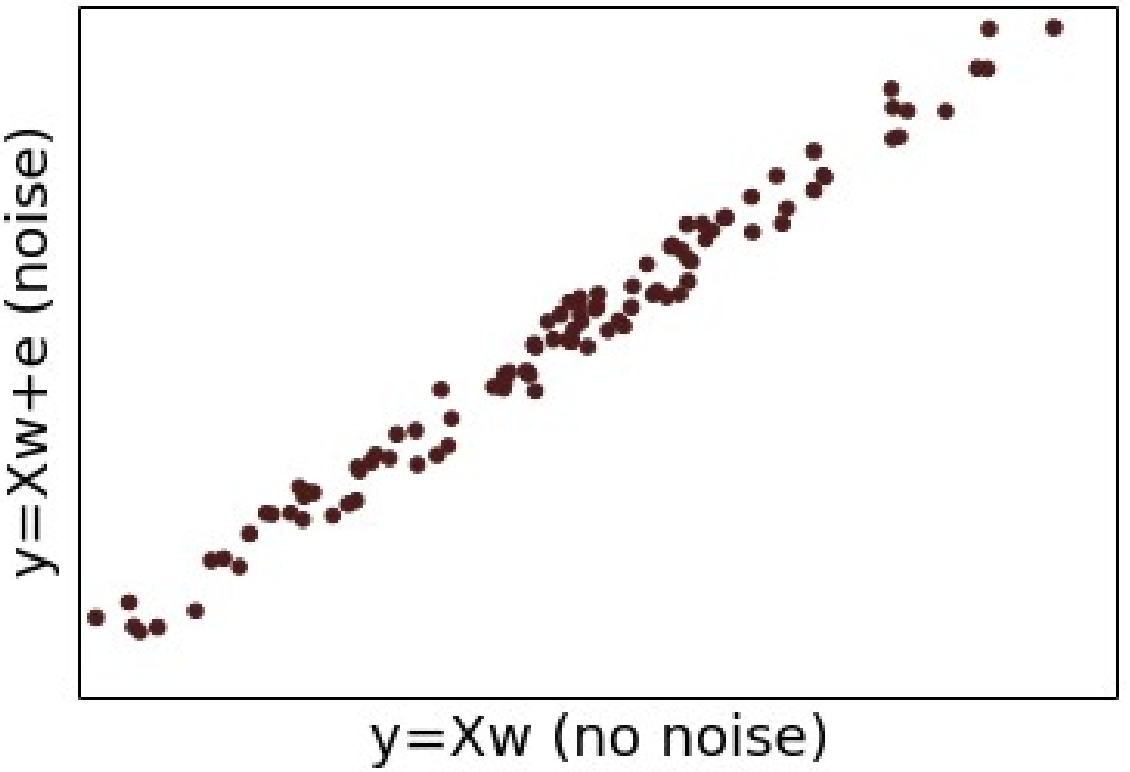
Keep relevant features only

- **Filter approaches:** apply a statistical test to assign a score to each feature
- **Wrapper approaches:** use a greedy search to find the best set of features for a given ML algorithm
- **Embedded approaches:** fit a **sparse** model, i.e. that is encouraged to *not* use all the features.

A p>>n simulation

$$\boldsymbol{y} = X\boldsymbol{w} + \epsilon$$

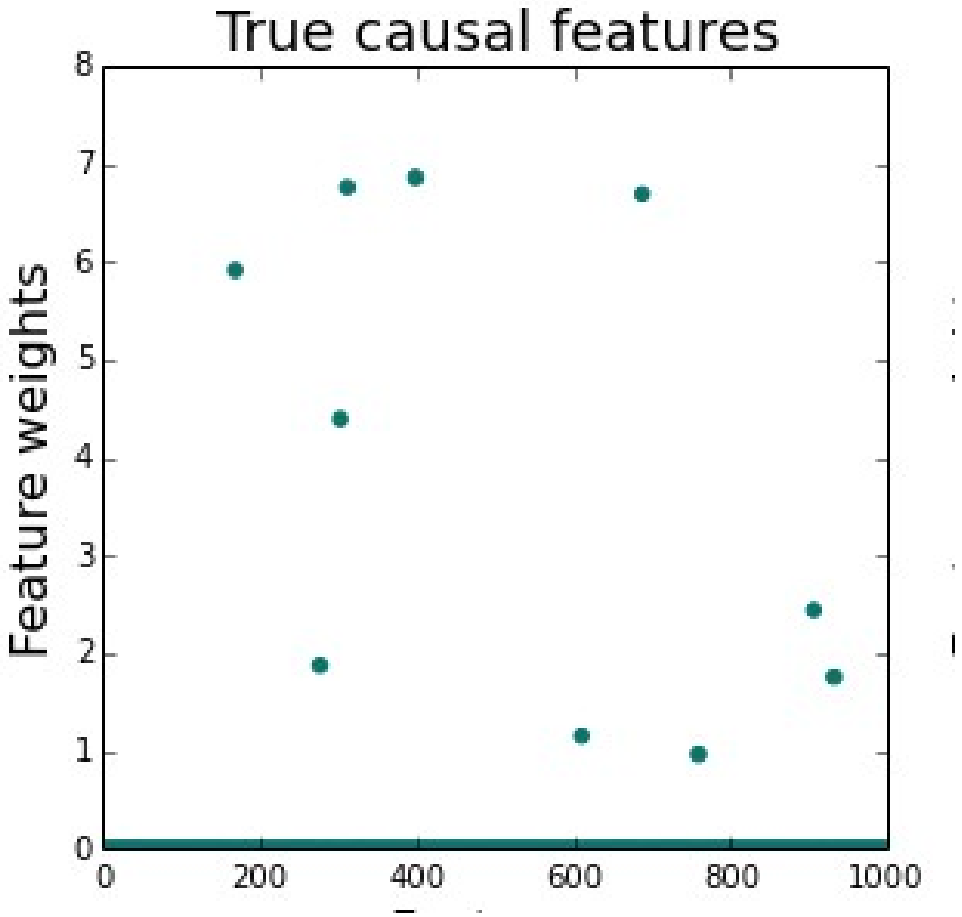
p=1 000, n=100, 10 causal features.



A p>>n simulation

$$\boldsymbol{y} = X\boldsymbol{w} + \epsilon$$

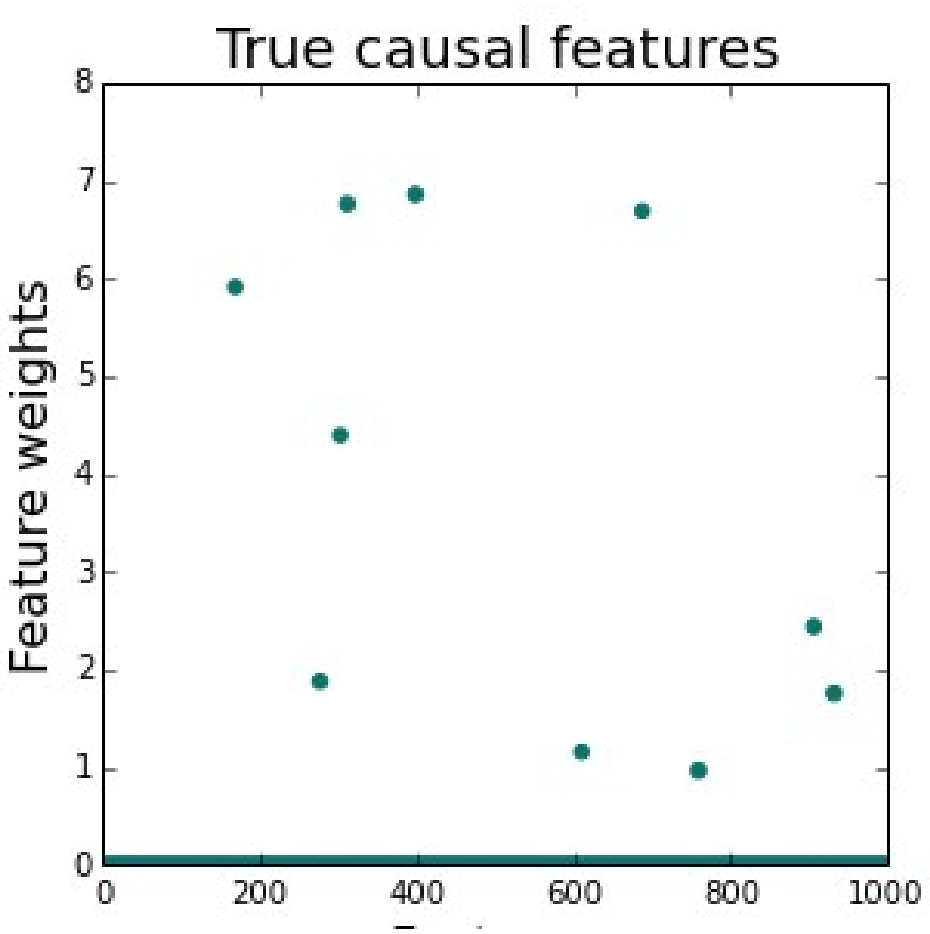
p=1 000, n=100, 10 causal features.

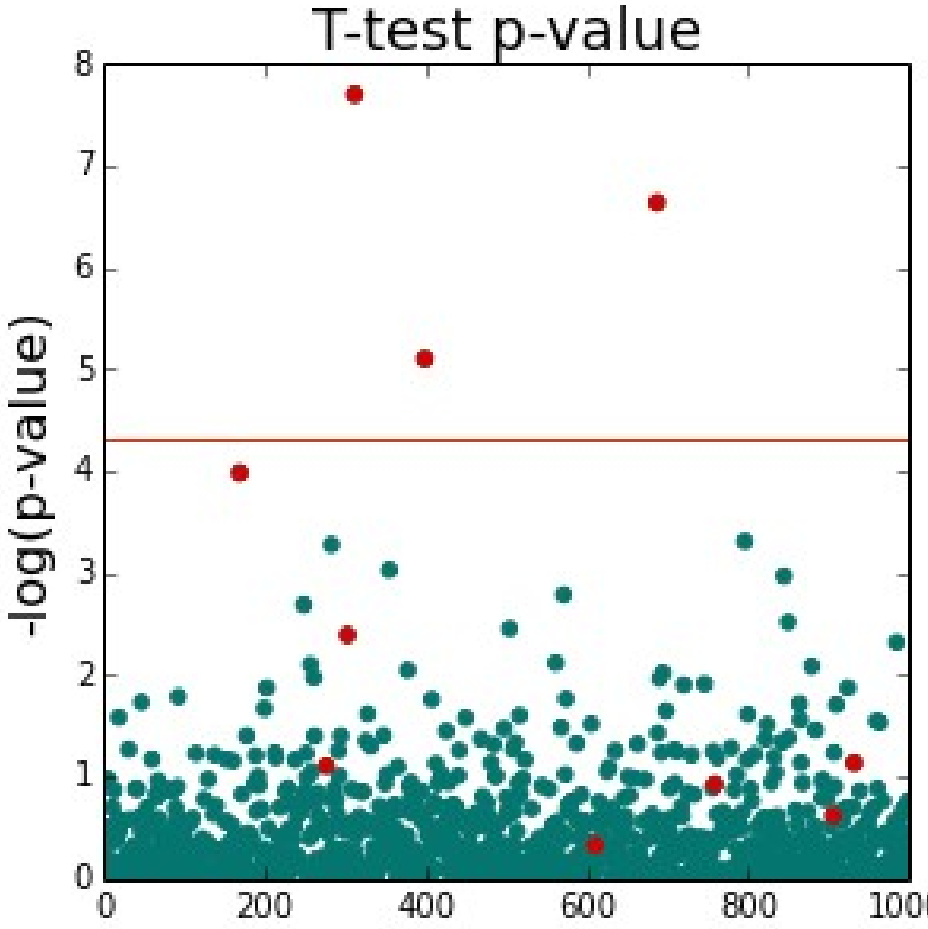


A p>>n simulation

$$y = Xw + \epsilon$$

Statistical test

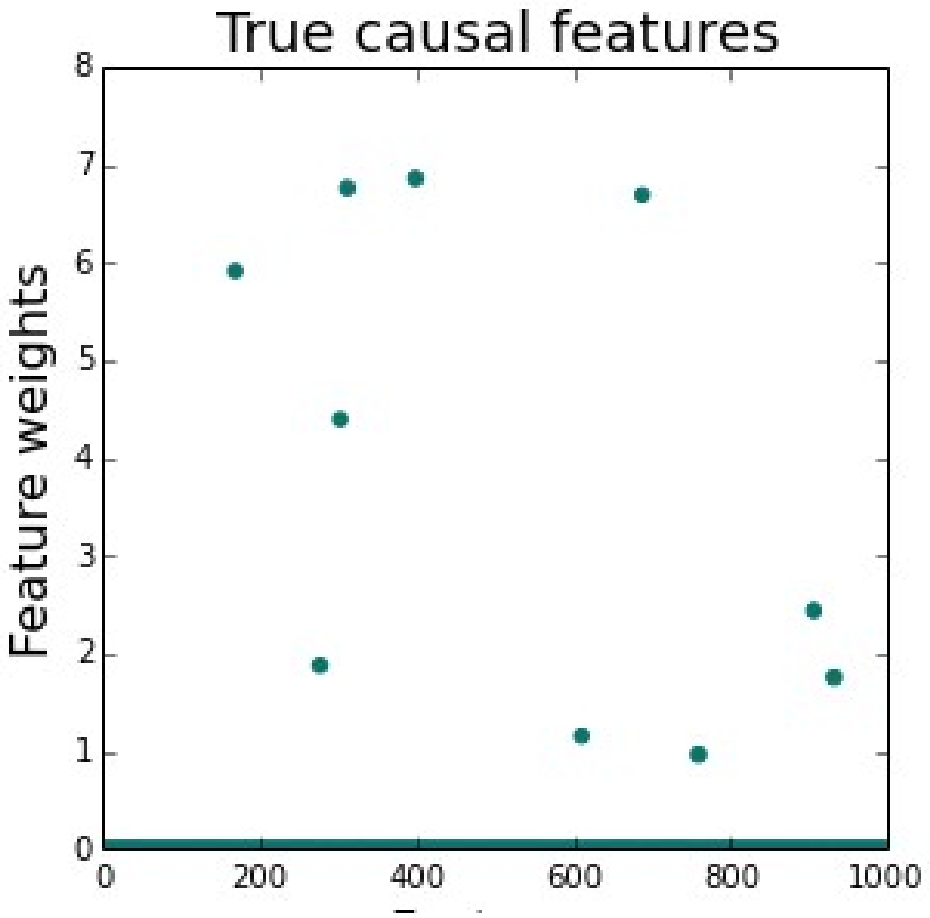


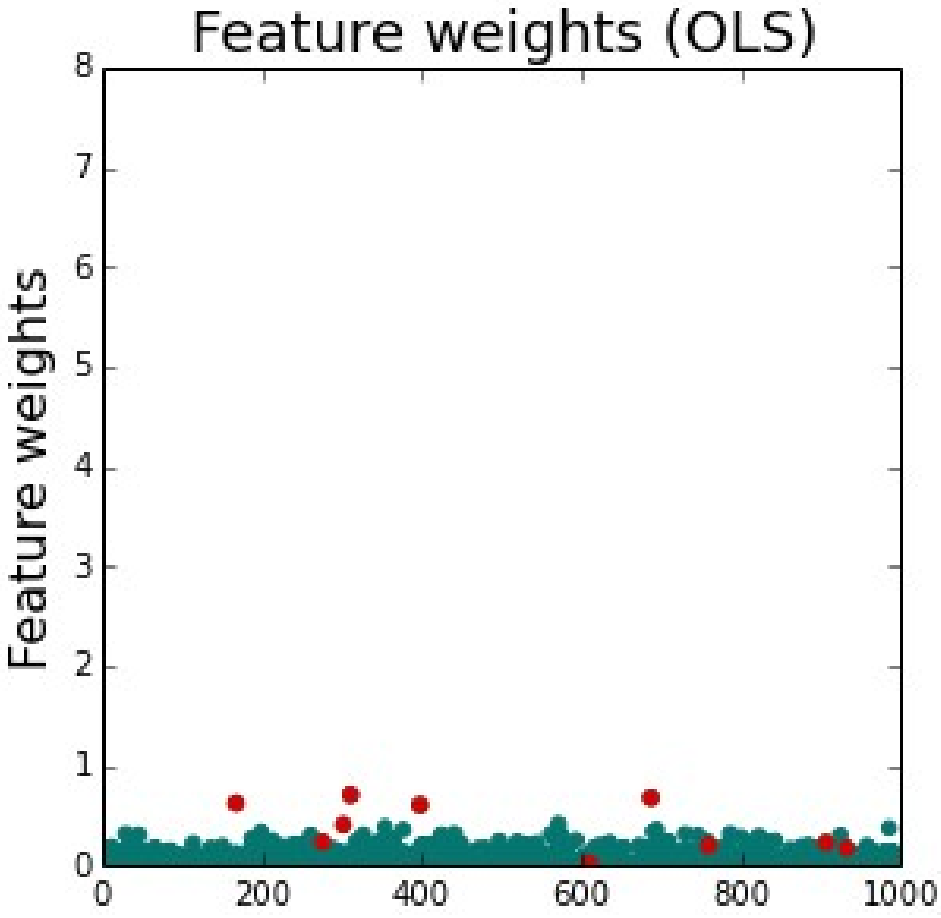


A p>>n simulation

$$y = Xw + \epsilon$$

Linear regression



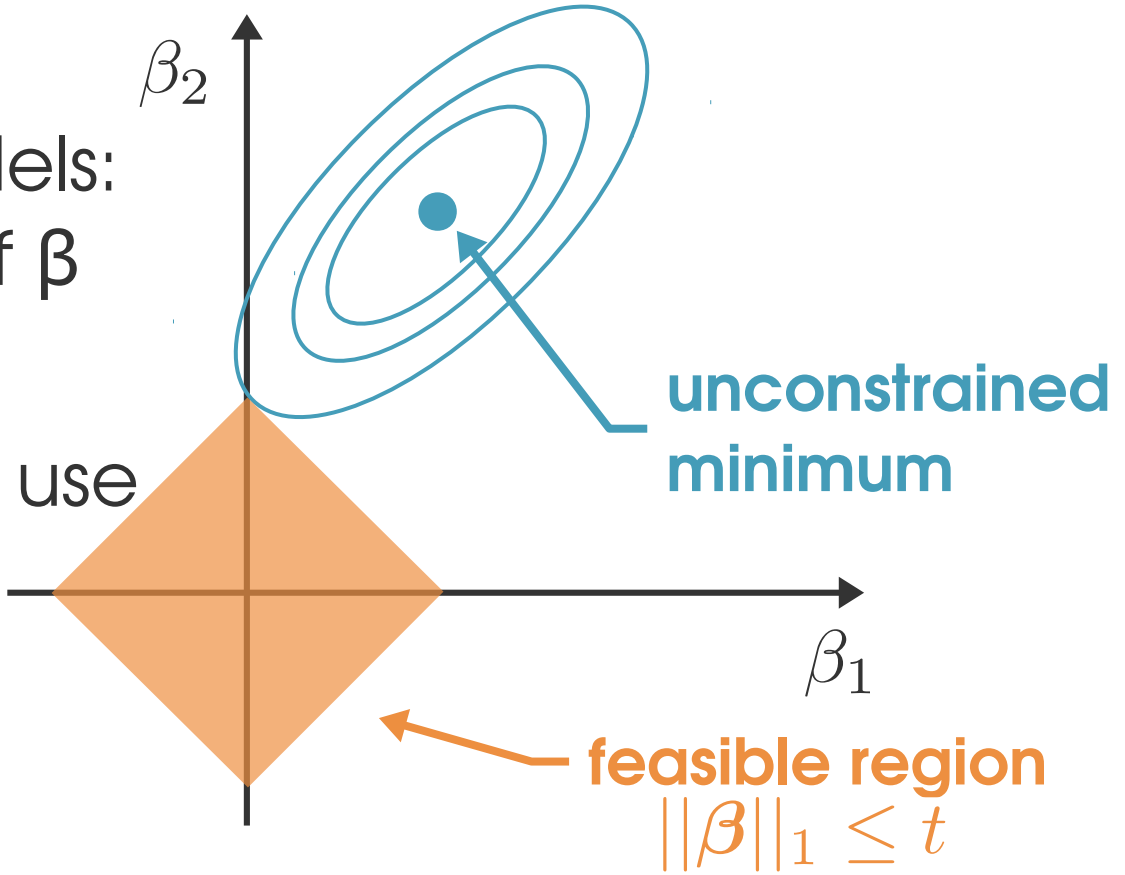


Lasso

$$\hat{\boldsymbol{\beta}}_{\text{lasso}} = \arg\min_{\boldsymbol{\beta}} ||\boldsymbol{y} - X\boldsymbol{\beta}||_2^2 + \lambda \; ||\boldsymbol{\beta}||_1$$

prediction error regularizer

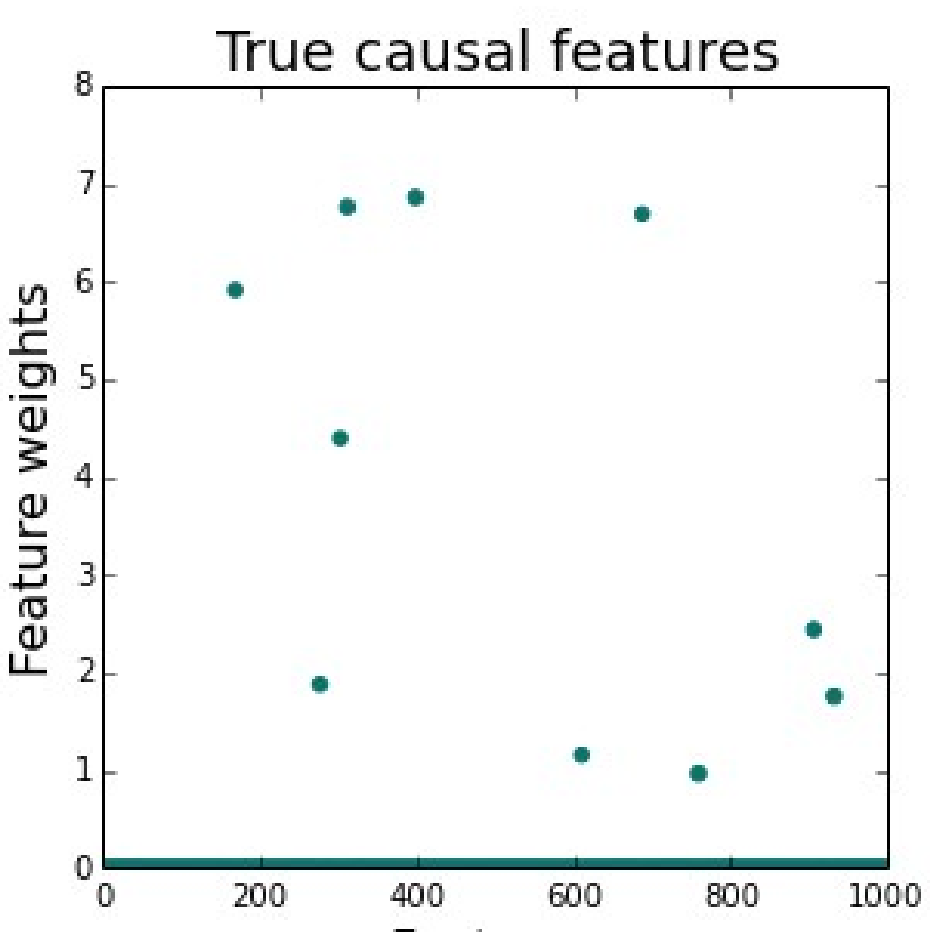
- Create **sparse** models: drive coefficients of β towards 0.
- No explicit solution, use **gradient descent.**

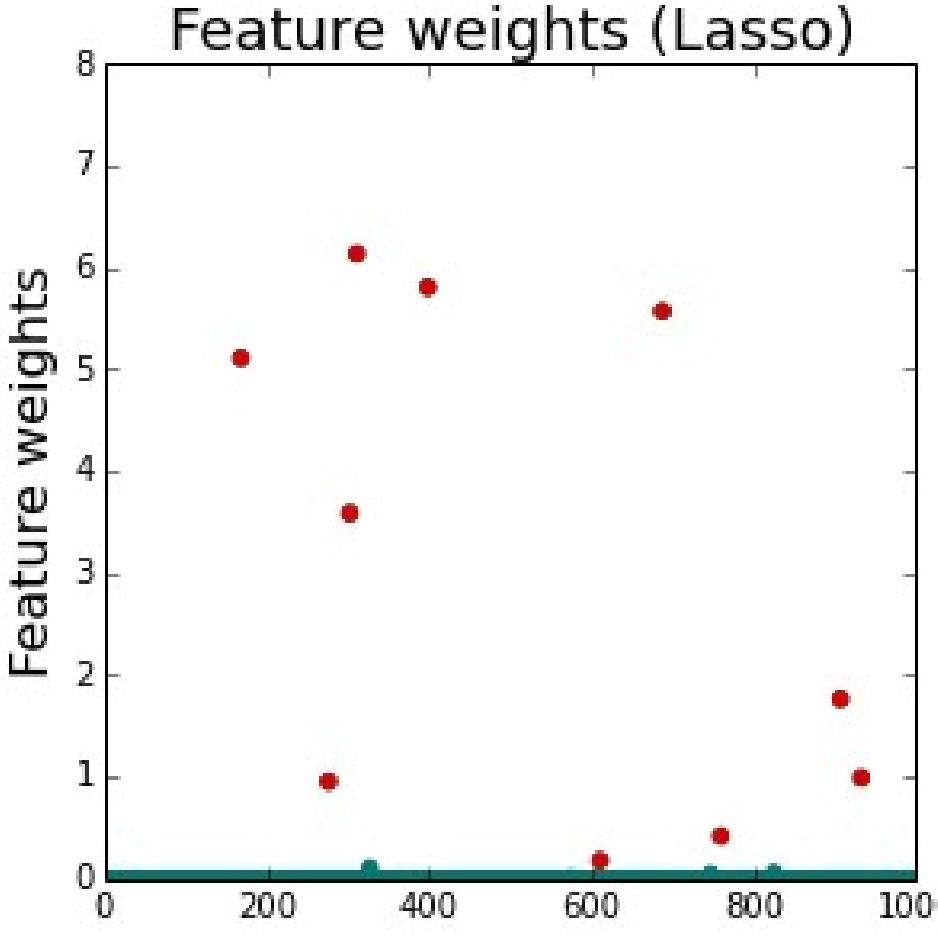


A p>>n simulation

$$y = Xw + \epsilon$$

Lasso





ElasticNet

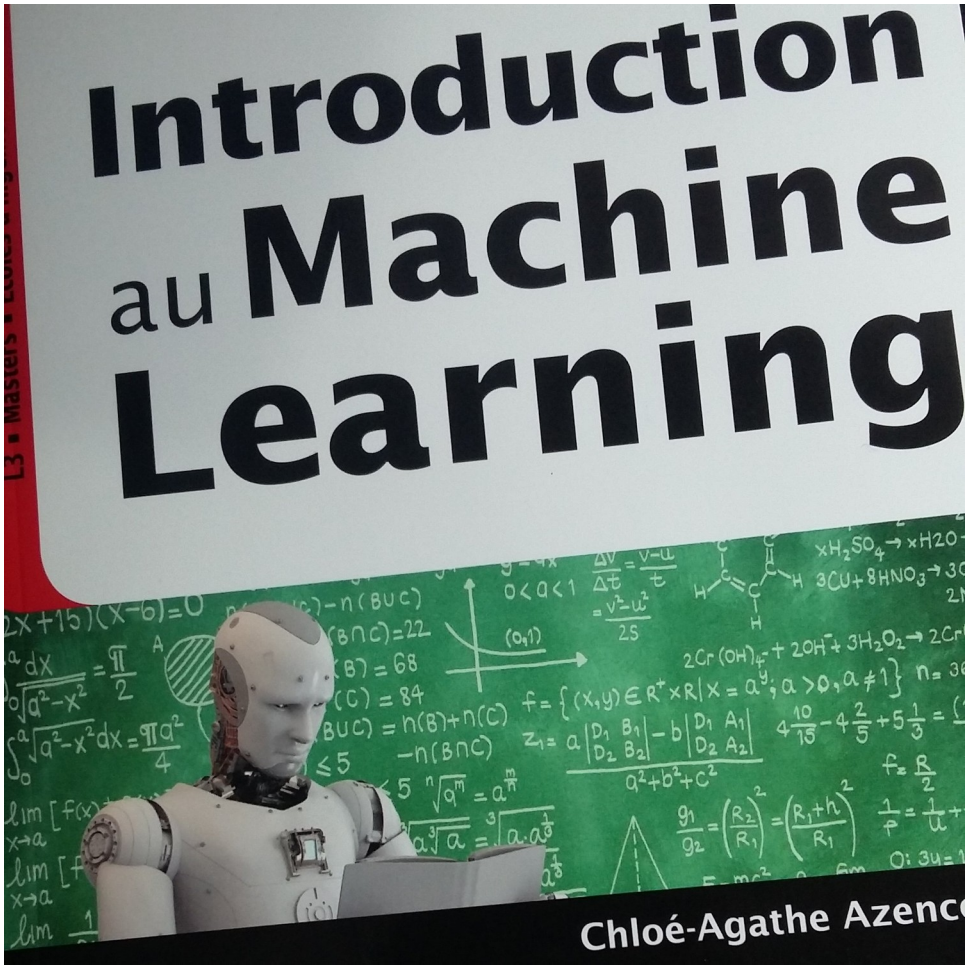
- Lasso tends to be **unstable:**
  - Picks one of several correlated variables at random
  - Different results on similar data sets
- **ElasticNet:** combine Ridge with Lasso

$$\hat{\boldsymbol{\beta}}_{\text{enet}} = \arg\min_{\boldsymbol{\beta}} ||\boldsymbol{y} - X\boldsymbol{\beta}||_2^2 + \lambda \ \ (\alpha||\boldsymbol{\beta}||_1 + (1-\alpha)||\boldsymbol{\beta}||_2^2)$$

ML Toolboxes

- **Python: scikit-learn**
  `http://scikit-learn.org`

- **R: Machine Learning Task View**
  `http://cran.r-project.org/web/views/MachineLearning.html`
- **Matlab™: Machine Learning with MATLAB**
  `http://mathworks.com/machine-learning/index.html`
  - Statistics and Machine Learning Toolbox
  - Deep Learning Toolbox
- **Deep learning software**
  - **Theano** `http://deeplearning.net/software/theano/`
  - **TensorFlow** `http://www.tensorflow.org/`
  - **Caffe** `http://caffe.berkeleyvision.org/`
  - **Keras** `https://keras.io/`


Introduction
au Machine
Learning
Chloé-Agathe Azenc

Summary

Machine learning =

data + model + objective function

- Catalog:
  - Supervised vs unsupervised
  - Parametric vs non-parametric
  - Linear models, SVMs, random forests, neural networks.
- **Key concerns:**
  - Avoid overfitting ⇒ regularization
  - Measure generalization performance.
- **p >> n setting:**
  - Feature selection
  - Sparsity (Lasso, ElasticNet)