

Machine learning and (geo)physical modelling-1

julien.brajard@nersc.no

October 2023

NERSC

<https://github.com/brajard/MAT330>

Overview of the next lectures

1. Lecture 1 (Tuesday 03 Oct. 2023):
Generalities and Principles of Machine Learning
2. Lecture 2 (Wednesday 04 Oct. 2023):
Neural networks, deep learning, training
3. Lecture 3 (Tuesday 24 Oct. 2023):
Practical work

julien.brajard@nersc.no

References

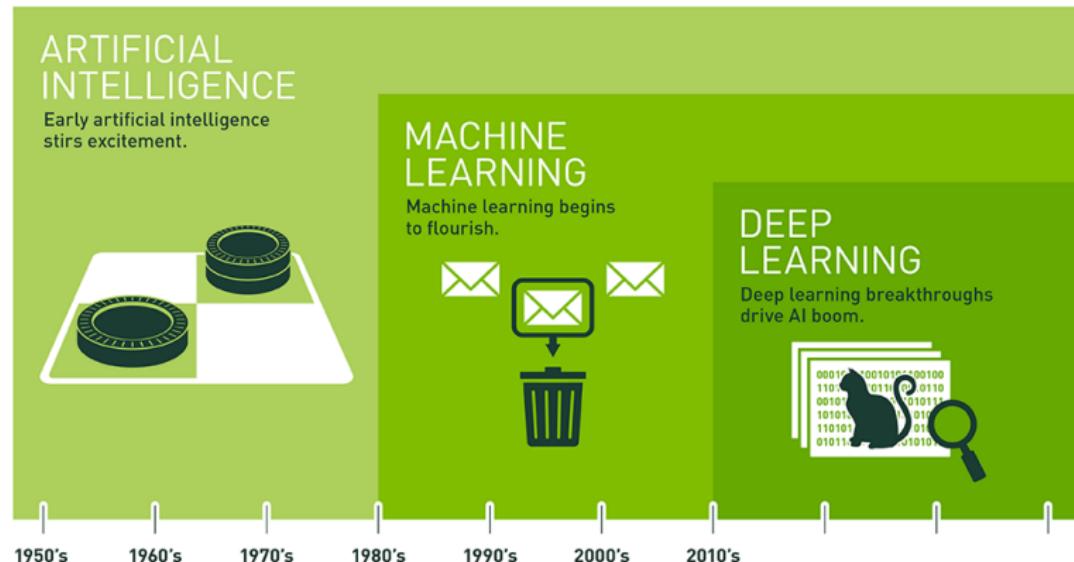
-  Francois Chollet.
Deep learning with Python.
Simon and Schuster, 2021.
-  Ian Goodfellow, Yoshua Bengio, and Aaron Courville.
Deep Learning.
MIT Press, 2016.
<http://www.deeplearningbook.org>.
-  Jake VanderPlas.
Python Data Science Handbook: Essential Tools for Working with Data.
O'Reilly Media, Inc., 1st edition, 2016.
<https://jakevdp.github.io/PythonDataScienceHandbook/>.

Table of contents

1. Introduction
2. Generalities on Machine Learning
3. Model selection/validation
4. Steps of a machine learning process
5. Feature processing
6. L1/L2 regularization

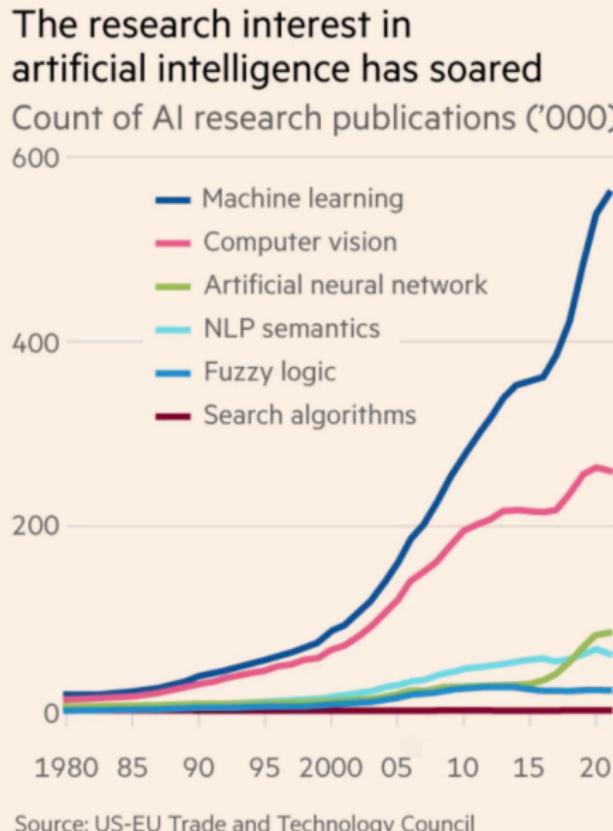
Introduction

Scope of the lecture: Machine Learning

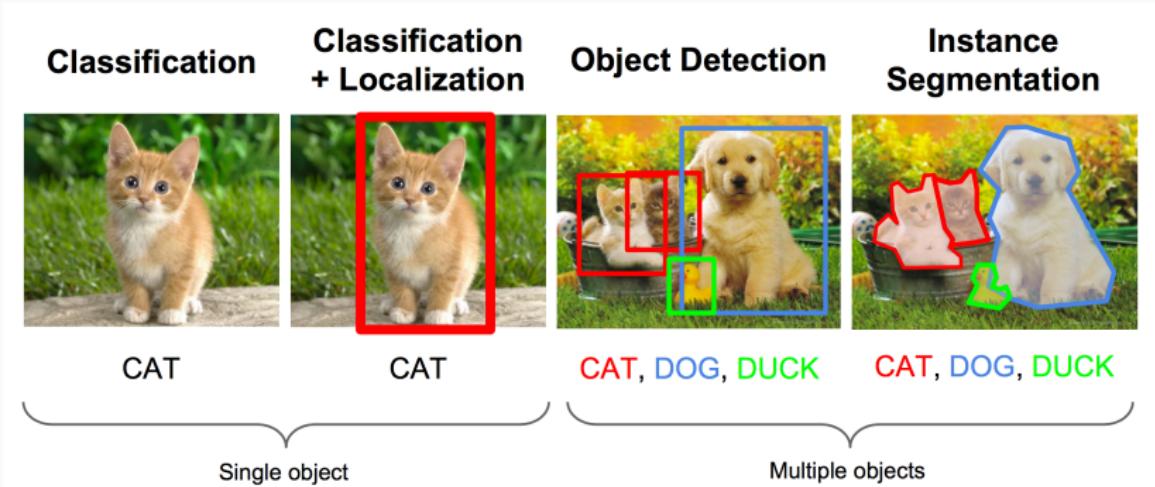


Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

A (very) active field

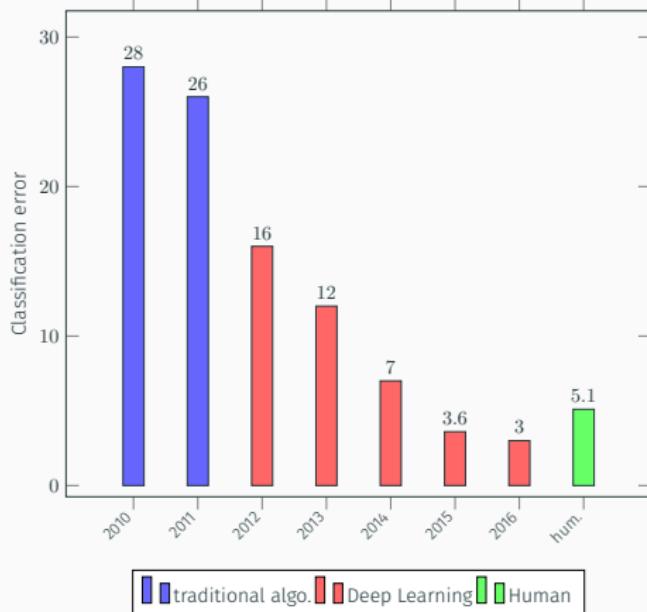


Example 1: Computer Vision



Li, Karpathy and Johnson, 2016, Stanford CS231n course

Example 1: Computer Vision



Deep learning architectures were based on Convolutional Neural Networks (CNN).

Example 2: Playing Games

- 1997: Deep Blue defeats Kasparov at Chess.
- 2016: AlphaGo's victory again Lee Sedol at Go.
- 2017: AlphaGo Zero learns how to play Go only by playing against itself. It outperformed previous AlphaGo version
(Reinforcement learning)
- 2017: DeepStack beats professional human poker players.



Example 3: Text to image

Objective: Generate an image from a textual description.

- 2021: DALL·E
- 2022: Midjourney
- 2022: Stable diffusion

Example 3: Text to image

Objective: Generate an image from a textual description.

- 2021: DALL·E
- 2022: Midjourney
- 2022: Stable diffusion



photography style with a 22mm lens of a handsome redhead biker man with a short beard.

Text to image - Progress in free tools

A test on:

<https://deepai.org/machine-learning-model/text2img>

Prompt: “*students attending a machine learning course*”

Text to image - Progress in free tools

A test on:

<https://deepai.org/machine-learning-model/text2img>

Prompt: “*students attending a machine learning course*”

Result in Oct. 2022



Text to image - Progress in free tools

A test on:

<https://deepai.org/machine-learning-model/text2img>

Prompt: “*students attending a machine learning course*”

Result in Oct. 2022



Result in Oct. 2023



Example 4: Large language models (LLM)

Examples: GPT-4 (OpenAI), LLaMA (Meta), PaLM 2 (Google)

How I tried to delegate my job



I have to give 6 hours of an introduction course to machine learning to master students. The objective is to understand the main concepts of machine learning from a practical point of view. Can you suggest an outline?



Certainly! Here's a suggested outline for a 6-hour introduction to machine learning course for master's students:



Day 1: Introduction to Machine Learning

Duration: 2 hours

Session 1: Introduction to Machine Learning

- What is Machine Learning?
- Types of Machine Learning: Supervised, Unsupervised, and Reinforcement Learning
- Real-world Applications of Machine Learning

Session 2: Data and Preprocessing

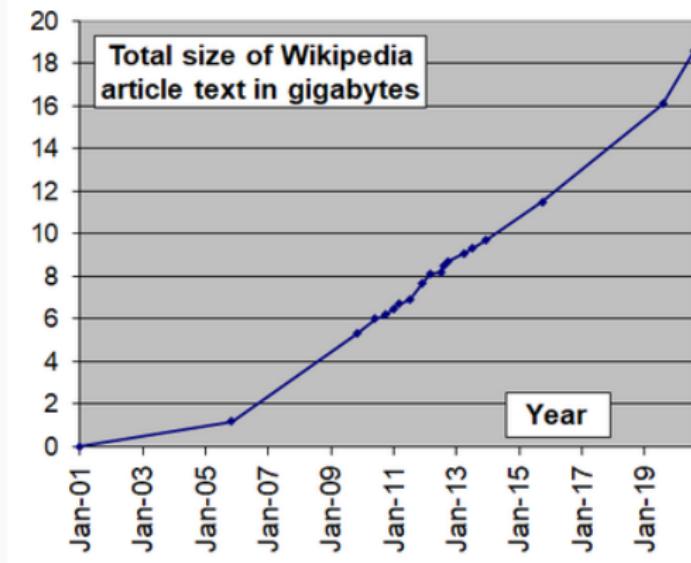
- Importance of Data in Machine Learning
- Data Types: Structured and Unstructured Data
- Data Preprocessing: Cleaning, Transformation, and Feature Engineering

Session 3: Supervised Learning

- Supervised Learning Overview
- Classification vs. Regression
- Key Algorithms: Linear Regression, Logistic Regression, and Decision Trees
- Hands-on Exercise: Building a Simple Supervised Learning Model

Reasons for these recent achievements?

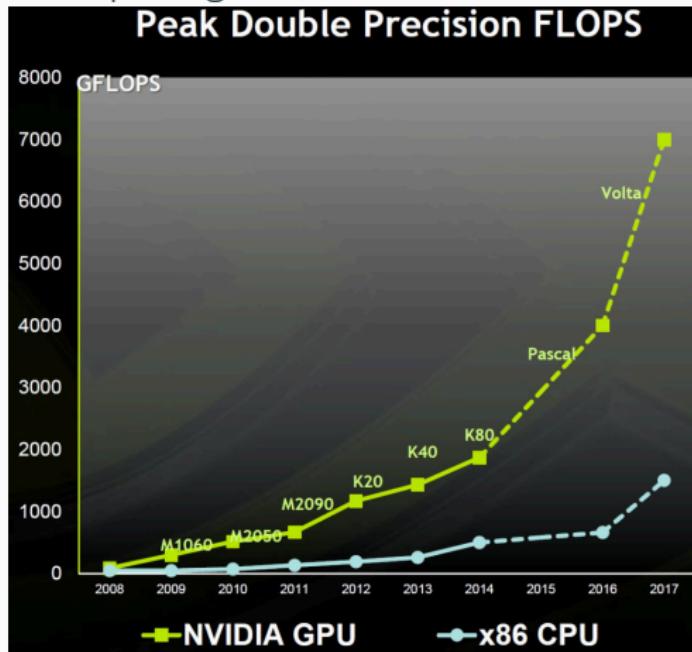
- Increasing of the datasets in size and quality



source: *Wikipedia*

Reasons for these recent achievements?

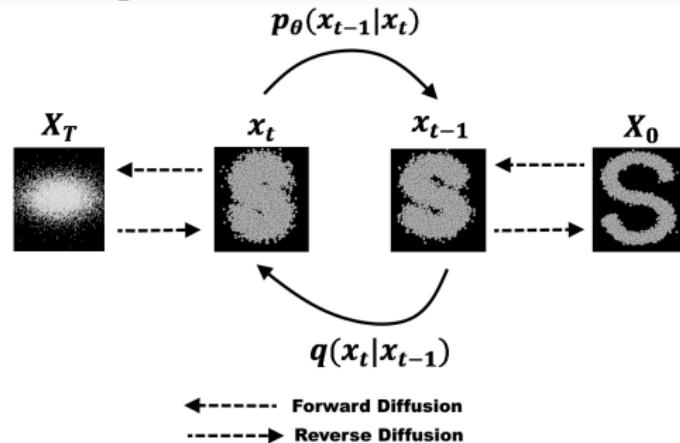
- Increasing of the datasets in size and quality
- Progress in computing resources.



source: NVIDIA

Reasons for these recent achievements?

- Increasing of the datasets in size and quality
- Progress in computing resources.
- Scientific research on new algorithms (e.g adapted to image processing)



Reasons for these recent achievements?

- Increasing of the datasets in size and quality
- Progress in computing resources.
- Scientific research on new algorithms (e.g adapted to image processing)
- Very efficient software (GPU, cloud computing, automatic differentiation, ...)



Reasons for these recent achievements?

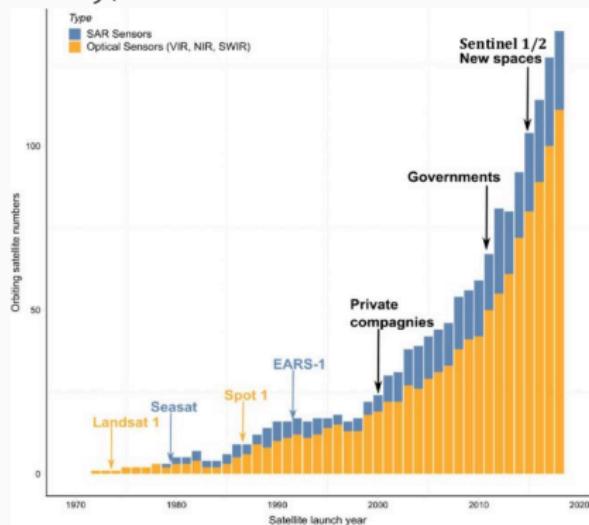
- Increasing of the datasets in size and quality
- Progress in computing resources.
- Scientific research on new algorithms (e.g adapted to image processing)
- Very efficient software (GPU, cloud computing, automatic differentiation, ...)
- Free software and open data culture.



Apply Machine-Learning to physical (Earth-system) modelling?

Why is it a good idea?

- A increasing number of geophysical data (one spatial mission: 24 TB/day)

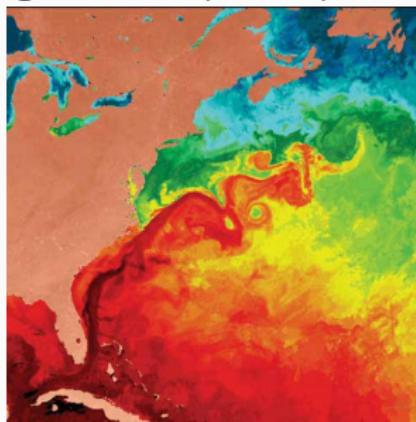


Earth System observavtion satellites

Apply Machine-Learning to physical (Earth-system) modelling?

Why is it a good idea?

- A increasing number of geophysical data (one spatial mission: 24 TB/day)
- Data with highly significant spatial patterns



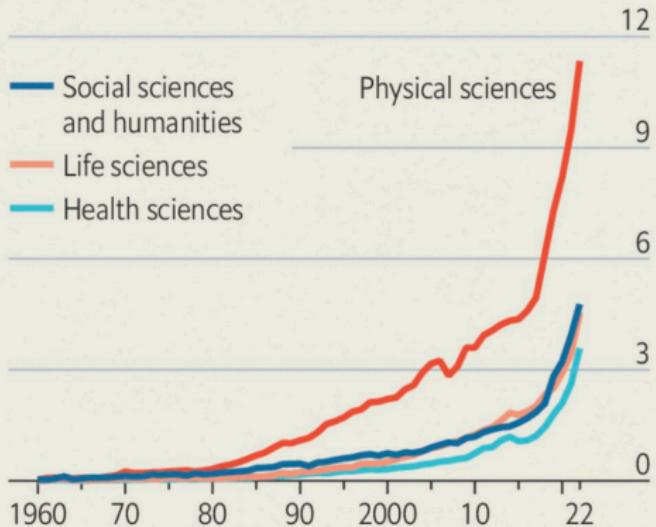
Sea Surface temperature of the gulf stream

source: *Talley (2000)*

AI in physical sciences

Machines learning

AI-related scholarly publications*, % of total
By main research field



*Peer-reviewed research from the Lens database

Source: CSIRO

Why is physical modelling specific?

NASDAQ Composite stock market index over the last 10 years

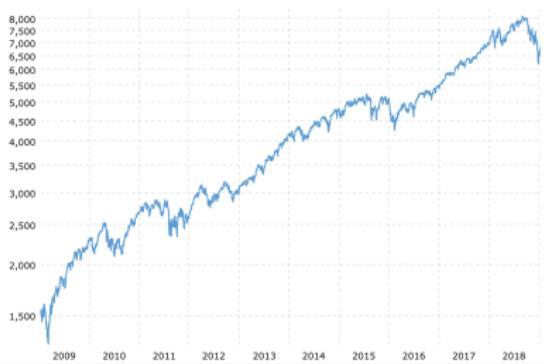
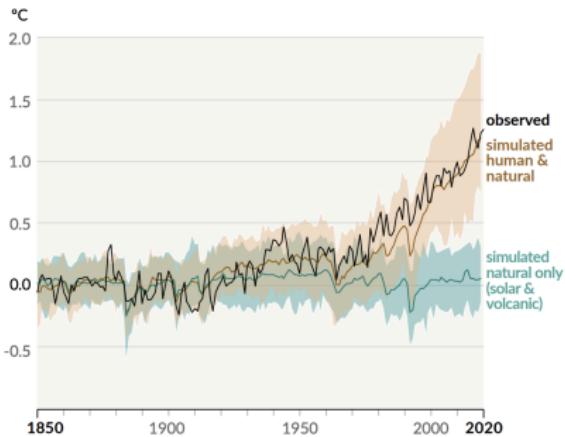


Figure 1: IPCC, AR6, WG1

b) Change in global surface temperature (annual average) as observed and simulated using **human & natural** and **only natural** factors (both 1850-2020)



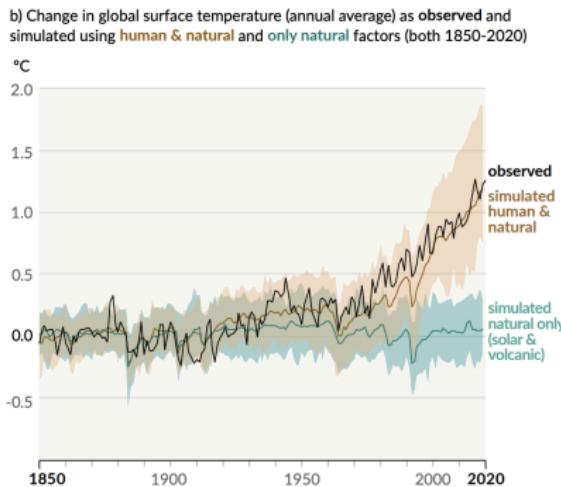
Why is physical modelling specific?

NASDAQ Composite stock market index over the last 10 years



Mostly unknown dynamical processes

Figure 1: IPCC, AR6, WG1



Mostly known dynamical processes (based on physical principles)

What about data assimilation?

Machine learning and data assimilation are closely linked.

Some references:

- Geer, A.J., 2021. Learning earth system models from observations: machine learning or data assimilation?. *Philosophical Transactions of the Royal Society A*, 379(2194)
- Brajard et al. 2019. Connections between data assimilation and machine learning to emulate a numerical model. *Proceedings of the 9th International Workshop on Climate informatics*
- Bocquet et al. 2019. Data assimilation as a learning tool to infer ordinary differential equation representations of dynamical models. *Nonlinear processes in geophysics*. 26(3).
- Abarbanel, H.D., Rozdeba, P.J. and Shirman, S., 2018. Machine learning: Deepest learning as statistical data assimilation problems. *Neural computation*, 30(8).

Generalities on Machine Learning

What is this about ?

Can we extract knowledge, make some predictions,
determine a "model" using this large amount of data ?

What is this about ?

Can we extract knowledge, make some predictions,
determine a "model" using this large amount of data ?



→ Digit ∈ {0, ..., 9}

Base of images

What is this about ?

Can we extract knowledge, make some predictions,
determine a "model" using this large amount of data ?



→ Digit ∈ {0, ..., 9}

Base of images

- From high dimensional data (thousands to millions dimensions) to reduced dimensional data (less than 100)
- From disorganized data to organized information
- Can we teach a machine how to do that ?

Two types of settings

1. **Supervised learning:** we have a set of labeled data with examples of targets.

Two types of settings

1. **Supervised learning**: we have a set of labeled data with examples of targets.
2. **Unsupervised learning**: we only have unlabeled data, we have no examples of what we want to obtain. We want to extract a "useful" representation of these data or some coherent categories.

Two types of settings

1. **Supervised learning**: we have a set of labeled data with examples of targets.
2. **Unsupervised learning**: we only have unlabeled data, we have no examples of what we want to obtain. We want to extract a "useful" representation of these data or some coherent categories.
 - Determine typical behaviors of clients in a supermarket knowing what they have bought.

Two types of settings

1. **Supervised learning**: we have a set of labeled data with examples of targets.
2. **Unsupervised learning**: we only have unlabeled data, we have no examples of what we want to obtain. We want to extract a "useful" representation of these data or some coherent categories.
 - Determine typical behaviors of clients in a supermarket knowing what they have bought.
3. **Semi-Supervised Learning**: Only a few subsets of the data are labeled

Two types of settings

1. **Supervised learning:** we have a set of labeled data with examples of targets.
2. **Unsupervised learning:** we only have unlabeled data, we have no examples of what we want to obtain. We want to extract a "useful" representation of these data or some coherent categories.
 - Determine typical behaviors of clients in a supermarket knowing what they have bought.
3. **Semi-Supervised Learning:** Only a few subsets of the data are labeled
4. **Reinforcement Learning:** We can initiate and observe the interaction of an agent with its environment. We want to optimize the behavior of the agent.

Two types of settings

1. **Supervised learning:** we have a set of labeled data with examples of targets.
2. **Unsupervised learning:** we only have unlabeled data, we have no examples of what we want to obtain. We want to extract a "useful" representation of these data or some coherent categories.
 - Determine typical behaviors of clients in a supermarket knowing what they have bought.
3. **Semi-Supervised Learning:** Only a few subsets of the data are labeled
4. **Reinforcement Learning:** We can initiate and observe the interaction of an agent with its environment. We want to optimize the behavior of the agent.
 - Playing a chess game.

Two classes of Machine Learning problems

1. **Regression:** Determination of a quantitative variable from a set of data
 - The price of a building from various predictors (Surface, ...)
 - A physical value (Temperature, humidity, ...) in the future knowing the past
 - ...

Two classes of Machine Learning problems

1. **Regression:** Determination of a quantitative variable from a set of data
 - The price of a building from various predictors (Surface, ...)
 - A physical value (Temperature, humidity, ...) in the future knowing the past
 - ...
2. **Classification:** Determination of a class
 - A digit from a image
 - Identification of the content of an image
 - ...

A Machine

$$y = \mathcal{M}(x, \theta)$$

- x : input
- y : output
- \mathcal{M} : a model (named "machine")
- θ : parameters of the model \mathcal{M} .

Machine learning consists in optimizing θ using a set of data.
This is the training process.

The Machine Learning recipe

A Machine

$$y = \mathcal{M}(x, \theta)$$

What are **the ingredients?**

The Machine Learning recipe

A Machine

$$y = \mathcal{M}(x, \theta)$$

What are **the ingredients?**

- Some **data**
 - x, y : supervised learning
 - only x : unsupervised learning
 - x and some subset of y : semi-supervised learning

The Machine Learning recipe

A Machine

$$y = \mathcal{M}(x, \theta)$$

What are **the ingredients?**

- Some **data**
 - x, y : supervised learning
 - only x : unsupervised learning
 - x and some subset of y : semi-supervised learning
- An **objective**
 - y is quantitative: regression
 - y is a class: classification

The Machine Learning recipe

A Machine

$$y = \mathcal{M}(x, \theta)$$

What are **the ingredients?**

- Some **data**
 - x, y : supervised learning
 - only x : unsupervised learning
 - x and some subset of y : semi-supervised learning
- An **objective**
 - y is quantitative: regression
 - y is a class: classification
- A computational architecture (the **machine**)
 - linear
 - non-linear
 - neural networks, random forest, ...

The Machine Learning recipe

A Machine

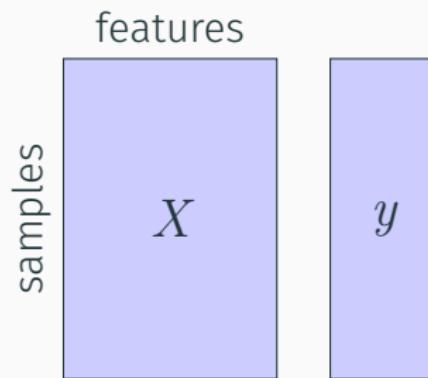
$$y = \mathcal{M}(x, \theta)$$

What are **the ingredients?**

- Some **data**
 - x, y : supervised learning
 - only x : unsupervised learning
 - x and some subset of y : semi-supervised learning
- An **objective**
 - y is quantitative: regression
 - y is a class: classification
- A computational architecture (the **machine**)
 - linear
 - non-linear
 - neural networks, random forest, ...
- A **learning** process
 - Estimation of θ

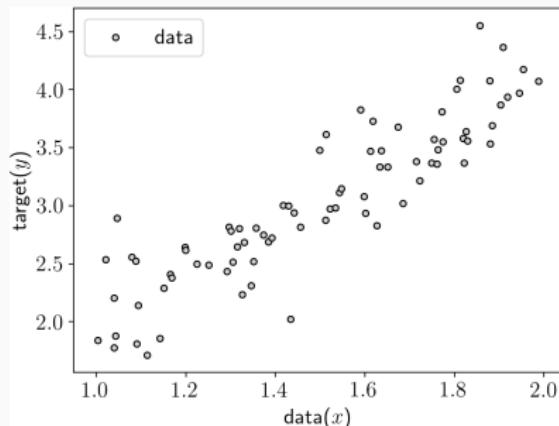
Multidimensional data

Generally, we have multidimensional data X and a one-dimensional target y .



An illustration

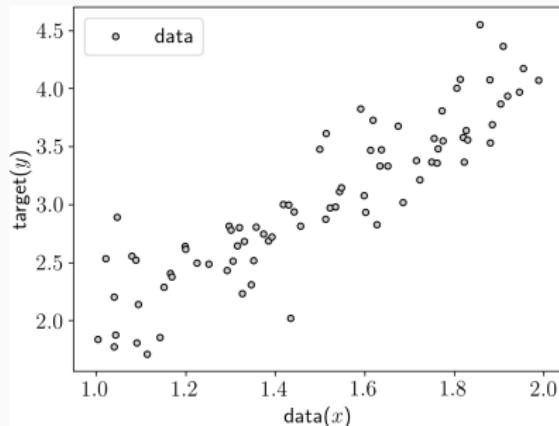
- Some Data



- y is known: supervised learning
- y is quantitative: regression

An illustration

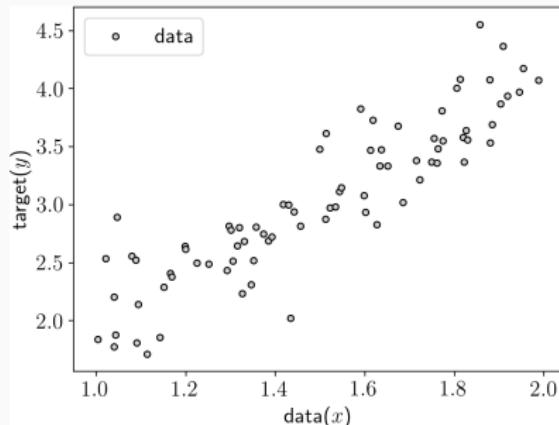
- Some Data



- y is known: supervised learning
- y is quantitative: regression
- An **Objective**: Estimate \hat{y} from x by minimizing $(\hat{y} - y)^2$
(Least-square objective)

An illustration

- Some Data

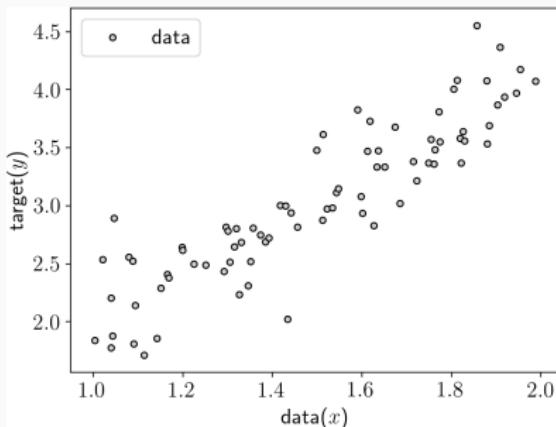


- A model: $y = \theta_1 X + \theta_0$ (linear)

- y is known: supervised learning
- y is quantitative: regression
- An Objective: Estimate \hat{y} from x by minimizing $(\hat{y} - y)^2$
(Least-square objective)

An illustration

- Some Data

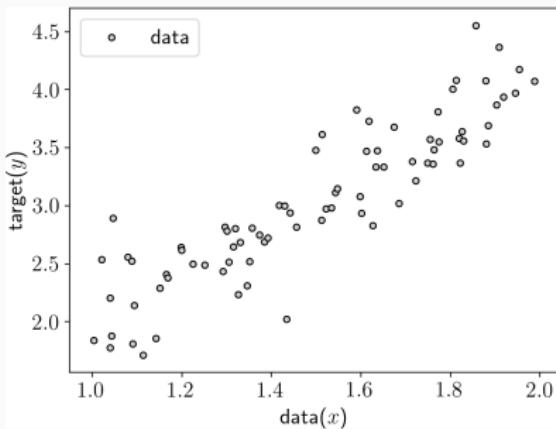


- A **model**: $y = \theta_1 X + \theta_0$ (linear)
- A **learning process**:
$$\theta = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T y$$

- y is known: supervised learning
- y is quantitative: regression
- An **Objective**: Estimate \hat{y} from x by minimizing $(\hat{y} - y)^2$
(Least-square objective)

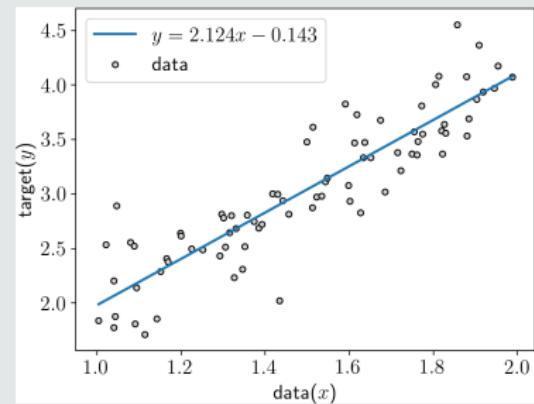
An illustration

- Some Data



- A **model**: $y = \theta_1 X + \theta_0$ (linear)
- A **learning process**:
$$\theta = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T y$$

Result



- y is known: supervised learning
- y is quantitative: regression
- An **Objective**: Estimate \hat{y} from x by minimizing $(\hat{y} - y)^2$
(Least-square objective)

Model selection/validation

Choice of the model

Polynomial regression

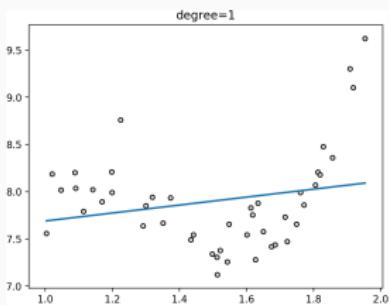
$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_d x^d = \sum_{i=0}^d \theta_i X^i$$

Choice of the model

Polynomial regression

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_d x^d = \sum_{i=0}^d \theta_i X^i$$

degree = 1 (linear)

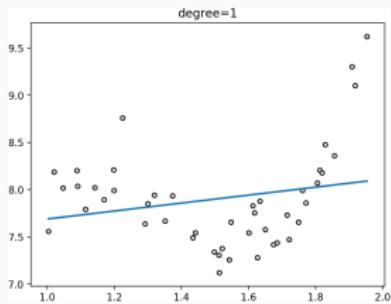


Choice of the model

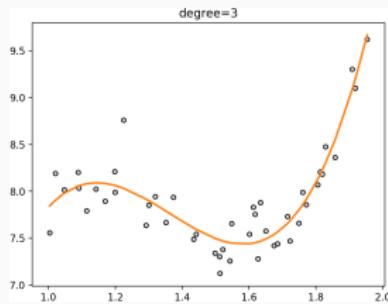
Polynomial regression

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_d x^d = \sum_{i=0}^d \theta_i X^i$$

degree = 1 (linear)



degree = 3

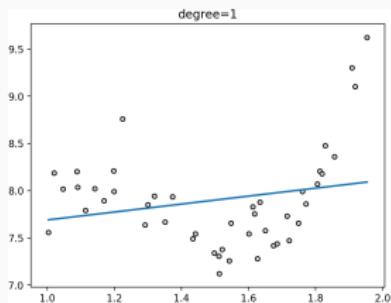


Choice of the model

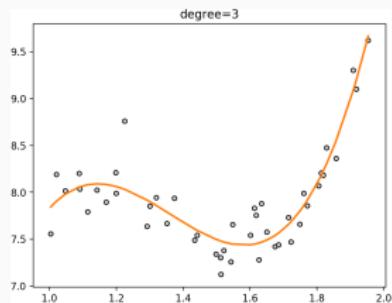
Polynomial regression

$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_d x^d = \sum_{i=0}^d \theta_i X^i$$

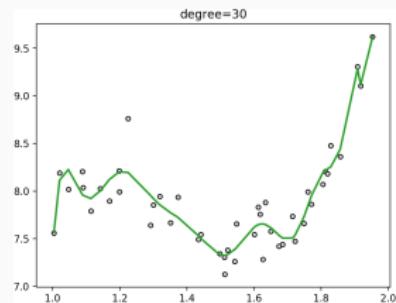
degree = 1 (linear)



degree = 3



degree = 30



What is the best model?

Train/Validation split

The idea

Evaluate a score on an independent dataset

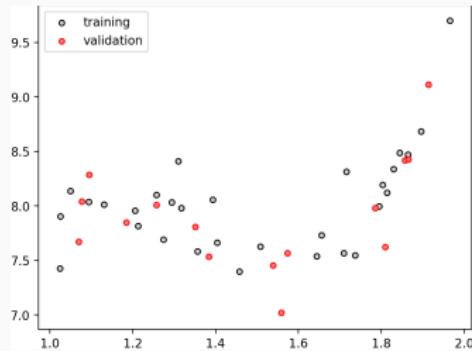
Train/Validation split

The idea

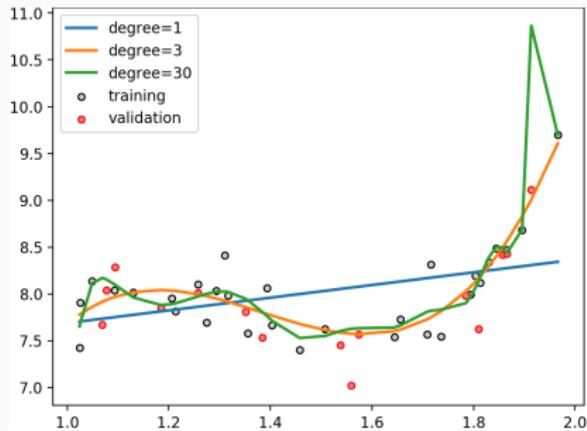
Evaluate a score on an independent dataset

In our example we can randomly divide (X, y) in two datasets:

- The training dataset X_{train}, y_{train} used to fit the model.
- The validation dataset X_{val}, y_{val} used to compute the score (e.g., correlation, mean-squared error)



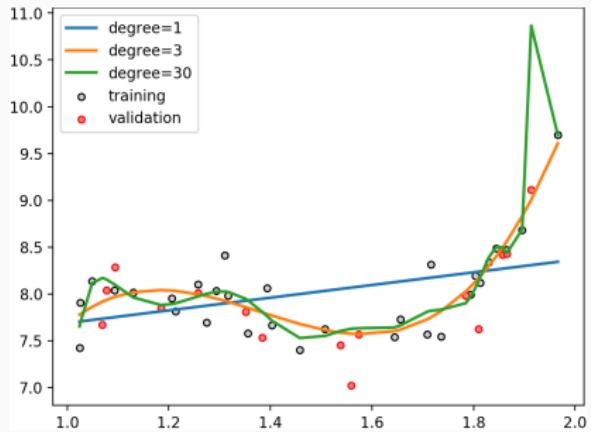
Choice of the model



Score: Mean Square Error (MSE)

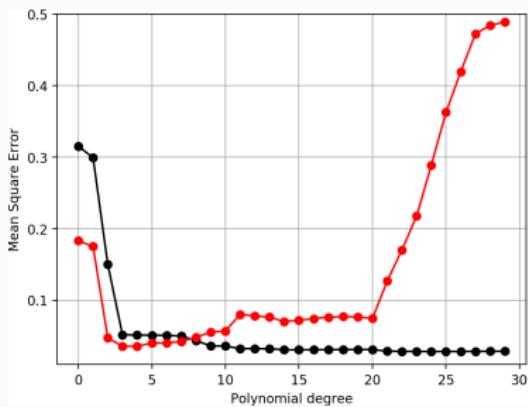
Deg.	Train Score	Val. Score
1	0.17	0.23
3	0.045	0.062
30	0.035	0.27

Choice of the model



Score: Mean Square Error (MSE)

Deg.	Train Score	Val. Score
1	0.17	0.23
3	0.045	0.062
30	0.035	0.27

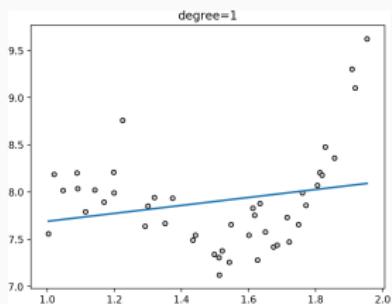


Choice of the model

Polynomial regression

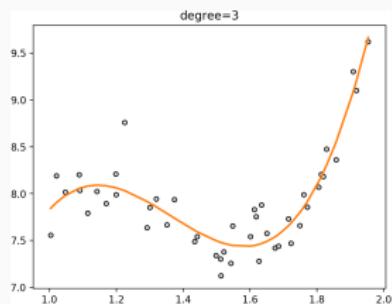
$$y = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_d x^d = \sum_{i=0}^d \theta_i X^i$$

degree = 1 (linear)



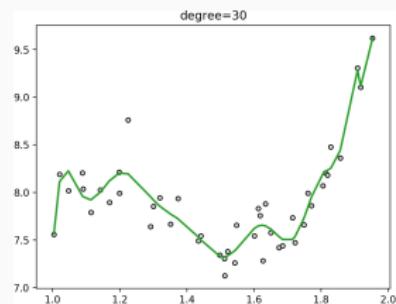
underfitting

degree = 3



good fit

degree = 30



overfitting

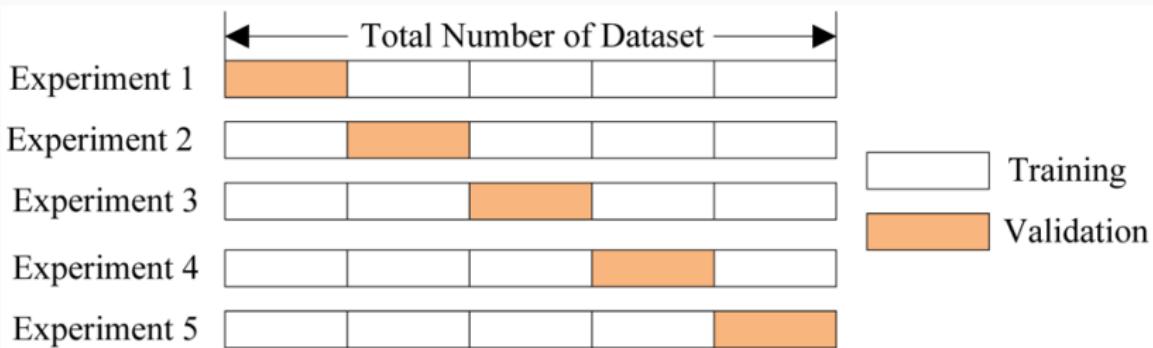
Drawbacks

- drastically reduces the number of samples that can be used for learning the model
- Results can depend on a particular random choice for the pair of (train, validation) sets.

More Robust: cross validation

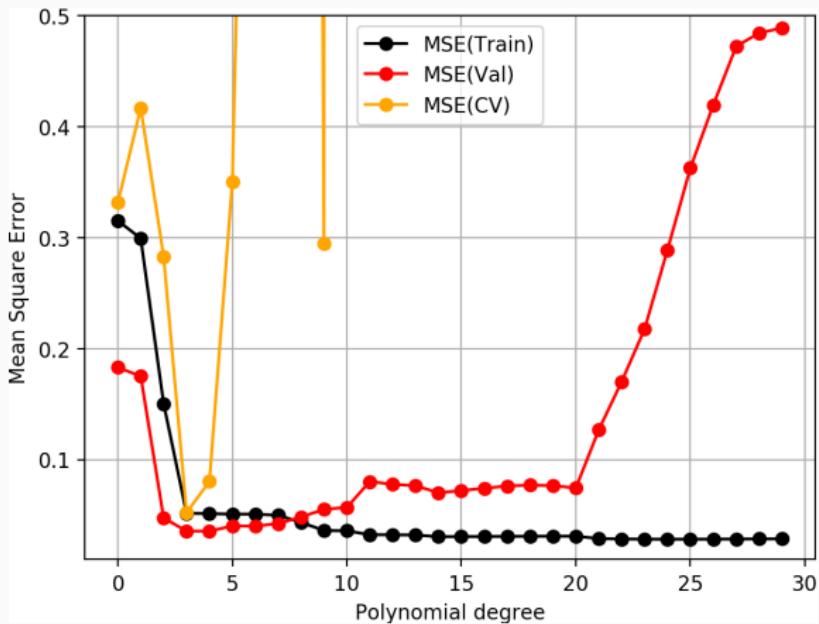
The idea

- Dividing the data in n folds,
- Learning n model (each time with a different training set),
- Compute the mean score over n validation set.



Cross-Validation

Fold	MSE
1	0.052
2	0.043
3	0.137
4	0.025
5	0.048
6	0.144
7	0.011
8	0.025
9	0.010
10	0.028
Mean	0.05



Wrapping up

1. When applying machine learning techniques there are **hyperparameters** to be determined (e.g., degree of the polynomial in polynomial regression).

Wrapping up

1. When applying machine learning techniques there are **hyperparameters** to be determined (e.g., degree of the polynomial in polynomial regression).
2. These **hyperparameters** can be determined by splitting the data into training/validation or by cross-validation.

Wrapping up

1. When applying machine learning techniques there are **hyperparameters** to be determined (e.g., degree of the polynomial in polynomial regression).
2. These **hyperparameters** can be determined by splitting the data into training/validation or by cross-validation.
3. But then... the validation set was used to determine the best machine learning process

Wrapping up

1. When applying machine learning techniques there are **hyperparameters** to be determined (e.g., degree of the polynomial in polynomial regression).
2. These **hyperparameters** can be determined by splitting the data into training/validation or by cross-validation.
3. But then... the validation set was used to determine the best machine learning process
4. To evaluate independently the performance of our model, we should compute the score on a **third independent dataset: The test dataset**.

Wrapping up

1. When applying machine learning techniques there are **hyperparameters** to be determined (e.g., degree of the polynomial in polynomial regression).
2. These **hyperparameters** can be determined by splitting the data into training/validation or by cross-validation.
3. But then... the validation set was used to determine the best machine learning process
4. To evaluate independently the performance of our model, we should compute the score on a **third independent dataset: The test dataset**.

Wrapping up

1. When applying machine learning techniques there are **hyperparameters** to be determined (e.g., degree of the polynomial in polynomial regression).
2. These **hyperparameters** can be determined by splitting the data into training/validation or by cross-validation.
3. But then... the validation set was used to determine the best machine learning process
4. To evaluate independently the performance of our model, we should compute the score on a **third independent dataset: The test dataset.**

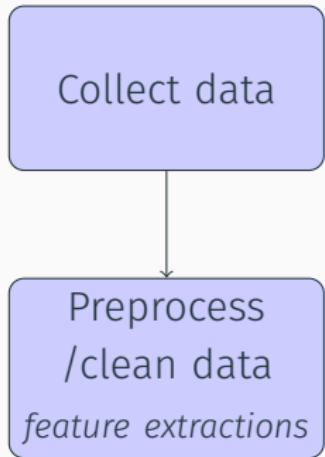
More on that in next lecture...

Steps of a machine learning process

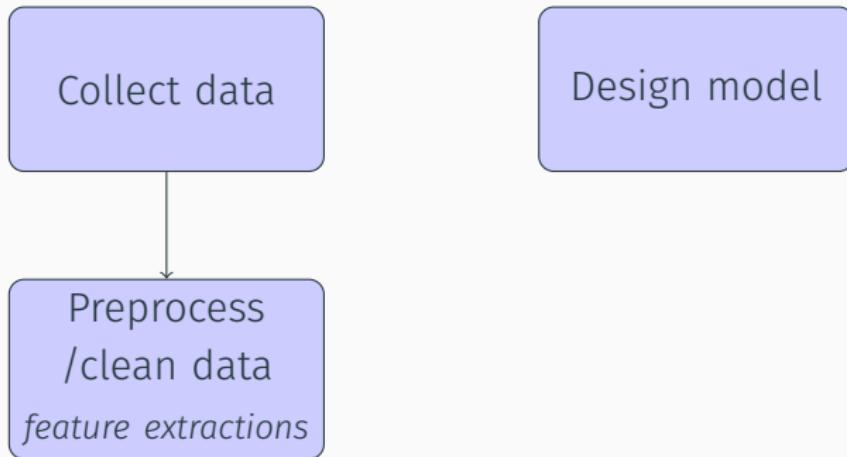
Steps

Collect data

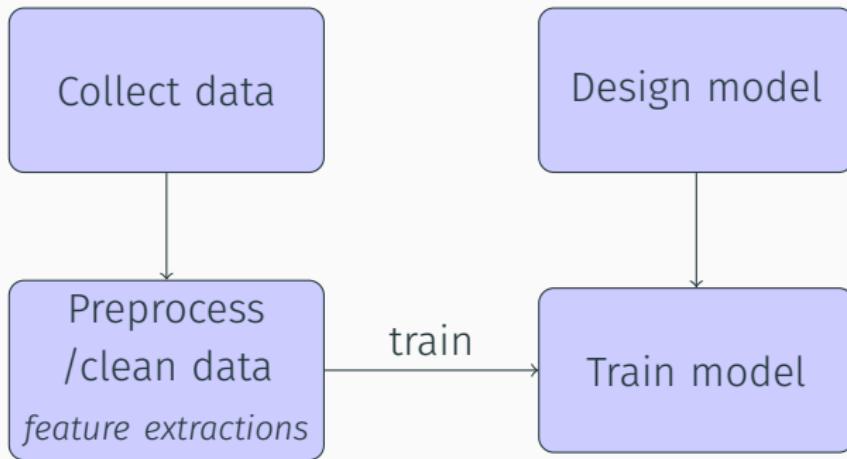
Steps



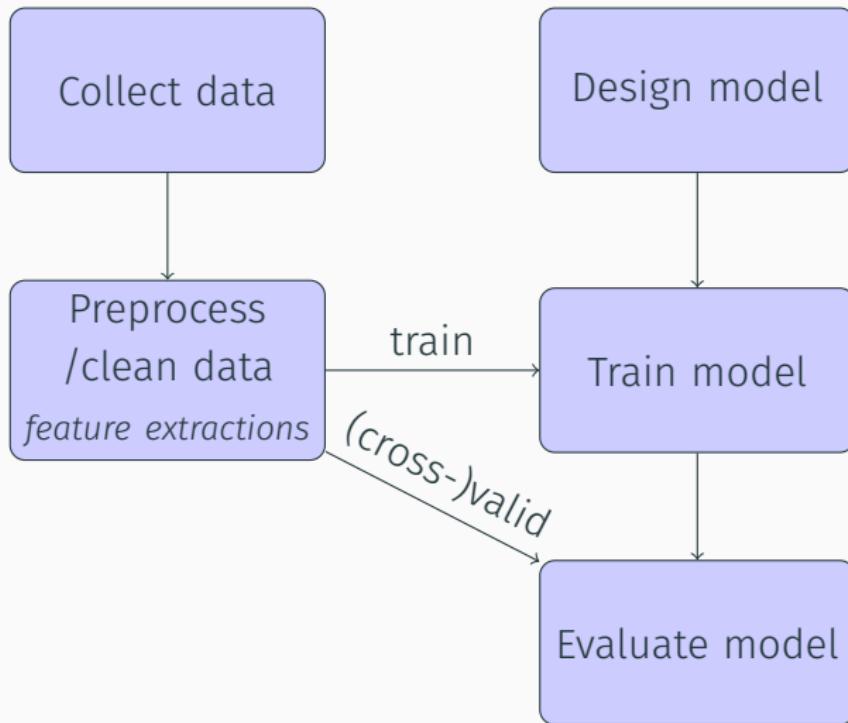
Steps



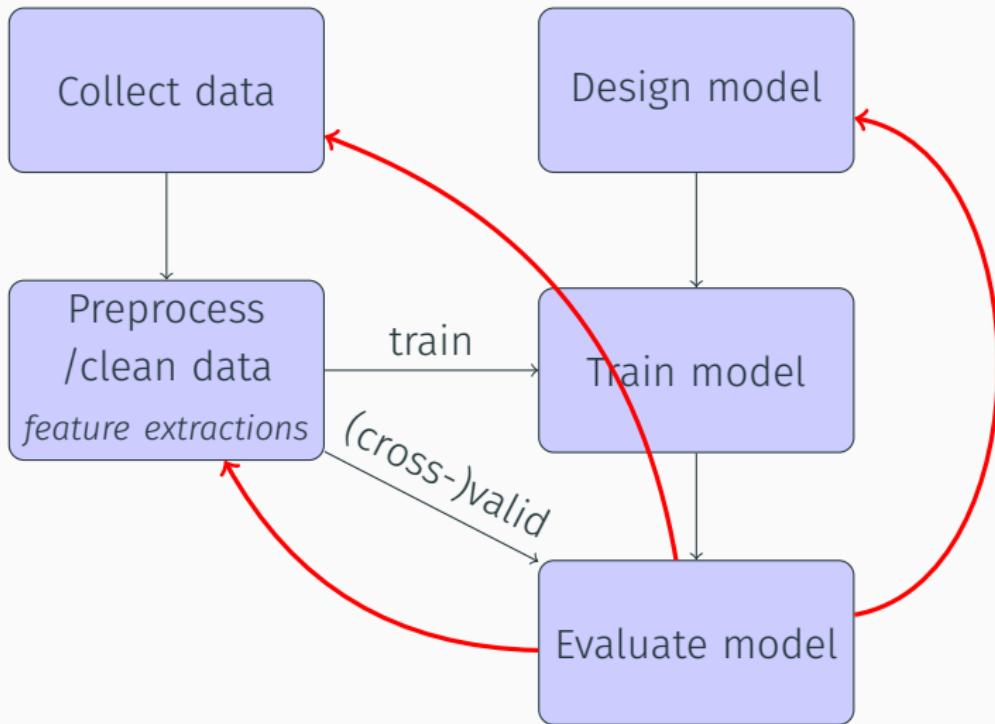
Steps



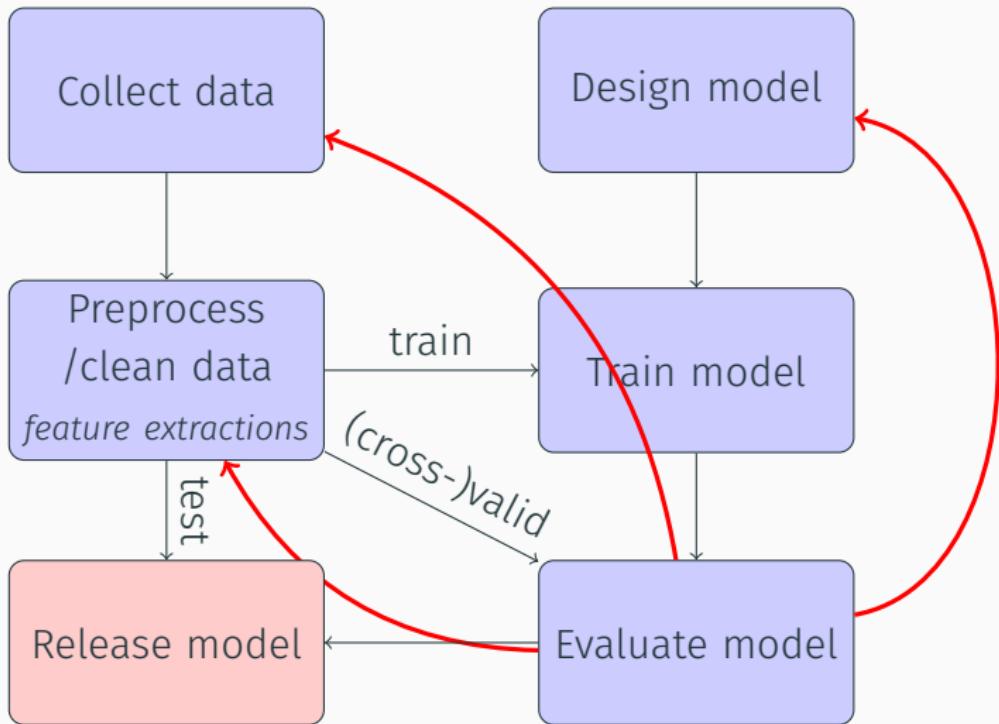
Steps



Steps



Steps



In summary

From one dataset, 3 sub-datasets have to be extracted:

- A training dataset
- A validation dataset

Can be done iteratively in a cross-validation procedure.

Some parameters of the model (e.g. polynomial order in a polynomial regression) were determined from the validation dataset.

- A test dataset (independent from the two others) to estimate the final performance of the model.

Feature processing

Type of features

- quantitative/continuous features (e.g. distance to employment centers, temperature)

Type of features

- quantitative/continuous features (e.g. distance to employment centers, temperature)
- ordinal/discrete features (e.g. number of rooms, category of a hurricane)

Type of features

- quantitative/continuous features (e.g. distance to employment centers, temperature)
- ordinal/discrete features (e.g. number of rooms, category of a hurricane)
- categorical features (e.g. name of the neighborhood, name of the ocean)

Type of features

- quantitative/continuous features (e.g. distance to employment centers, temperature)
- ordinal/discrete features (e.g. number of rooms, category of a hurricane)
- categorical features (e.g. name of the neighborhood, name of the ocean)

Type of features

- quantitative/continuous features (e.g. distance to employment centers, temperature)
- ordinal/discrete features (e.g. number of rooms, category of a hurricane)
- categorical features (e.g. name of the neighborhood, name of the ocean)

Encoding of the features?

Feature encoding

Type	Examples		Encoding
Quantitative	distance	{1.2, 2.3, 0.1}	{1.2, 2.3, 0.1}
Ordinal	rooms	{2, 3, 4}	{2, 3, 4}
Qualitative	Ocean	{Atlantic, Indian, Pacific}	{[1, 0, 0], [0, 1, 0], [0, 0, 1]}

Feature encoding

Type	Examples		Encoding
Quantitative	distance	{1.2, 2.3, 0.1}	{1.2, 2.3, 0.1}
Ordinal	rooms	{2, 3, 4}	{2, 3, 4}
Qualitative	Ocean	{Atlantic, Indian, Pacific}	{[1, 0, 0], [0, 1, 0], [0, 0, 1]}

Qualitative variable: one-hot encoding.

- You must **not** encode qualitative features with integer 1, 2, 3, it would mean that Pacific > Indian > Atlantic.
- In sklearn there is a function that makes the one-hot encoding: `OneHotEncoder()`
- If the number of modalities (number of different features) is high, encoding qualitative feature produces a big-sized vector.

Embedding

A common way to deal with features with a lot of modalities:
Embedding

Principle of embedding

Let's consider a qualitative variable with n modalities,
represented by the n -dimensional binary vector \mathbf{x}

Embedding consists in representing this variable by a vector
 $\mathbf{v} \in \mathbb{R}^p, p << n$

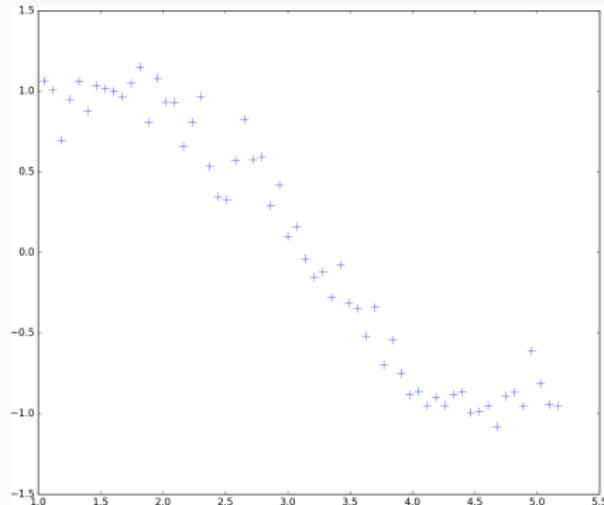
The embedding is represented by a matrix $\mathbf{M} \in \mathbb{R}^{n \times p}$ such as:

$$\mathbf{v} = \mathbf{M} \cdot \mathbf{x}$$

Coefficients of \mathbf{M} have to be optimized given an objective criteria (that depends on your problem)

L1/L2 regularization

Example of a non-linear relationship



An idea

We could take a polynomial hypothesis model:

$$h_{\theta}(x) = \theta_0 x^0 + \theta_1 x^1 + \dots + \theta_p x^p$$

Example

$\{(x_1, y_1), \dots, (x_n, y_n)\}$ is the learning dataset.

For a given polynomial degree p , parameters $\boldsymbol{\theta}$ are determined minimizing the least-mean square cost function:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2$$

with $h_{\boldsymbol{\theta}}(x) = \theta_0 x^0 + \theta_1 x^1 + \dots + \theta_p x^p$

- It can be determined using a gradient descent method

Example

$\{(x_1, y_1), \dots, (x_n, y_n)\}$ is the learning dataset.

For a given polynomial degree p , parameters $\boldsymbol{\theta}$ are determined minimizing the least-mean square cost function:

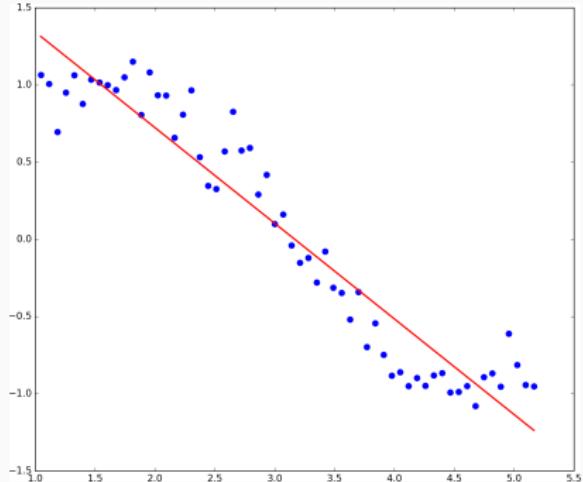
$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2$$

with $h_{\boldsymbol{\theta}}(x) = \theta_0 x^0 + \theta_1 x^1 + \dots + \theta_p x^p$

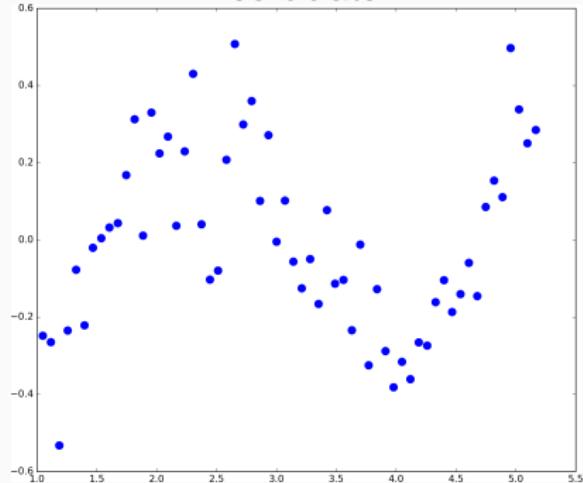
- It can be determined using a gradient descent method
- If the degree of the polynomial $p = 1$, it is a simple linear regression

A first result

$p = 1$ (linear regression)



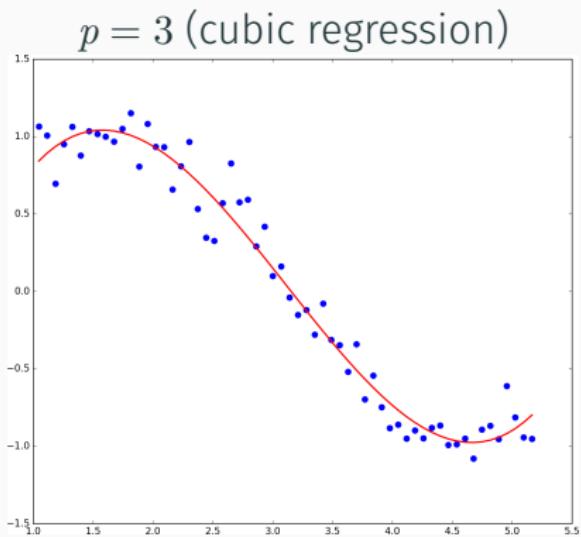
residuals



Prediction error

$$\text{err} = \frac{1}{n} \sum \text{res}^2 = 5.46e-2$$

Increasing the polynomial degree ?



Prediction error

$$\text{err} = \frac{1}{n} \sum \text{res}^2 = 1.80e-2$$

Is it different from the linear regression ?

Let's consider :

$$\boldsymbol{x} = \begin{pmatrix} 1 \\ x^1 \\ x^2 \\ \vdots \\ x^p \end{pmatrix}$$

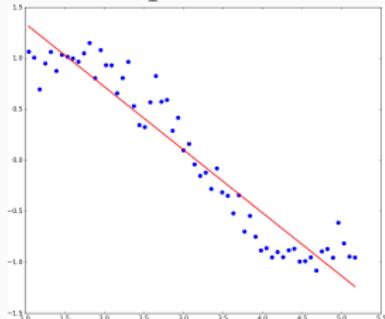
then

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 x^1 + \dots + \theta_p x^p = \boldsymbol{\theta}^T \boldsymbol{x}$$

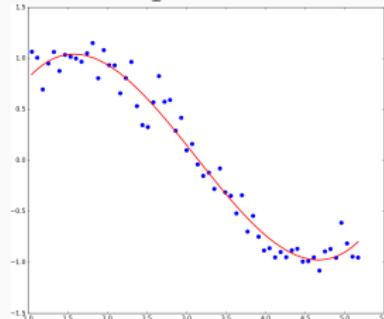
By extending a scalar predictor to a vector, polynomial regression is equivalent to linear regression.

Increasing the degree ?

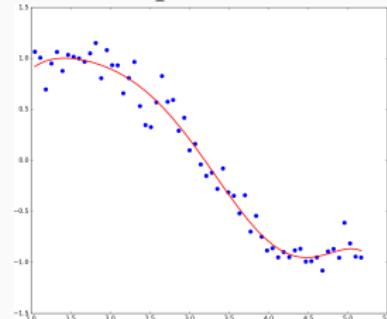
$p = 1$



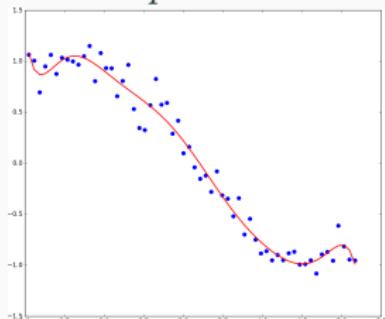
$p = 3$



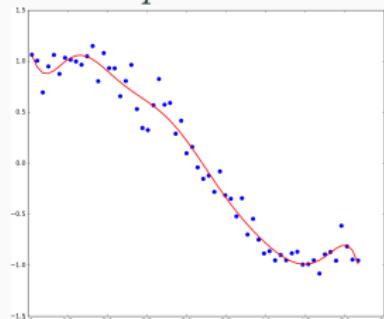
$p = 4$



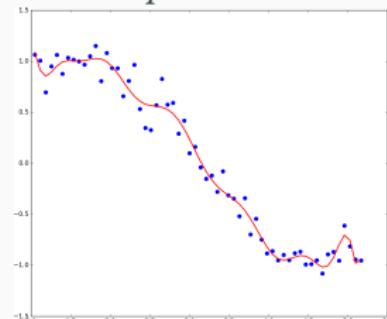
$p = 9$



$p = 12$



$p = 15$



Overfitting

When there are too many parameters to fit, the model can reproduce a random noise, it is called **overfitting**

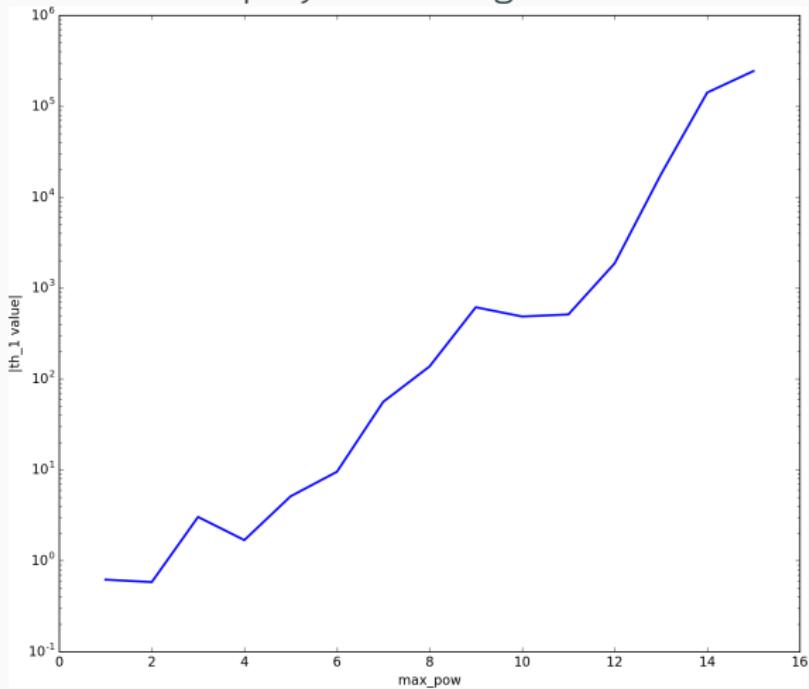
Overfitting

When there are too many parameters to fit, the model can reproduce a random noise, it is called **overfitting**

	rmse	th_0	th_1	th_2	th_3	th_4	th_5	th_6	th_7	th_8
max_pow_1	+5.47e-02	+1.96e+00	-6.20e-01	NaN						
max_pow_2	+5.46e-02	+1.91e+00	-5.83e-01	-5.96e-03	NaN	NaN	NaN	NaN	NaN	NaN
max_pow_3	+1.84e-02	-1.08e+00	+3.03e+00	-1.29e+00	+1.37e-01	NaN	NaN	NaN	NaN	NaN
max_pow_4	+1.80e-02	-2.66e-01	+1.69e+00	-5.32e-01	-3.57e-02	+1.39e-02	NaN	NaN	NaN	NaN
max_pow_5	+1.70e-02	+2.99e+00	-5.12e+00	+4.72e+00	-1.93e+00	+3.35e-01	-2.07e-02	NaN	NaN	NaN
max_pow_6	+1.65e-02	-2.80e+00	+9.52e+00	-9.71e+00	+5.23e+00	-1.55e+00	+2.33e-01	-1.36e-02	NaN	NaN
max_pow_7	+1.55e-02	+1.93e+01	-5.60e+01	+6.90e+01	-4.46e+01	+1.65e+01	-3.53e+00	+4.05e-01	-1.92e-02	NaN
max_pow_8	+1.53e-02	+4.32e+01	-1.37e+02	+1.84e+02	-1.33e+02	+5.77e+01	-1.53e+01	+2.42e+00	-2.10e-01	+7.68e-03
max_pow_9	+1.46e-02	+1.68e+02	-6.15e+02	+9.63e+02	-8.46e+02	+4.61e+02	-1.62e+02	+3.68e+01	-5.22e+00	+4.22e-01
max_pow_10	+1.46e-02	+1.38e+02	-4.86e+02	+7.26e+02	-5.96e+02	+2.93e+02	-8.75e+01	+1.45e+01	-8.06e-01	-1.38e-01
max_pow_11	+1.45e-02	-7.49e+01	+5.12e+02	-1.33e+03	+1.87e+03	-1.61e+03	+9.14e+02	-3.50e+02	+9.14e+01	-1.61e+01
max_pow_12	+1.45e-02	-3.39e+02	+1.87e+03	-4.42e+03	+6.01e+03	-5.25e+03	+3.12e+03	-1.30e+03	+3.84e+02	-8.03e+01
max_pow_13	+1.43e-02	+3.20e+03	-1.78e+04	+4.46e+04	-6.66e+04	+6.61e+04	-4.61e+04	+2.32e+04	-8.55e+03	+2.30e+03
max_pow_14	+1.31e-02	+2.38e+04	-1.41e+05	+3.79e+05	-6.10e+05	+6.57e+05	-5.03e+05	+2.82e+05	-1.17e+05	+3.66e+04
max_pow_15	+1.17e-02	-3.62e+04	+2.44e+05	-7.46e+05	+1.38e+06	-1.71e+06	+1.53e+06	-1.00e+06	+4.98e+05	-1.88e+05

High parameters values

Value of the parameters $|\theta_1|$ with respect with the degree of the polynomial regression



Regularization

The idea

The idea of regularization is to perform a regression minimizing a cost function that includes a term to penalize "big" values for the parameters:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha P(\boldsymbol{\theta})$$

Regularization

The idea

The idea of regularization is to perform a regression minimizing a cost function that includes a term to penalize "big" values for the parameters:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha P(\boldsymbol{\theta})$$

We consider two penalty terms:

- Ridge Regularization (L2): $P(\boldsymbol{\theta}) = \sum_{i=0}^p \theta_i^2$

Regularization

The idea

The idea of regularization is to perform a regression minimizing a cost function that includes a term to penalize "big" values for the parameters:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha P(\boldsymbol{\theta})$$

We consider two penalty terms:

- Ridge Regularization (L2): $P(\boldsymbol{\theta}) = \sum_{i=0}^p \theta_i^2$
- Lasso Regularization (L1): $P(\boldsymbol{\theta}) = \sum_{i=0}^p |\theta_i|$

Regularization

The idea

The idea of regularization is to perform a regression minimizing a cost function that includes a term to penalize "big" values for the parameters:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha P(\boldsymbol{\theta})$$

We consider two penalty terms:

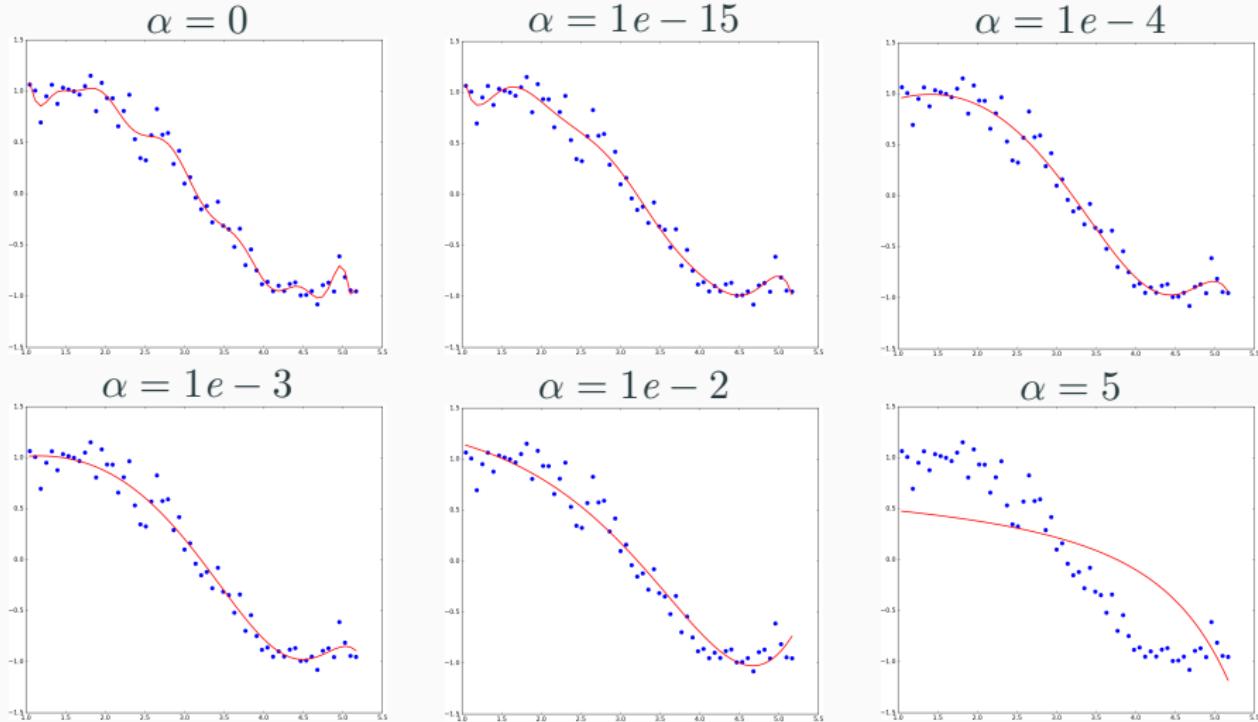
- Ridge Regularization (L2): $P(\boldsymbol{\theta}) = \sum_{i=0}^p \theta_i^2$
- Lasso Regularization (L1): $P(\boldsymbol{\theta}) = \sum_{i=0}^p |\theta_i|$
- Elastic Net combines both regularizations

Ridge regression (L2)

Ridge regression is a linear regression with a Ridge regularization:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha \sum_{i=0}^p \theta_i^2$$

Results for $p = 15$ and varying α



Determination of the parameters in the ridge regression

Considering the cost function J:

$$J(\boldsymbol{\theta}) = J_{lms}(\boldsymbol{\theta}) + \alpha \sum_{i=0}^p \theta_i^2$$

In a gradient algorithm, update of the parameters:

$$\theta_i^{k+1} = \theta_i^k - \nu \cdot \left(\frac{\partial J_{lms}}{\partial \theta_i} - 2\alpha \theta_i^k \right)$$

So the update rule is:

$$\theta_i^{k+1} = \theta_i^k (1 - 2\nu\alpha) - \Delta_{lms}$$

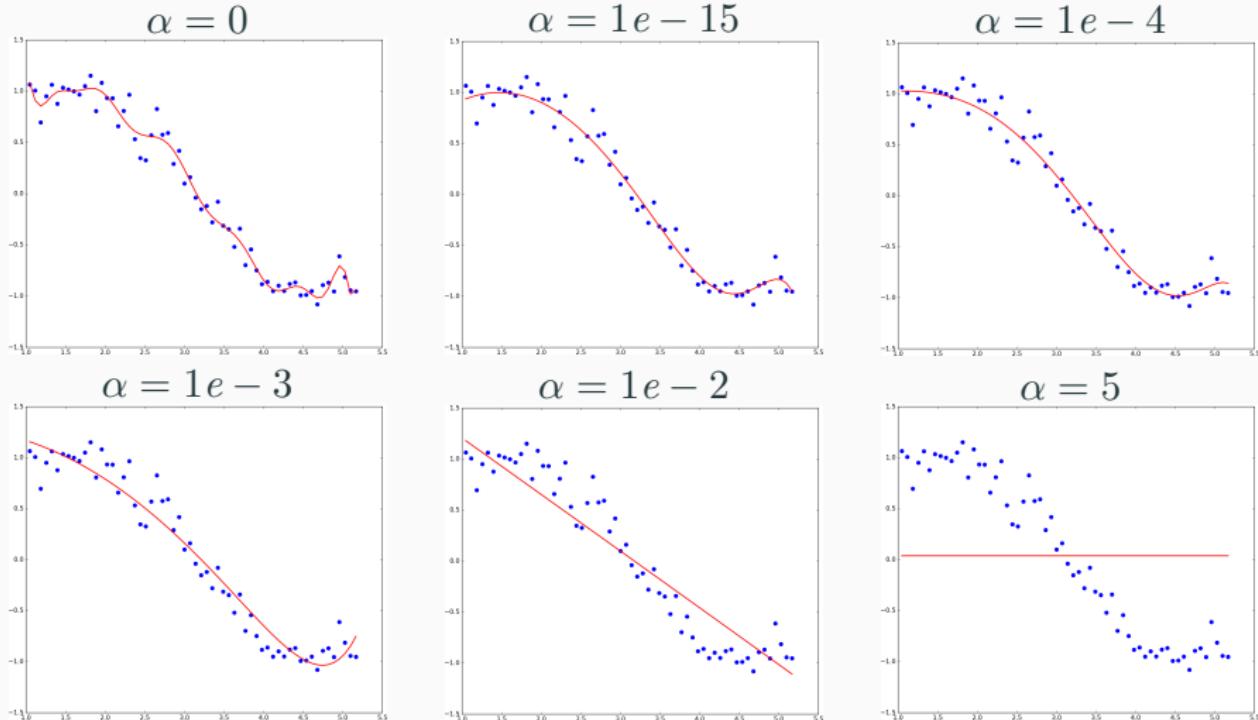
where Δ_{lms} is the update in case of non-regularized regression

Lasso regression (L1)

Lasso regression is a linear regression with a Lasso regularization:

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum (y_i - h_{\boldsymbol{\theta}}(x_i))^2 + \alpha \sum_{i=0}^p |\theta_i|$$

Results for $p = 15$ and varying α



Determination of the parameters in the lasso regression

Considering the cost function J:

$$J(\boldsymbol{\theta}) = J_{lms}(\boldsymbol{\theta}) + \alpha \sum_{i=0}^p |\theta_i|$$

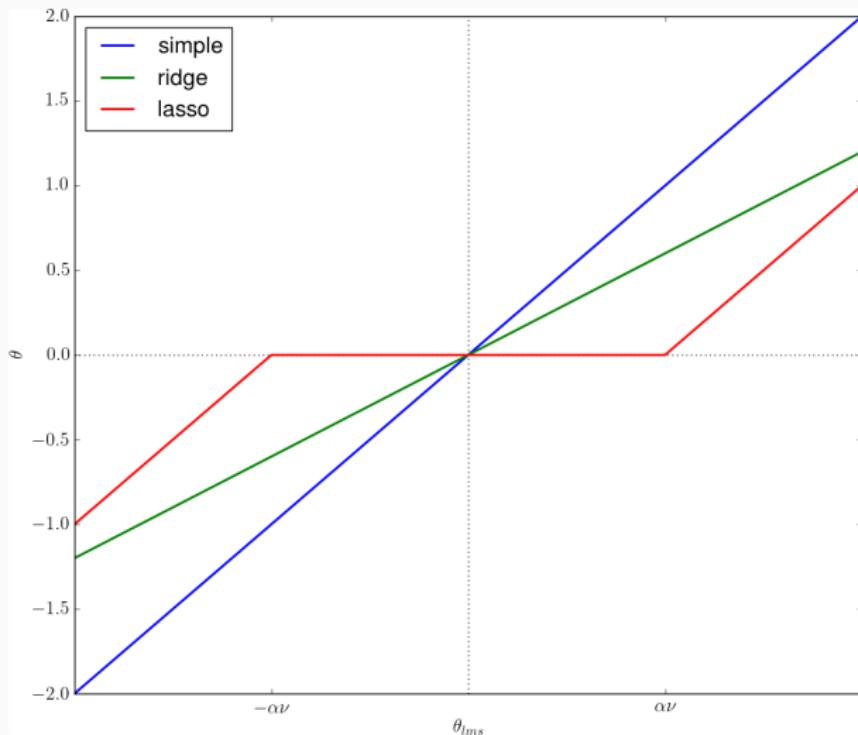
In a gradient algorithm, update of the parameters would be:

J is not differentiable. If we consider $\theta_{lms} = \theta_i^k - \nu \cdot \frac{\partial J_{lms}}{\partial \theta_i}$

The update rule is:

- $\theta_i^{k+1} = \theta_{lms} + \alpha\nu$ if $\theta_{lms} < -\alpha\nu$
- $\theta_i^k = 0$ if $-\alpha\nu < \theta_{lms} < \alpha\nu$
- $\theta_i^{k+1} = \theta_{lms} - \alpha\nu$ if $\theta_{lms} > \alpha\nu$

Summary of parameters updates



Ridge (L2)

- Prevents the overfitting
- includes all the features (dimensions) of the predictor, so it can be useless for high-dimensional predictors

Comparison Ridge/Lasso

Ridge (L2)

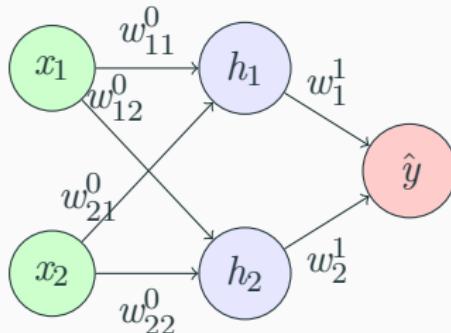
- Prevents the overfitting
- includes all the features (dimensions) of the predictor, so it can be useless for high-dimensional predictors

Lasso (L1)

- Provides sparse solutions and reduces the dimension of the predictor
- If some features in the predictor are correlated, arbitrarily select one from the others.

In a neural network

L2 (Ridge) and L1 (Lasso) regularization are widely used in Neural Network architectures.



- Non-regularized loss function: $L(\mathbf{w}) = \|\hat{y}(\mathbf{w}) - y\|^2$.
- L2-regularized loss function:
$$L(\mathbf{w}) = \|\hat{y}(\mathbf{w}) - y\|^2 + \alpha \sum_{i=0}^p |w_i| .$$
- L1-regularized loss function:
$$L(\mathbf{w}) = \|\hat{y}(\mathbf{w}) - y\|^2 + \alpha \sum_{i=0}^p w_i^2.$$

Advantage

L1/L2 regularization prevents overfitting even for large architecture (a big weight vector to optimize)

Wrapping-up

Advantage

L1/L2 regularization prevents overfitting even for large architecture (a big weight vector to optimize)

But...

It comes with an extra hyperparameter to tune (using, e.g., cross-validation): α

An example using scikit-learn

California House Pricing



Questions addressed in this lecture

- What are the more common areas of application of machine learning? [Cho21, 1]
- What are the different classes of ML problems? [Van16, 5.1]
- What are the different types of learning (supervised, ...)? [Van16, 5.1]
- How to validate/select a model? [Van16, 5.3]
- What is the cross-validation? [Van16, 5.3]
- How to encode qualitative features? [Van16, 5.4]
- What is L1(Lasso)/L2(Ridge) regularization? [Van16, 5.6];[Cho21, 4.4]

Refs

[Van16, *n*]: Jake VanderPlas, *Python Data Science Handbook*, section *n*

[Cho21, *n*]: Chollet, *Deep Learning with Python*, chapter *n*