Why deep learning?

Some technical and practical aspects

Clovis Galiez

Grenoble

Statistiques pour les sciences du Vivant et de l'Homme

February 24, 2021

AI? What is it?
What for?
What types?

Our scope of AI today

Long-standing dream (back to the anciant Greeks)[1] of having intelligent artificial creatures.

[1] See [The Quest for Artificial Intelligence, Nils J. Nilsson]

Our scope of AI today

Long-standing dream (back to the anciant Greeks)[1] of having intelligent artificial creatures.

We focus here on the modern acceptation of AI: a computer program able to autonomously react to a context to achieve a goal.

[1] See [The Quest for Artificial Intelligence, Nils J. Nilsson]

Our scope of AI today

Long-standing dream (back to the anciant Greeks)[1] of having intelligent artificial creatures.

We focus here on the modern acceptation of AI: a computer program able to autonomously react to a context to achieve a goal.

[1]See [The Quest for Artificial Intelligence, Nils J. Nilsson]

Our scope of AI today

Long-standing dream (back to the anciant Greeks)[1] of having intelligent artificial creatures.

We focus here on the modern acceptation of AI: a computer program able to autonomously react to a context to achieve a goal.

In particular, we will focus on (the distinction between):

- machine learning
- neural networks
- deep learning

[1] See [The Quest for Artificial Intelligence, Nils J. Nilsson]

Controversies

In the media:

- + AI solve all problems: ecology, unemployment, etc.
- - AI is dangerous: "big data is watching you"
- - AI is not fair: biases

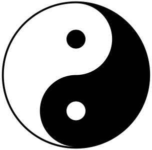
Interesting article about biases in The Consversation.

In the scientific community:

- + AI solves everything: you can predict anything if you have the data
- - AI does not explain anything: it's only black boxes

We will try today to get a fairer judgement on AI.

Be fair but critical



What is the right place of AI?

Manichean views

Think in background of two examples of AI applications (can be softwares, proof of concepts, etc.), one you would characterize as **good**, one as **bad**. Try to think in particular what would be the **societal impacts** if the examples you chose were generalized in the world.
Let's check in 2h :)

Today's outline

- AI? What for? What types?
- Machine learning
  - Learn from experience (data)
  - Underfitting (beware of biases)
  - Overfitting
- Neural networks
  - Properties
  - Optimizing the loss
  - Architecture matters
- Applications
- Timeline and future of AI
  - Biases
  - Ethics
  - Open problems

AI what for?

Any idea?

AI what for?

Any idea?

- Operational (automation of tasks)
  - Robotics (self-driving cars)
  - Indexing images, tagging
- Modeling
  - Extraction of information
  - Prediction

AI what for?

Any idea?

- Operational (automation of tasks): can be done by a human.
  We trust the human more than the AI.
  - Robotics (self-driving cars)
  - Indexing images, tagging
- Modeling
  - Extraction of information
  - Prediction

AI what for?

Any idea?

- Operational (automation of tasks): can be done by a human.
  We trust the human more than the AI.
  - Robotics (self-driving cars)
  - Indexing images, tagging
- Modeling: outcome can't be done by a human.
  We trust the AI more than human.
  - Extraction of information
  - Prediction

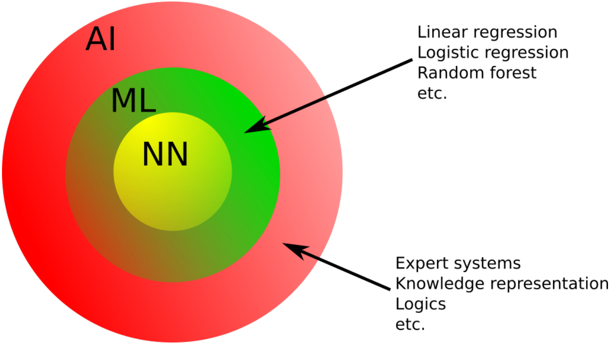
AI what for?

Any idea?

- Operational (automation of tasks): can be done by a human.
  We trust the human more than the AI.
  - Robotics (self-driving cars)
  - Indexing images, tagging
- Modeling: outcome can't be done by a human.
  We trust the AI more than human.
  - Extraction of information
  - Prediction

In both cases we are interested by the quality of the prediction, but may be also interested by bound guarantees or explanation of the "black-box".

AI taxonomy



We will focus on the *data science* part of artificial intelligence : **machine learning**.

Some machine learning methods

What machine learning tool you already know?

Some machine learning methods

What machine learning tool you already know?

For classification tasks:

- Linear Discriminant Analysis (LDA)
- Logistic regression
- Support Vector Machine (SVM)
- Random forests
- Artificial neural networks

For regression tasks:

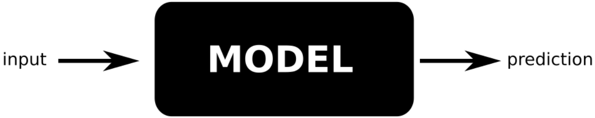
- Linear regression
- Regressive artificial neural networks

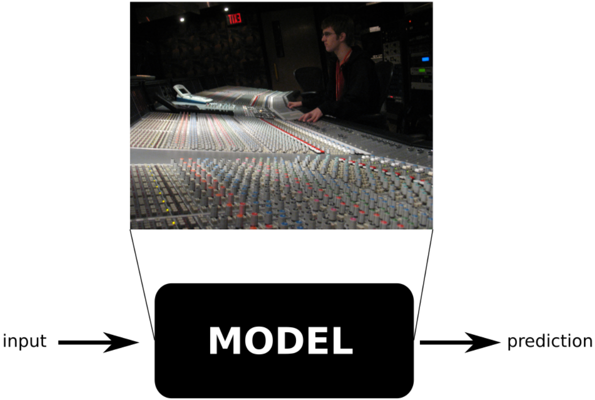
But also include parts of symbolic AI:

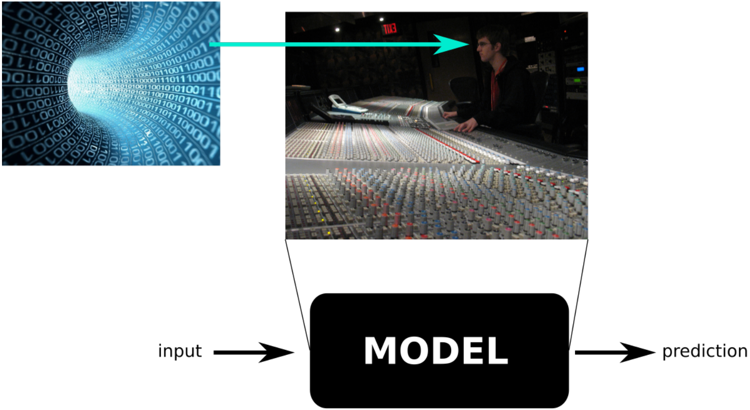
- Grammar inference
- Logic rule inference

Machine learning

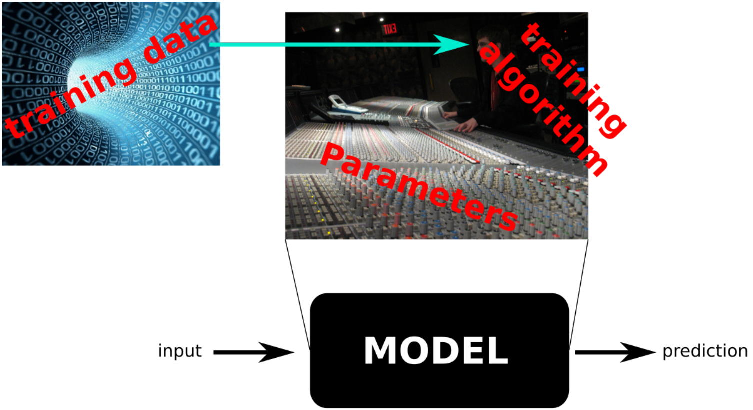
Machine learning




input
MODEL
prediction


input
MODEL
prediction

Machine learning



Some order of magnitude: number of parameters

Let's simplify: consider binary parameters $(0/1)$. With N parameters, how many possible combinations?

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters $(0/1)$. With N parameters, how many possible combinations? 2^N.

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters $(0/1)$. With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: ___ parameters.

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters $(0/1)$. With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: **175B** parameters.

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters $(0/1)$. With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: **175B** parameters.

Exercise

If I test 10B possibility per second (10GHz), how many seconds are necessary to try out all possible combinations?

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters (0/1). With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: **175B** parameters.

Exercise

If I test 10B possibility per second (10GHz), how many seconds are necessary to try out all possible combinations?
Number of possible combinations:
$2^{175.10^9}$

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters (0/1). With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: **175B** parameters.

Exercise

If I test 10B possibility per second (10GHz), how many seconds are necessary to try out all possible combinations?
Number of possible combinations:
$2^{175.10^9} = (2^{10})^{17.5\,10^9}$

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters $(0/1)$. With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: **175B** parameters.

Exercise

If I test 10B possibility per second (10GHz), how many seconds are necessary to try out all possible combinations?
Number of possible combinations:
$2^{175.10^9} = (2^{10})^{17.5\,10^9} \approx (10^3)^{17.5\,10^9}$

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters (0/1). With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: **175B** parameters.

Exercise

If I test 10B possibility per second (10GHz), how many seconds are necessary to try out all possible combinations?
Number of possible combinations:
$2^{175.10^9} = (2^{10})^{17.5\,10^9} \approx (10^3)^{17.5\,10^9} \approx 10^{52.000.000.000}$
Computational time required: $10^{52.000.000.000-10}$

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters (0/1). With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: **175B** parameters.

Exercise

If I test 10B possibility per second (10GHz), how many seconds are necessary to try out all possible combinations?
Number of possible combinations:
$2^{175.10^9} = (2^{10})^{17.5\,10^9} \approx (10^3)^{17.5\,10^9} \approx 10^{52.000.000.000}$
Computational time required: $10^{52.000.000.000-10} = 10^{51.999.999.990}$s
Age of universe:

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters (0/1). With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: **175B** parameters.

Exercise

If I test 10B possibility per second (10GHz), how many seconds are necessary to try out all possible combinations?

Number of possible combinations:

$2^{175.10^9} = (2^{10})^{17.5\,10^9} \approx (10^3)^{17.5\,10^9} \approx 10^{52.000.000.000}$

Computational time required: $10^{52.000.000.000-10} = 10^{51.999.999.990}$s

Age of universe: $13.8\,10^9$y

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters (0/1). With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: **175B** parameters.

Exercise

If I test 10B possibility per second (10GHz), how many seconds are necessary to try out all possible combinations?

Number of possible combinations:

$2^{175.10^9} = (2^{10})^{17.5\,10^9} \approx (10^3)^{17.5\,10^9} \approx 10^{52.000.000.000}$

Computational time required: $10^{52.000.000.000-10} = 10^{51.999.999.990}$s

Age of universe: $13.8\,10^9$y $\approx 10^{17}$s (!)

[2] [Brown et al. 20]

Some order of magnitude: number of parameters

Let's simplify: consider binary parameters (0/1). With N parameters, how many possible combinations? 2^N.

GPT-3[2] NLP (Natural Language Processing) model: **175B** parameters.

Exercise

If I test 10B possibility per second (10GHz), how many seconds are necessary to try out all possible combinations?

Number of possible combinations:

$2^{175.10^9} = \left(2^{10}\right)^{17.5\,10^9} \approx \left(10^3\right)^{17.5\,10^9} \approx 10^{52.000.000.000}$

Computational time required: $10^{52.000.000.000-10} = 10^{51.999.999.990}$s

Age of universe: $13.8\,10^9$y $\approx 10^{17}$s (!)

How it is possible to train such a model?

[2] [Brown et al. 20]

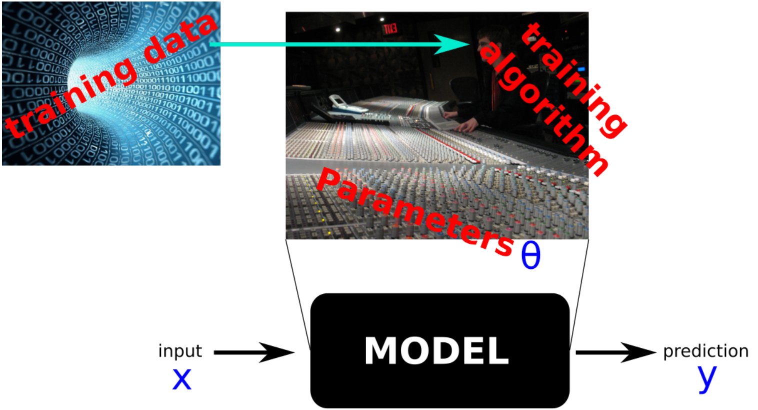
Some order of magnitude: training data

- Recent GPT-3 NLP (Natural Language Processing) model: trained on a corpus of 300B tokens ($\approx 1TB$).

Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

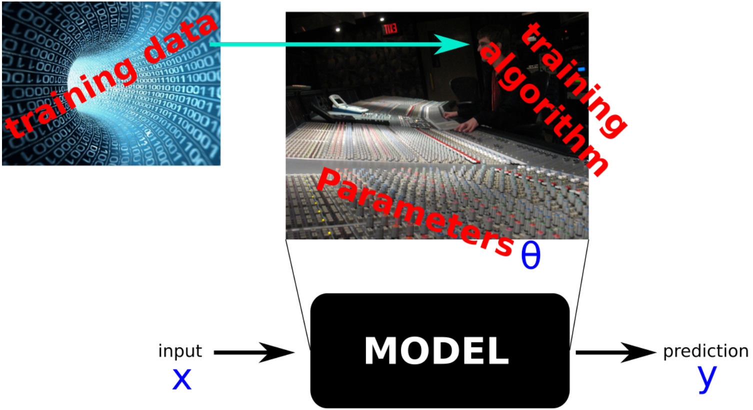
- Open Image Dataset V6 (18TB):
  - 9M images
  - 2M with labels from 600 classes

Training ML models



What training data?

Training ML models



What training data? Observations of (x, y):
$(x_1, y_1), \ldots (x_T, y_T)$

Worked-out example

You want to know what is the fuel consumption of this car at 180km/h.

Worked-out example

You want to know what is the fuel consumption of this car at 180km/h.

Problem

You **cannot measure it**, because either:

Worked-out example

You want to know what is the fuel consumption of this car at 180km/h.

Problem

You **cannot measure it**, because either:

- You don't feel like it  (operational)

Worked-out example

You want to know what is the fuel consumption of this car at 180km/h.

Problem

You **cannot measure it**, because either:

- You don't feel like it  (operational)
- Nobody will do it  (modeling)

What to do?

What to do? Make a model :) !

Worked-out example: choose the model

We want to model the **fuel consumption** y (L/100km) with respect to the **car speed** x (in km/h).

Worked-out example: choose the model

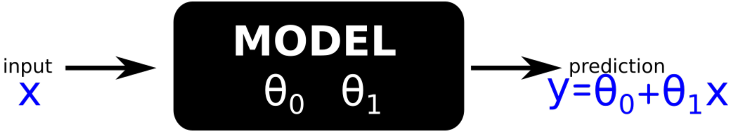
We want to model the **fuel consumption** y (L/100km) with respect to the **car speed** x (in km/h).

What model could you use for the dependency between x and y?

Worked-out example: choose the model

We want to model the **fuel consumption** y (L/100km) with respect to the **car speed** x (in km/h).

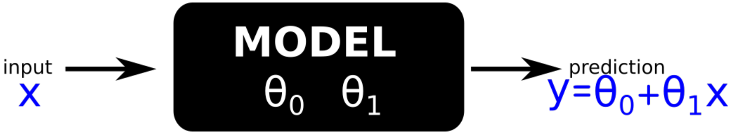
What model could you use for the dependency between x and y?



Worked-out example: choose the model

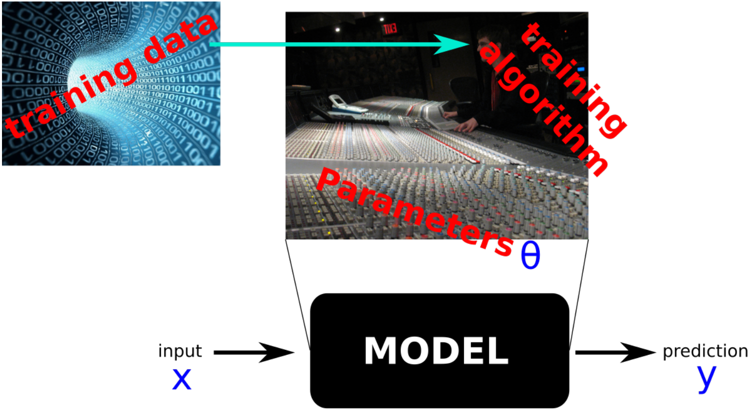
We want to model the **fuel consumption** y (L/100km) with respect to the **car speed** x (in km/h).

What model could you use for the dependency between x and y?



Need to find the right θ_0 and θ_1.
We say we need to **fit** the model.

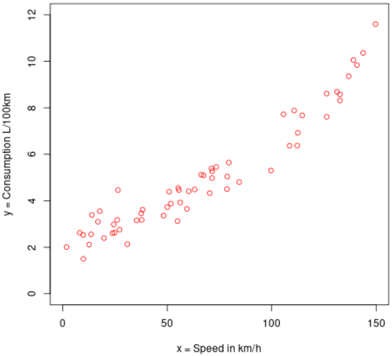
Worked-out example: fitting



Worked-out example: the training data

Your neighbor, gives you her home-made measurements.

It consists in $(x_i, y_i), i = 1, ..60$



Worked-out example: fitting the model

Goal: find optimal θ_0, θ_1 such that:

$$y \approx \theta_0 + \theta_1 . x$$

Worked-out example: fitting the model

Goal: find optimal θ_0, θ_1 such that:
$$\forall i = 1, ..60, y_i \approx \theta_0 + \theta_1 . x_i$$

Worked-out example: fitting the model

Goal: find optimal θ_0, θ_1 such that:
$\forall i = 1, ..60, |y_i - (\theta_0 + \theta_1.x_i)|$ is small

Worked-out example: fitting the model

Goal: find optimal θ_0, θ_1 such that:
$\forall i = 1, ..60, [y_i - (\theta_0 + \theta_1.x_i)]^2$ is small

Worked-out example: fitting the model

Goal: find optimal θ_0, θ_1 such that:
$\forall i = 1, ..60, [y_i - (\theta_0 + \theta_1.x_i)]^2$ is small

More formally, θ_0, θ_1 such that

$$\mathcal{L}(\theta_0, \theta_1) = \frac{1}{60} \sum_{i=1}^{60} [y_i - (\theta_0 + \theta_1.x_i)]^2 \tag{1}$$

is **minimal**.

Worked-out example: fitting the model

Goal: find optimal θ_0, θ_1 such that:
$\forall i = 1, ..60, [y_i - (\theta_0 + \theta_1.x_i)]^2$ is small

More formally, θ_0, θ_1 such that

$$\mathcal{L}(\theta_0, \theta_1) = \frac{1}{60} \sum_{i=1}^{60} [y_i - (\theta_0 + \theta_1.x_i)]^2 \tag{1}$$

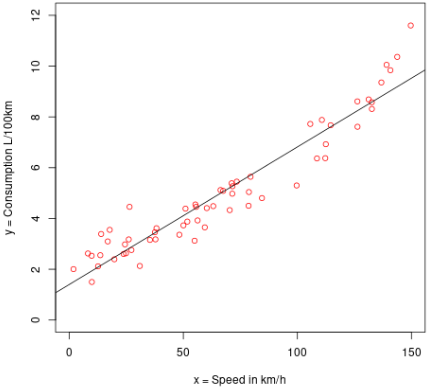
is **minimal**.

Definition

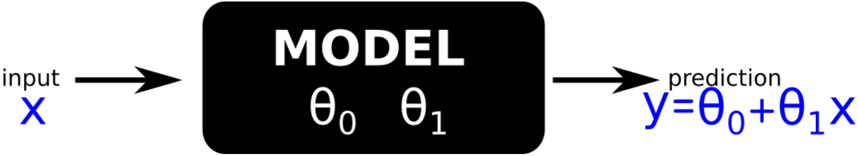
$\mathcal{L}$ is called a **loss function** for the model[a].

[a] This one in particular is called the *mean squared error*

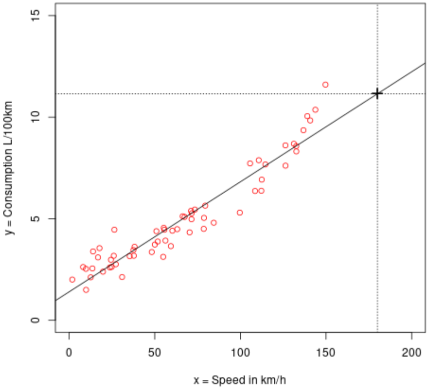
Worked-out example: the fit



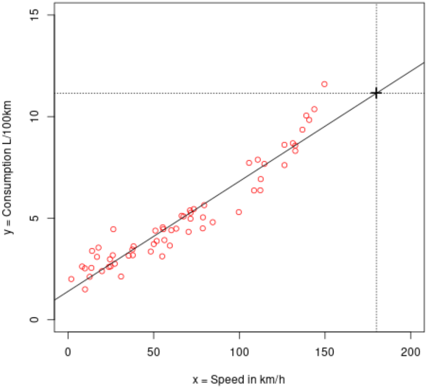
Worked-out example: the prediction



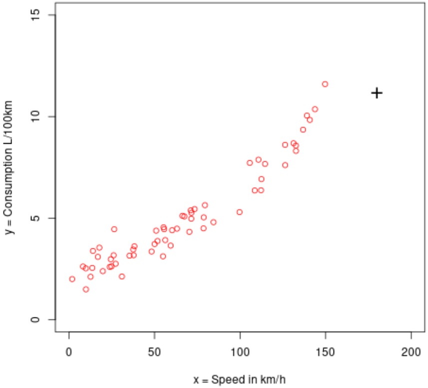
Worked-out example: the prediction



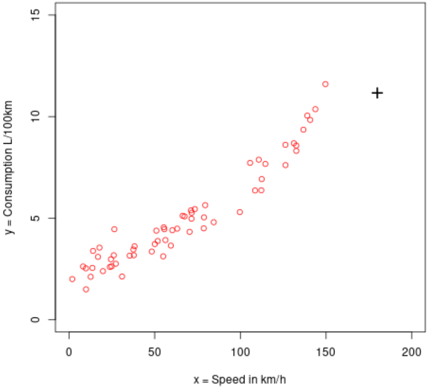
Worked-out example: toward more complex models



Worked-out example: toward more complex models

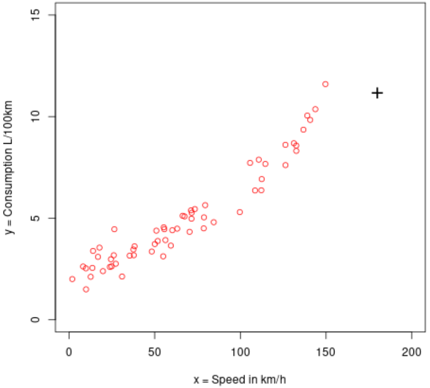


Worked-out example: toward more complex models



Any comment?

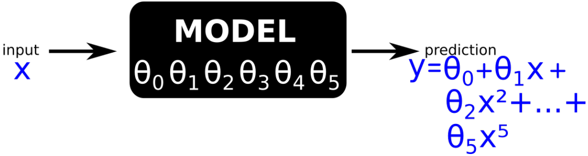
Worked-out example: toward more complex models



Any comment?

This phenomenon is known as **underfitting**

Worked-out example: toward more complex models



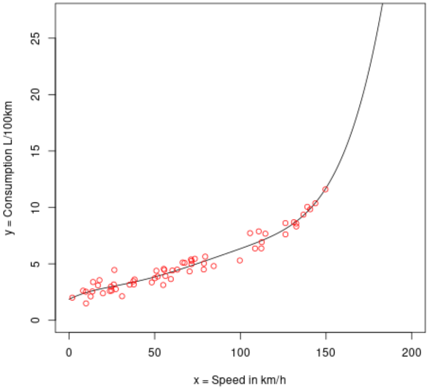
Worked-out example: toward more complex models

$\theta_0, \theta_1 ..., \theta_5$ such that

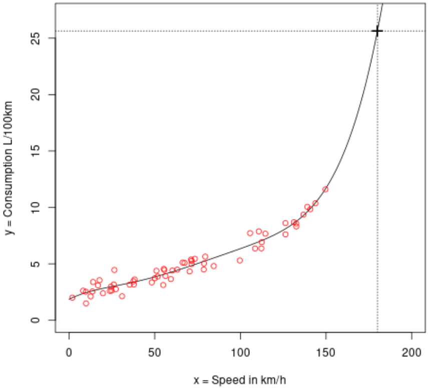
$$\mathcal{L}(\theta_0, ..., \theta_5) = \frac{1}{60} \sum_{i=1}^{60} [y_i - (\sum_{j=0}^{5} \theta_j . x_i^j)]^2 \tag{2}$$

is **minimal**.

Worked-out example: toward more complex models



Worked-out example: toward more complex models



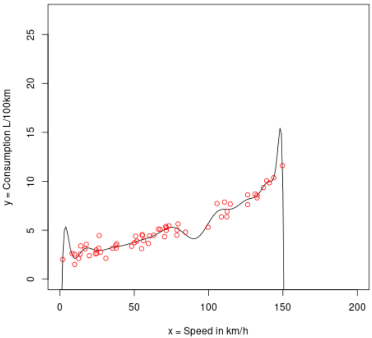
Worked-out example: toward more complex models

Informal definition

We will say that the polynomial model of degree 5 is more **expressive** than the linear regression.

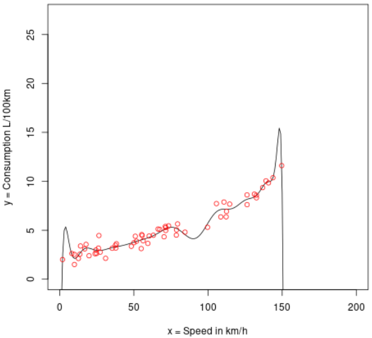
Parameters: the more the better?

Parameters: the more the better?



With 30 parameters: $\theta_0, ...\theta_{29}$

Parameters: the more the better?

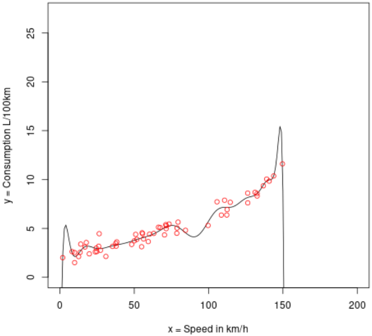


With 30 parameters: $\theta_0, ...\theta_{29}$

Definition

The phenomenon is called **overfitting**.

Parameters: the more the better?

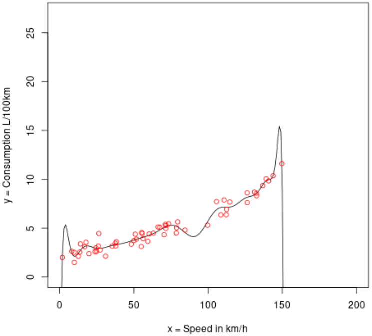


With 30 parameters: $\theta_0, ...\theta_{29}$

Definition

The phenomenon is called **overfitting**. Mainly happens because of **hyperparametrization**.

Parameters: the more the better?

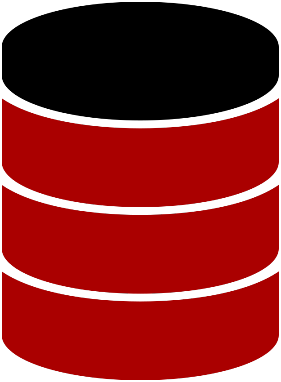


With 30 parameters: $\theta_0, ...\theta_{29}$

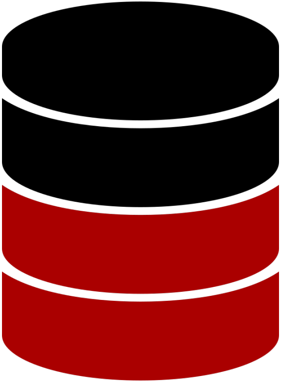
Definition

The phenomenon is called **overfitting**. Mainly happens because of **hyperparametrization**.

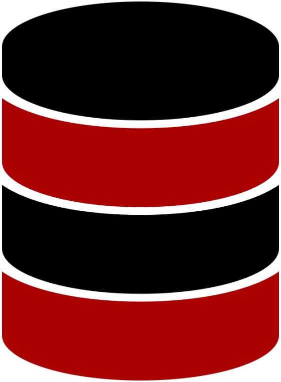
Cross-validation to control overfitting



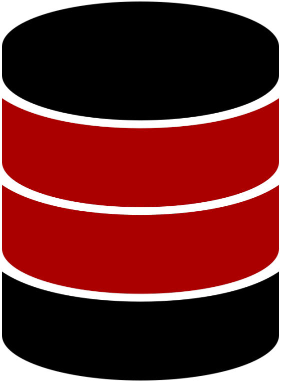
Cross-validation to control overfitting



Cross-validation to control overfitting

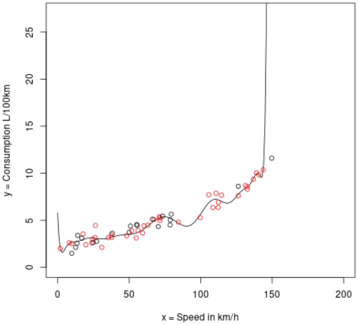


Cross-validation to control overfitting



Cross-validation example: 30 parameters

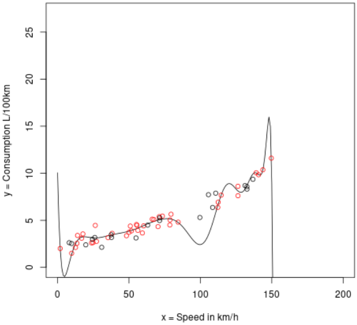
Red: training set Black: validation set



30 parameters, fold 1

Cross-validation example: 30 parameters

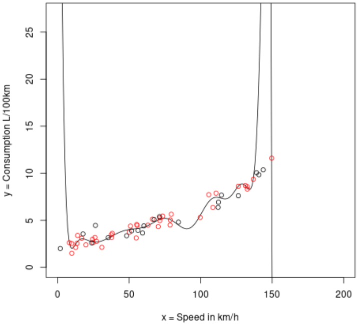
Red: training set Black: validation set



30 parameters, fold 2

Cross-validation example: 30 parameters

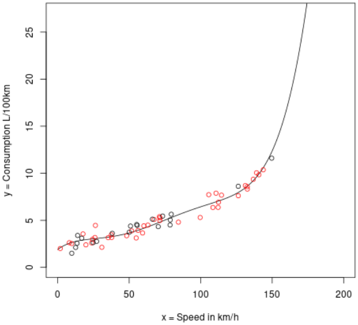
Red: training set Black: validation set



30 parameters, fold 3

Cross-validation example: 6 parameters

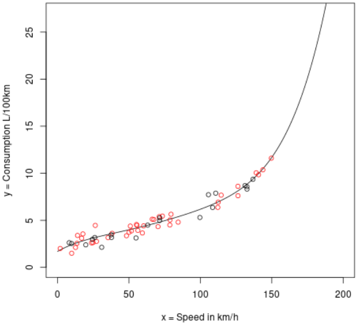
Red: training set Black: validation set



6 parameters, fold 1

Cross-validation example: 6 parameters

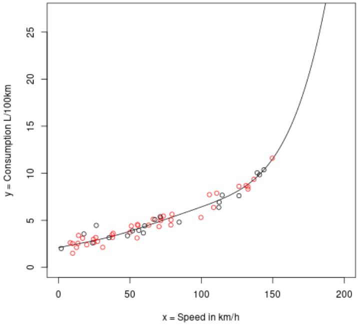
Red: training set Black: validation set



6 parameters, fold 2

Cross-validation example: 6 parameters

Red: training set Black: validation set



6 parameters, fold 3

What do you observe?

- Lower error on training with ___parameters
- If the error is on the validation is much higher than on training set, it means that the model ___.
- More variance among fits with ___parameters[3]

[3] As we will see, things are different with neural nets

What do you observe?

- Lower error on training with **more** parameters
- If the error is on the validation is much higher than on training set, it means that the model ___.
- More variance among fits with ___parameters[3]

[3] As we will see, things are different with neural nets

What do you observe?

- Lower error on training with **more** parameters
- If the error is on the validation is much higher than on training set, it means that the model **ovrefit**.
- More variance among fits with ___parameters[3]

[3] As we will see, things are different with neural nets

What do you observe?

- Lower error on training with **more** parameters
- If the error is on the validation is much higher than on training set, it means that the model **ovrefit**.
- More variance among fits with **more** parameters[3]

[3] As we will see, things are different with neural nets

What do you observe?

- Lower error on training with **more** parameters
- If the error is on the validation is much higher than on training set, it means that the model **ovrefit**.
- More variance among fits with **more** parameters[3]

More specifically, one can decompose the error of the model as:

$$\begin{aligned}
\mathbb{E}[||y-\hat{y}||^2] &= \mathbb{E}[||y-\mathbb{E}[\hat{y}]+\mathbb{E}[\hat{y}]-\hat{y}||^2] \\
&= \mathbb{E}[||y-\mathbb{E}[\hat{y}]||^2] + 2\times 0 + \mathbb{E}[||\mathbb{E}[\hat{y}]-\hat{y}||^2] \\
&= ||y-\mathbb{E}[\hat{y}]||^2 + \mathbb{E}[||\mathbb{E}[\hat{y}]-\hat{y}||^2] \\
&= \text{bias}^2 + \text{variance}
\end{aligned} \tag{2}$$

[3] As we will see, things are different with neural nets

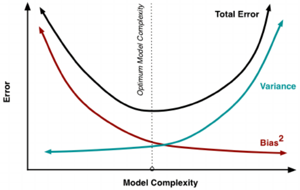
What do you observe?

- Lower error on training with **more** parameters
- If the error is on the validation is much higher than on training set, it means that the model **ovrefit**.
- More variance among fits with **more** parameters[3]

More specifically, one can decompose the error of the model as:

$$\begin{aligned}\mathbb{E}[||y-\hat{y}||^2] &= \mathbb{E}[||y-\mathbb{E}[\hat{y}]+\mathbb{E}[\hat{y}]-\hat{y}||^2] \\ &= \mathbb{E}[||y-\mathbb{E}[\hat{y}]||^2] + 2\times 0 + \mathbb{E}[||\mathbb{E}[\hat{y}]-\hat{y}||^2] \\ &= ||y-\mathbb{E}[\hat{y}]||^2 + \mathbb{E}[||\mathbb{E}[\hat{y}]-\hat{y}||^2] \\ &= \text{bias}^2 + \text{variance}\end{aligned} \tag{2}$$

This is known as the **bias-variance trade-off**:



[3] As we will see, things are different with neural nets

Quizz - Checkpoint

- We ___ a model by minimizing its ___ function on a ___ set.
- A model can have billion of ___ which make it ___ to try out all combinations.

Quizz - Checkpoint

- We **fit** a model by minimizing its ___ function on a ___ set.
- A model can have billion of ___ which make it ___ to try out all combinations.

Quizz - Checkpoint

- We **fit** a model by minimizing its **loss** function on a ___ set.
- A model can have billion of ___ which make it ___ to try out all combinations.

Quizz - Checkpoint

- We **fit** a model by minimizing its **loss** function on a **training** set.
- A model can have billion of ___ which make it ___ to try out all combinations.

Quizz - Checkpoint

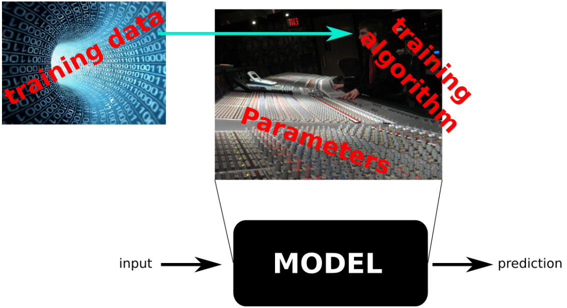
- We **fit** a model by minimizing its **loss** function on a **training** set.
- A model can have billion of **parameters** which make it ___ to try out all combinations.

Quizz - Checkpoint

- We **fit** a model by minimizing its **loss** function on a **training** set.
- A model can have billion of **parameters** which make it **impossible** to try out all combinations.

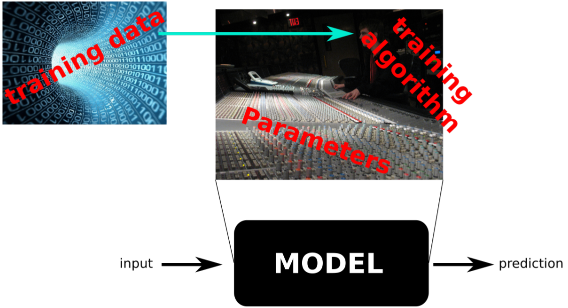
Quizz - Checkpoint

- We **fit** a model by minimizing its **loss** function on a **training** set.
- A model can have billion of **parameters** which make it **impossible** to try out all combinations.



Quizz - Checkpoint

- We **fit** a model by minimizing its **loss** function on a **training** set.
- A model can have billion of **parameters** which make it **impossible** to try out all combinations.



Remaining questions:

- What are the training algorithms?
- How is it possible to train with billions of parameters?

Learning the parameters

Learning algorithms

Of course...

Learning algorithm depends on the model to be fitted.

Learning algorithms

Of course...

Learning algorithm depends on the model to be fitted. But their job is to **minimize** a certain **loss**.

Learning algorithms

Of course...

Learning algorithm depends on the model to be fitted. But their job is to **minimize** a certain **loss**.

Examples:

- Small discrete models: enumeration and pruning
- Analytic solution for the minimum (*e.g.* linear regression)
- Convex loss function $\rightarrow$ gradient descent methods
- EM algorithms
- etc.

Loss with billions of parameters and datapoints

We would like $\theta_0, \theta_1 ..., \theta_p$ such that

$$\mathcal{L}(\theta_0, ..., \theta_p) = \frac{1}{N} \sum_{i=1}^{N} [y_i - (\sum_{j=0}^{p} \theta_j . x_i^j)]^2 \tag{3}$$

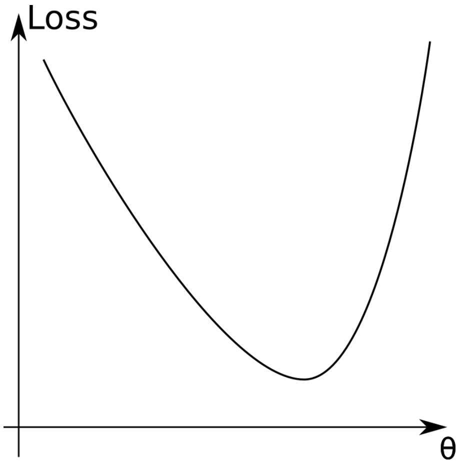
is **minimal**.

One would like to find the minimum of this loss.

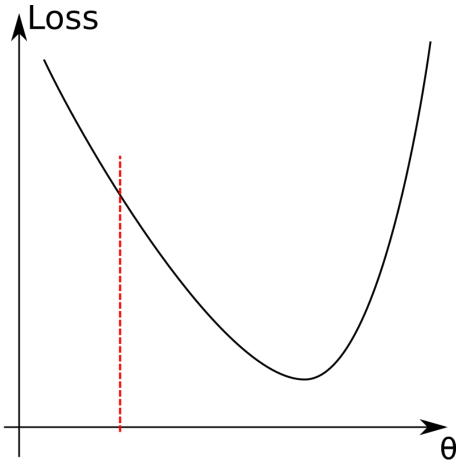
Issue: $p, n \sim 10^9$

- Cannot test every parameter combination
- Every computation of the loss is costly.

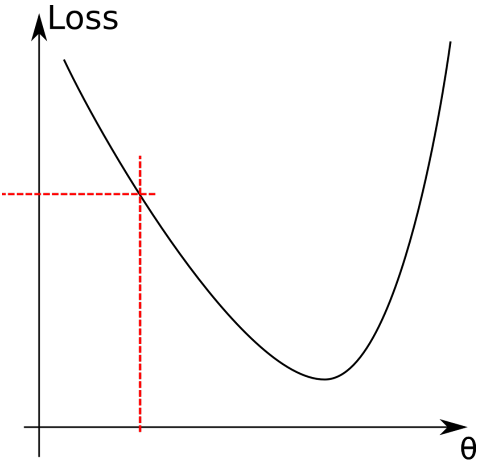
Gradient descent for convex loss



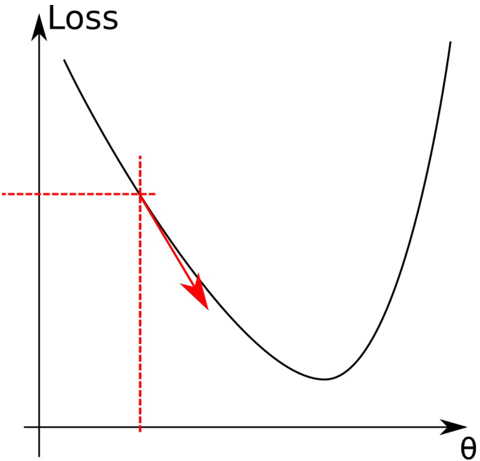
Gradient descent for convex loss



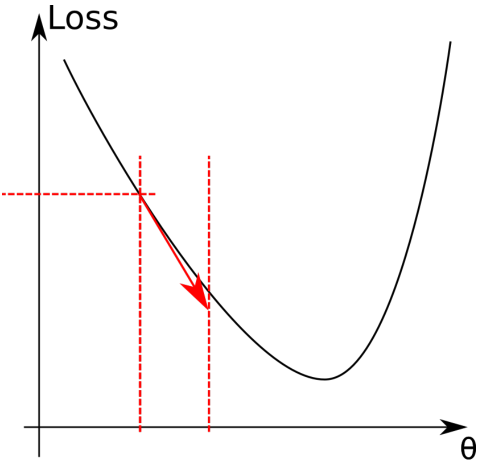
Gradient descent for convex loss



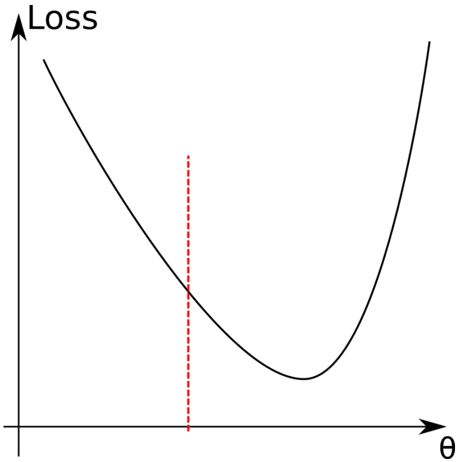
Gradient descent for convex loss



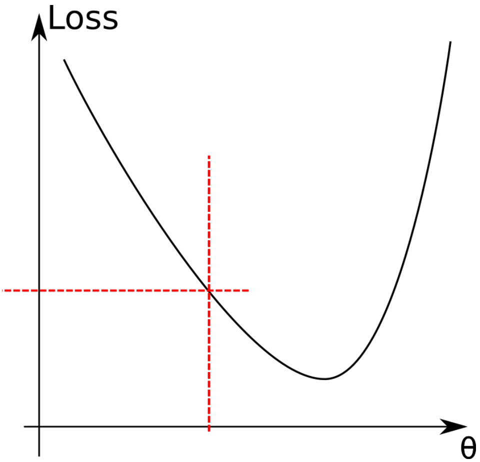
Gradient descent for convex loss



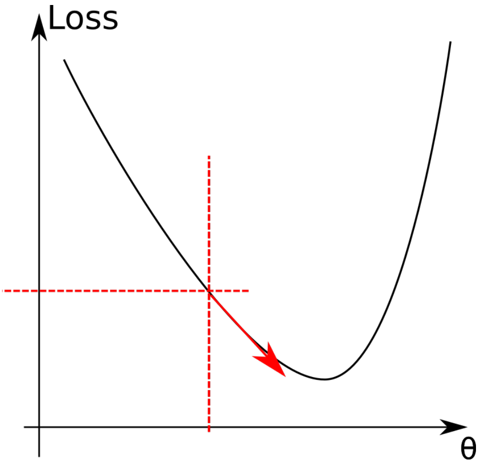
Gradient descent for convex loss



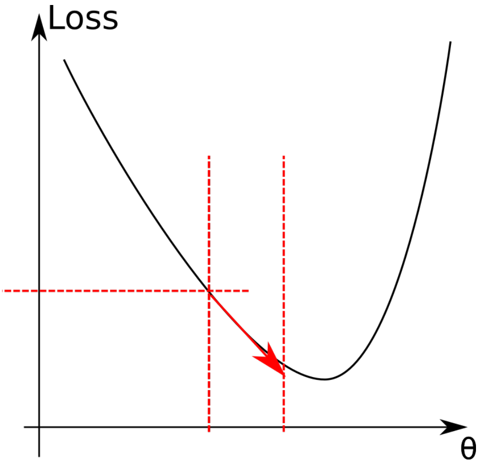
Gradient descent for convex loss



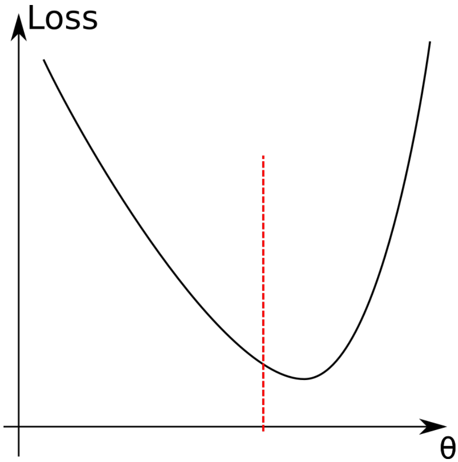
Gradient descent for convex loss



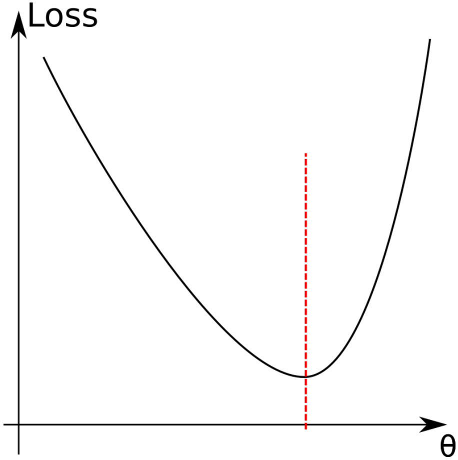
Gradient descent for convex loss



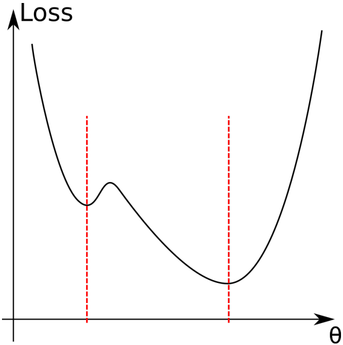
Gradient descent for convex loss



Gradient descent for convex loss

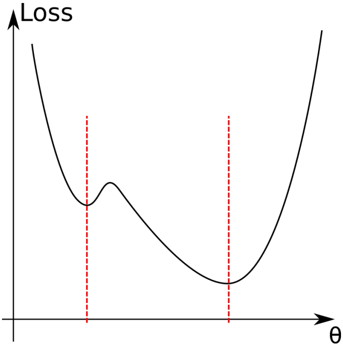


Gradient descent for **non**-convex loss



The gradient descent stops when it reaches a **critical point**: $\nabla \mathcal{L}(\theta_n) = \vec{0}$

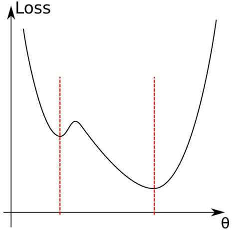
Gradient descent for **non**-convex loss



The gradient descent stops when it reaches a **critical point**: $\nabla\mathcal{L}(\theta_n) = \vec{0}$

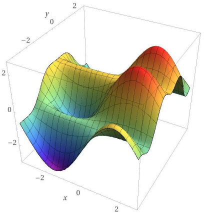
What are the possible *types* of critical points?

Critical points in higher dimension



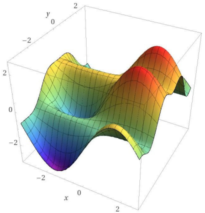
1 parameter

Critical points in higher dimension



≥ 2 parameters

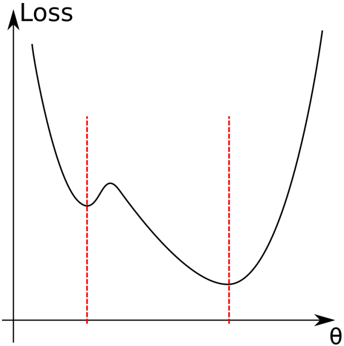
Critical points in higher dimension



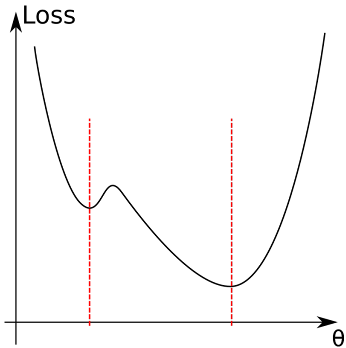
Type	Hessian[4]
Local minimum	all eigenvalues > 0
Local maximum	all eigenvalues < 0
Saddle points	else

[4] $\mathcal{H}_{ij} = \frac{\partial^2 \mathcal{L}}{\partial \theta_i \partial \theta_j}$

Escaping *easy* local minima



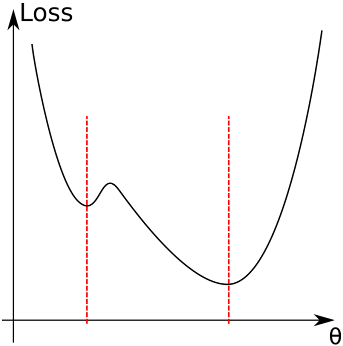
Escaping *easy* local minima



Gradient descent works well in practice for "small" non convexities by adding an inertia term:

$$\begin{aligned} \mathsf{grad}_{n+1} &= \nabla \mathcal{L}(\theta_n) \\ \theta_{n+1} &= \theta_n - \eta_{n+1}\mathsf{grad}_{n+1} \end{aligned} \tag{4}$$

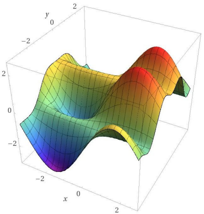
Escaping *easy* local minima



Gradient descent works well in practice for "small" non convexities by adding an inertia term:

$$\begin{aligned} \mathsf{grad}_{n+1} &= \nabla\mathcal{L}(\theta_n) + \nu_{n+1}.\mathsf{grad}_n \\ \theta_{n+1} &= \theta_n - \eta_{n+1}\mathsf{grad}_{n+1} \end{aligned} \tag{4}$$

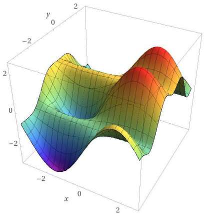
Escaping local non-minimal critical points



How is it possible to escape a critical point that is **not** a minimum?

[5] Would need to invert a 175.10^9 dimensional matrix for GPT3... $\approx 5.3.10^{33}$ operations :-/

Escaping local non-minimal critical points

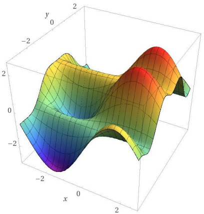


How is it possible to escape a critical point that is **not** a minimum?

- Compute the Hessian

[5] Would need to invert a 175.10^9 dimensional matrix for GPT3... $\approx 5.3.10^{33}$ operations :-/

Escaping local non-minimal critical points

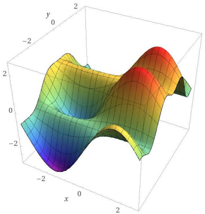


How is it possible to escape a critical point that is **not** a minimum?

- Compute the Hessian (too big if a lot of parameters[5])

[5]Would need to invert a 175.10^9 dimensional matrix for GPT3... $\approx 5.3.10^{33}$ operations :-/

Escaping local non-minimal critical points



How is it possible to escape a critical point that is **not** a minimum?

- Compute the Hessian (too big if a lot of parameters[5])
- **Add "noise" to the gradient !**

How to add some noise?

[5] Would need to invert a 175.10^9 dimensional matrix for GPT3... $\approx 5.3.10^{33}$ operations :-/

Stochastic gradient descent (SGD)

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} l(\theta, x_i, y_i)$$

where $\{(x_i, y_i)\}$ is the training set.

[6] By linearity of expectation: $\mathbb{E}_{\mathcal{B}}[\mathcal{L}_{\mathcal{B}}(\theta, X, Y)] = \mathcal{L}(\theta, X, Y)$

Stochastic gradient descent (SGD)

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} l(\theta, x_i, y_i)$$

where $\{(x_i, y_i)\}$ is the training set.

If one computes instead the gradient on a **random subset** $\mathcal{B}$ (coined a **batch**) of the training set, one has an unbiased[6] **noisy** estimate of the gradient:

$$\mathcal{L}_{\mathcal{B}}(\theta, X, Y) = \sum_{i \in \mathcal{B}} l(\theta, x_i, y_i) \tag{5}$$

[6] By linearity of expectation: $\mathbb{E}_{\mathcal{B}}[\mathcal{L}_{\mathcal{B}}(\theta, X, Y)] = \mathcal{L}(\theta, X, Y)$

SGD properties

- SGD is guaranteed to converge to the minimum for a convex loss.
- SGD can escape *easy* saddle points
- the computational cost is **much reduced**! See: $N_{\text{updates}} \times |\mathcal{B}|$
- with small batches can take advantage of modern hardware (GPUs)
- works very well in practice to find *good* local minima

SGD properties

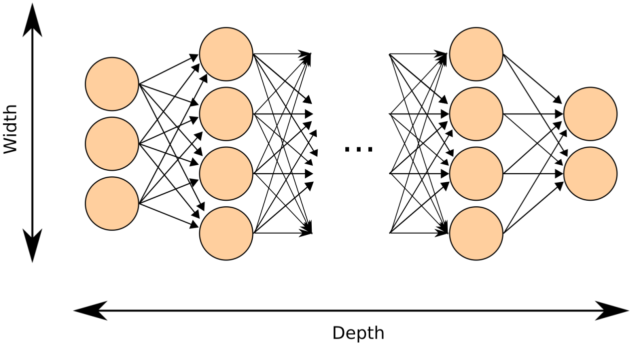
- SGD is guaranteed to converge to the minimum for a convex loss.
- SGD can escape *easy* saddle points
- the computational cost is **much reduced**! See: $N_{\text{updates}} \times |\mathcal{B}|$
- with small batches can take advantage of modern hardware (GPUs)
- works very well in practice to find *good* local minima

It seems that we have a candidate to train a model with billions of parameters...

Neural Networks

Neural networks

Dense feed-forward neural network:



Activation of neuron i in layer l: $z_{i,l} = \sigma_j l(\sum_{k \in \mathcal{I}_{i,l}} w_{l,i,k} z_{k,l-1})$

Parameters[7]: $w_{j,k}$'s.

σ_l: activation functions.

[7] GPT-3 has 96 layers and 175B parameters

Input and outputs

Inputs are vectors in $\mathbb{R}^m$, and output vectors in $\mathbb{R}^n$.

It can be:

-  in $\rightarrow$  out

Input and outputs

Inputs are vectors in $\mathbb{R}^m$, and output vectors in $\mathbb{R}^n$.

It can be:

-  in $\rightarrow$  out
-  in $\rightarrow$ *plant* out

Input and outputs

Inputs are vectors in $\mathbb{R}^m$, and output vectors in $\mathbb{R}^n$.

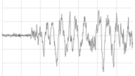
It can be:

-  in $\rightarrow$  out
-  in $\rightarrow$ *plant* out
- *Draw me a plant* in $\rightarrow$  out

Input and outputs

Inputs are vectors in $\mathbb{R}^m$, and output vectors in $\mathbb{R}^n$.

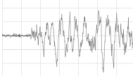
It can be:

-  in $\rightarrow$  out
-  in $\rightarrow$ *plant* out
- *Draw me a plant* in $\rightarrow$  out
-  in $\rightarrow$ "*plant*" out

Input and outputs

Inputs are vectors in $\mathbb{R}^m$, and output vectors in $\mathbb{R}^n$.

It can be:

-  in $\rightarrow$  out
-  in $\rightarrow$ *plant* out
- *Draw me a plant* in $\rightarrow$  out
-  in $\rightarrow$ *"plant"* out
- ...

Images can be encoded as 1 px $=$ 1 dimension of the vector, a signal as 1 time point $=$ 1 dimension, etc.

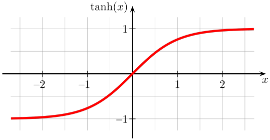
Time to play

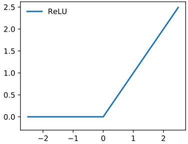
Let's see how an ANN looks like in practice:
https://playground.tensorflow.org

Activation function

What activation function?

- Its gradient has to be simple to compute
- It should "make sense"





Note that the linear regression can be recovered with 1 single output neuron and $f(x) = x$ as activation function.

Neural networks as a *"universal"* model

Theorem [Lu et al. NIPS'18]

For any $\epsilon > 0$, and any Lebesgue-integrable function $f : \mathbb{R}^m \to \mathbb{R}$, there exists a ReLU neural network η such that:

$$\int |f(x) - \eta(x)| dx < \epsilon \tag{6}$$

Moreover, one can also restict the maximal width to $m + 4$.

There are theoretical results ensuring that neural nets can approximate arbitrarily well any well-behaved function.

Time to play

Let's see how ReLU and depth allow to model complex data:
https://playground.tensorflow.org

Training: SGD to the rescue!

Which training algorithm?

- + Automatic computation of the gradient of the loss using *automatic differentiation*.
- - The loss is non-convex even with linear activation functions [Kawaguchi NIPS'16] .

Nevertheless...

Gradient descent works very well in practice.

Training: SGD to the rescue!

Which training algorithm?

- + Automatic computation of the gradient of the loss using *automatic differentiation*.
- - The loss is non-convex even with linear activation functions [Kawaguchi NIPS'16] .

Nevertheless...

Gradient descent works very well in practice.

We now have some theoretical results explaining (a bit) why it SGD works well for deep neural networks [Hardt et al. 16] (SGD and generalization error), [Chaudhari and Soatto ICLR'18] (SGD naturally regularizes).

Training a neural network: the big picture

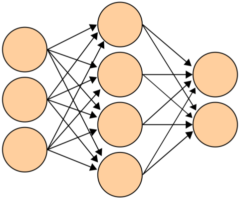
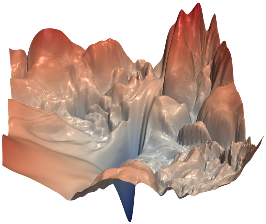
A (dense) NN is therefore simply a function
$f_w(x) = \sigma_d(\sum_{k_d} w_{d,1,k_d}\sigma_{d-1}(\sum ...\sigma_1(\sum_{k_1} w_{1,i,k_1}x_{k_1}))).$
How to fit the parameters $w_{j\,k\,l}$?

- Get a traning set (x_s, y_s), can range from kB to TB of data[8].
- Define a loss function, for instance $\mathcal{L}(w) = \sum_s [y_s - f_w(x_s)]^2$.
- Then compute[9] the derivatives $\frac{\partial \mathcal{L}}{\partial w_{i,j}}$
- Use your favorite variant of SGD, and find a good minimum of the loss.

[8] the more the better
[9] An analytical formula can be obtain by a computer for well-chosen activation functions

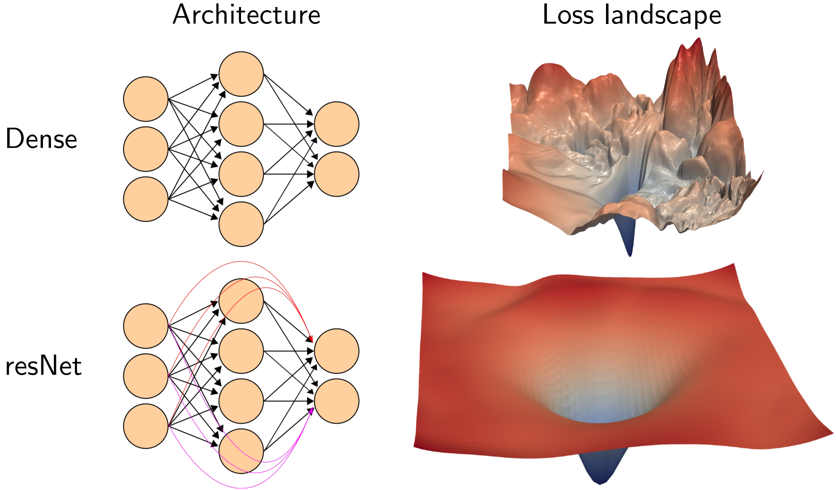
Architecture matters for training: resNet example

The loss function can change dramatically depending on the NN architecture.

	Architecture	Loss landscape
Dense		

Architecture matters for training: resNet example

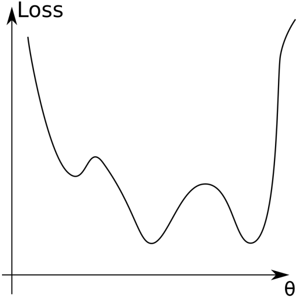
The loss function can change dramatically depending on the NN architecture.



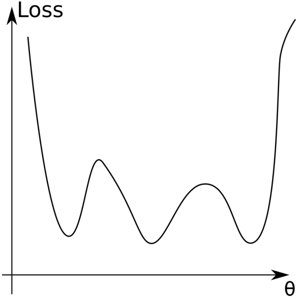
Architecture matters for training: depth

[Choromanska et al. 15] (spin-glass model)

The bigger the network, the fewer bad local minima.



$\rightarrow$



ANN and optimization

The loss of neural networks is ___.
The loss landscape depends on the ___. It therefore makes sense to use an architecture for which local minima are of ___ quality (like deep or specific like ResNet).
So... ___ neural networks are universal models that can be trained efficiently using ___. But the deeper the ANN, the ___ parameters are involved... so even with a huge load of data, it should ___, right?!

ANN and optimization

The loss of neural networks is **non-convex**.
The loss landscape depends on the ___. It therefore makes sense to use an architecture for which local minima are of ___ quality (like deep or specific like ResNet).
So... ___ neural networks are universal models that can be trained efficiently using ___. But the deeper the ANN, the ___ parameters are involved... so even with a huge load of data, it should ___, right?!

ANN and optimization

The loss of neural networks is **non-convex**.
The loss landscape depends on the **architecture**. It therefore makes sense to use an architecture for which local minima are of ___ quality (like deep or specific like ResNet).
So... ___ neural networks are universal models that can be trained efficiently using ___. But the deeper the ANN, the ___ parameters are involved... so even with a huge load of data, it should ___, right?!

ANN and optimization

The loss of neural networks is **non-convex**.
The loss landscape depends on the **architecture**. It therefore makes sense to use an architecture for which local minima are of **better** quality (like deep or specific like ResNet).
So... ___ neural networks are universal models that can be trained efficiently using ___. But the deeper the ANN, the ___ parameters are involved... so even with a huge load of data, it should ___, right?!

ANN and optimization

The loss of neural networks is **non-convex**.

The loss landscape depends on the **architecture**. It therefore makes sense to use an architecture for which local minima are of **better** quality (like deep or specific like ResNet).

So... **ReLU** neural networks are universal models that can be trained efficiently using ___. But the deeper the ANN, the ___ parameters are involved... so even with a huge load of data, it should ___, right?!

ANN and optimization

The loss of neural networks is **non-convex**.

The loss landscape depends on the **architecture**. It therefore makes sense to use an architecture for which local minima are of **better** quality (like deep or specific like ResNet).

So... **ReLU** neural networks are universal models that can be trained efficiently using **SGD**. But the deeper the ANN, the ___ parameters are involved... so even with a huge load of data, it should ___, right?!

ANN and optimization

The loss of neural networks is **non-convex**.
The loss landscape depends on the **architecture**. It therefore makes sense to use an architecture for which local minima are of **better** quality (like deep or specific like ResNet).
So... **ReLU** neural networks are universal models that can be trained efficiently using **SGD**. But the deeper the ANN, the **more** parameters are involved... so even with a huge load of data, it should ___, right?!

ANN and optimization

The loss of neural networks is **non-convex**.

The loss landscape depends on the **architecture**. It therefore makes sense to use an architecture for which local minima are of **better** quality (like deep or specific like ResNet).

So... **ReLU** neural networks are universal models that can be trained efficiently using **SGD**. But the deeper the ANN, the **more** parameters are involved... so even with a huge load of data, it should **overfit**, right?!

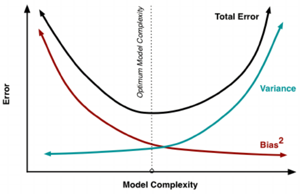
NN and overfitting

The reason why ANN tend not to overfit is not clear yet.

NN and overfitting

The reason why ANN tend not to overfit is not clear yet.

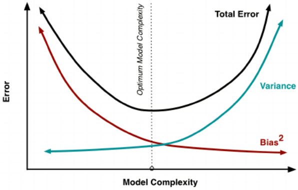
Variance increasing with the nb of parameters is not true for SGD-learnt ANN.

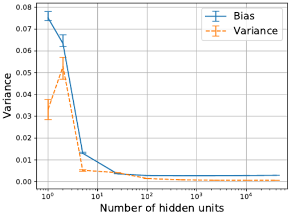


NN and overfitting

The reason why ANN tend not to overfit is not clear yet.

Variance increasing with the nb of parameters is not true for SGD-learnt ANN.



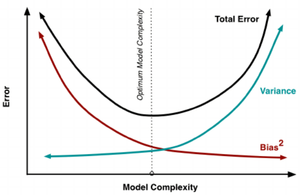


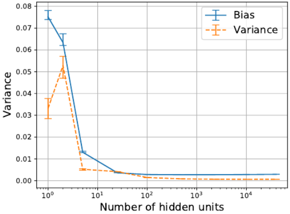
[Neal et al. 19]

NN and overfitting

The reason why ANN tend not to overfit is not clear yet.

Variance increasing with the nb of parameters is not true for SGD-learnt ANN.





[Neal et al. 19]

A big insight [Achille and Soatto JMLR'18]

One should rather measure the amount of information from the training data that is transferred to the weights during the fit: less and less amount is transfered the deeper you go.

Time to play

Let's see what neurons learn with respect to depth:
https://playground.tensorflow.org

Summary: neural nets, why does it work that well?

No real "breakthrough" but rather a concordance of events:

[10] Actually it may be also why biological neural nets have been selected by evolution

Summary: neural nets, why does it work that well?

No real "breakthrough" but rather a concordance of events:

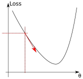
- More data (better as for any ML, allows for more depth) 

[10] Actually it may be also why biological neural nets have been selected by evolution

Summary: neural nets, why does it work that well?

No real "breakthrough" but rather a concordance of events:

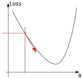
- More data (better as for any ML, allows for more depth)


- Better optimization algorithms (SGD and its variants)


[10] Actually it may be also why biological neural nets have been selected by evolution

Summary: neural nets, why does it work that well?

No real "breakthrough" but rather a concordance of events:

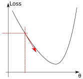
- More data (better as for any ML, allows for more depth) 
- Better optimization algorithms (SGD and its variants) 

- Better hardware (GPUs for parallel computations) 

[10] Actually it may be also why biological neural nets have been selected by evolution

Summary: neural nets, why does it work that well?

No real "breakthrough" but rather a concordance of events:

- More data (better as for any ML, allows for more depth) 
- Better optimization algorithms (SGD and its variants) 

- Better hardware (GPUs for parallel computations) 
- Some luck[10]: ANN tend *not* to overfit (but it has been noticed afterwards) 

[10] Actually it may be also why biological neural nets have been selected by evolution

Some architecture you need to know

What architecture are allowed?

What architecture are allowed?

Virtually any, as soon as you can compute the gradient of the loss :)

Image processing: convolutional neural nets (CNN)

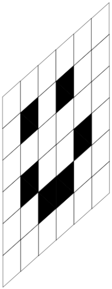


Image processing: convolutional neural nets (CNN)

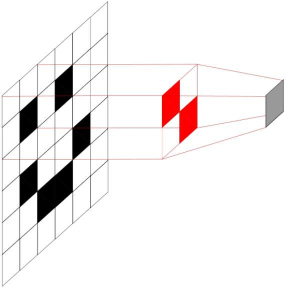


Image processing: convolutional neural nets (CNN)

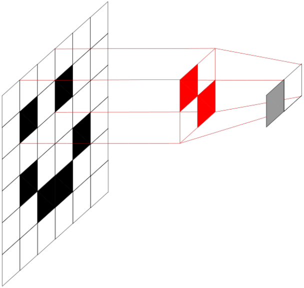


Image processing: convolutional neural nets (CNN)

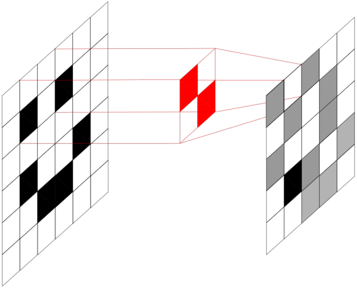


Image processing: convolutional neural nets (CNN)

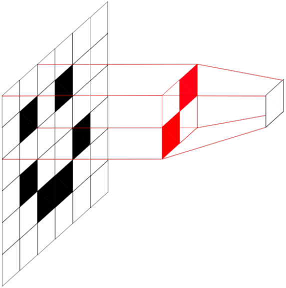


Image processing: convolutional neural nets (CNN)

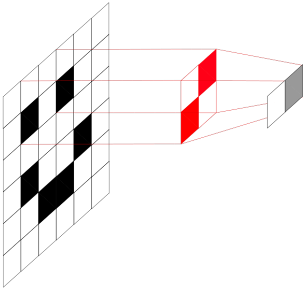


Image processing: convolutional neural nets (CNN)

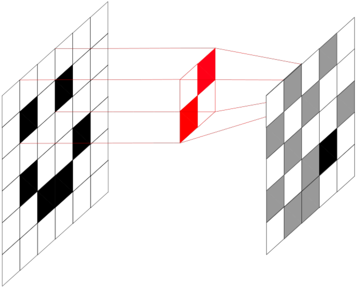


Image processing: convolutional neural nets (CNN)

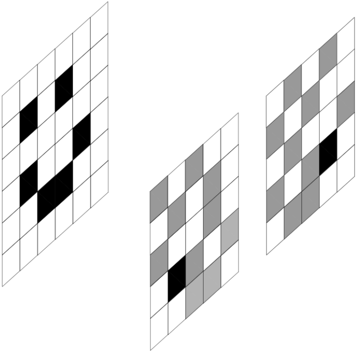
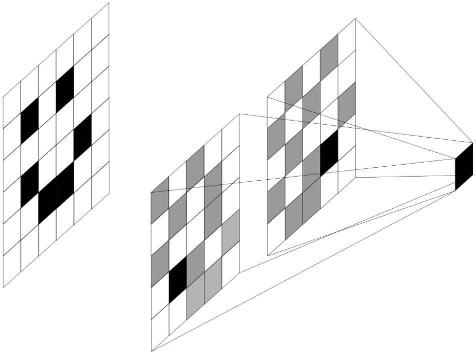
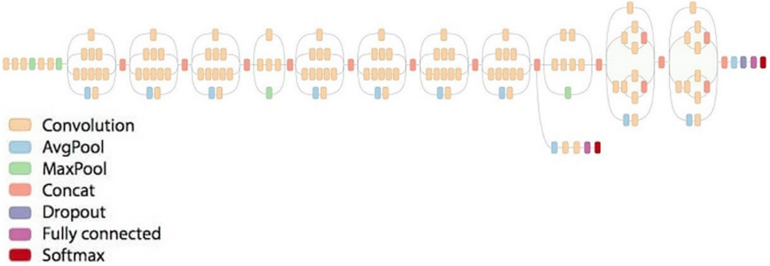


Image processing: convolutional neural nets (CNN)



Real-world CNN



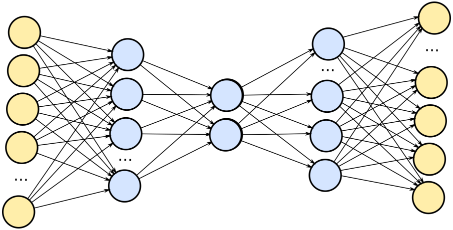
Source: InceptionV3

Applications: image recognition, segmentation [Minaee et al. 20] , etc.

Time to play

Demo ResNet50.

Compressing data with NN: Autoencoders



Try to accurately reconstruct the input (unsupervised).

Analogy with the "Chinese Whispers"[11]*.* The mid-layer is called a latent and contains a compressed version of the input. Works when the data has an underlying structure.

[11] "Téléphone Arabe" in French

Compressing data: autoencoders

Example: predicting high altitude pollution from satellite images

Problem

- Cannot generate a lot of training data: only few ballon can be sent per year.
- hourly acquisition of 2000x2000px satellite images

What will happen if we learn the pollution from the raw sattelite images?

Compressing data: autoencoders

Example: predicting high altitude pollution from satellite images

Problem

- Cannot generate a lot of training data: only few ballon can be sent per year.
- hourly acquisition of 2000x2000px satellite images

What will happen if we learn the pollution from the raw sattelite images?
Overfitting

Compressing data: autoencoders

Example: predicting high altitude pollution from satellite images

Problem

- Cannot generate a lot of training data: only few ballon can be sent per year.
- hourly acquisition of 2000x2000px satellite images

What will happen if we learn the pollution from the raw sattelite images?
Overfitting
You can: ___ the images of sattelite (trained on all unlabled data) and predict from the small ___ layer.

Compressing data: autoencoders

Example: predicting high altitude pollution from satellite images

Problem

- Cannot generate a lot of training data: only few ballon can be sent per year.
- hourly acquisition of 2000x2000px satellite images

What will happen if we learn the pollution from the raw sattelite images?
Overfitting
You can: **compress** the images of sattelite (trained on all unlabled data) and predict from the small ___ layer.

Compressing data: autoencoders

Example: predicting high altitude pollution from satellite images

Problem

- Cannot generate a lot of training data: only few ballon can be sent per year.
- hourly acquisition of 2000x2000px satellite images

What will happen if we learn the pollution from the raw sattelite images?
Overfitting
You can: **compress** the images of sattelite (trained on all unlabled data) and predict from the small **latent** layer.

Compressing data: autoencoders

Example: predicting high altitude pollution from satellite images

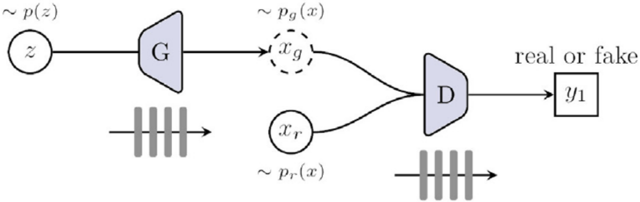
Problem

- Cannot generate a lot of training data: only few ballon can be sent per year.
- hourly acquisition of 2000x2000px satellite images

What will happen if we learn the pollution from the raw sattelite images?
Overfitting
You can: **compress** the images of sattelite (trained on all unlabled data) and predict from the small **latent** layer.

It is very common to have **a lot** of **un**labled data and **few labeled** data.

Generative Adversial Networks (GAN)

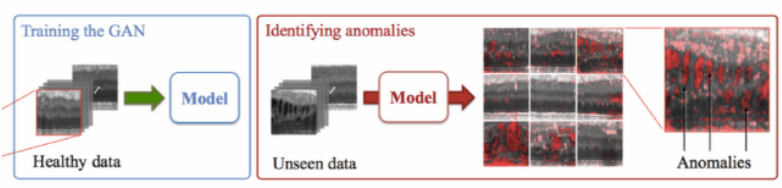


Equilibrium reached when $p_g = p_r$

Generative Adversial Networks (GAN)

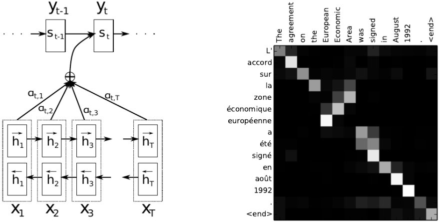
Can be use as a generative process

Generative Adversial Networks (GAN)



or as a classifier [Yi et al. 20]

Text processing: attention mechanism



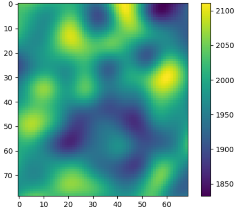
[Bahdanau et al. ICLR'15]

Improvement of the same idea: the Transformer architecture [Vaswani et al. 17] ,[Brown et al. 20]

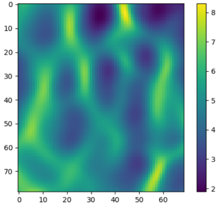
Applications

Surrogate models

Goal: use a neural network to approximate a costly model



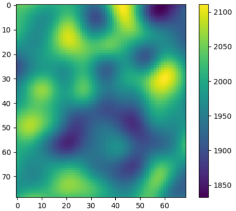
$\longrightarrow$



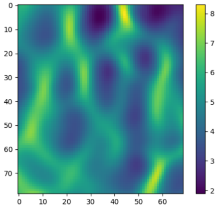
Topography Diff. eq. Wind prediction

[M. Roux, internship 20]

Goal: use a neural network to approximate a costly model



$\longrightarrow$



Topography CNN Wind prediction

[M. Roux, internship 20]

Glaciers grow and melt

AI and glacier evolution prediction

The causes influencing the evolution of glacier are complex:

- temperature 
- solar radiation 
- albedo of the glacier
- wind
- ...

Yearly mass balance can be estimated with physical models involving all these parameters.

Physical parameters are hard to get

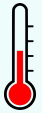
Measuring the physical parameters can be cumbersome.

...sometimes hard to evaluate (e.g. measuring properties of the ice).

Even with the fanciest physical model (that can also be wrong), the results can't be totally accurate.

Use unbiased and easy to measure *proxy* parameters

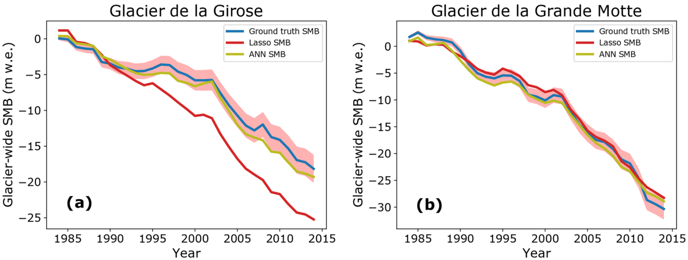
We can design a regression NN model so that we predict the mass balance:

 ,  ,… $\xrightarrow{\approx\text{predicts}}$ 

easy-measured features

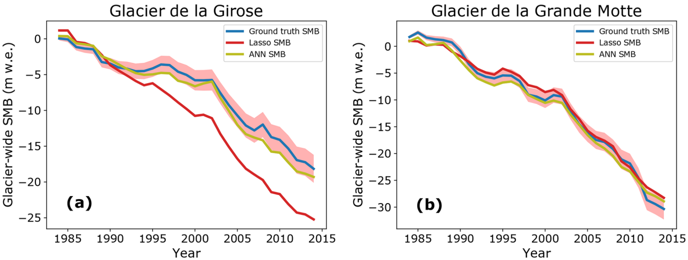
mass balance

Regression glacier model: better than linear



[Bolibar et al. 2020]

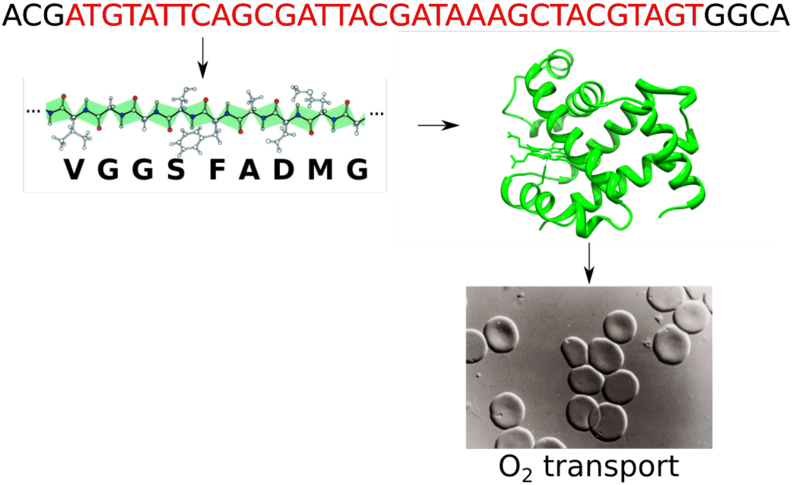
Regression glacier model: better than linear



[Bolibar et al. 2020]

Why not done before? Overfitting was hard to get rid of!

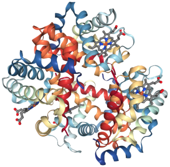
Protein sequence and structure



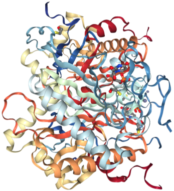
A recent big achievement: protein structure prediction

Goal: predict the structure from sequence

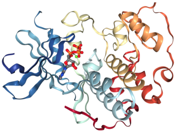
```
>1A3N:A|PDBID|CHAIN|SEQUENCE
VLSPADKTNVKAAWGKVGAHAGEYGAEALER
MFLSFPTTKTYFPHFDLSHGSAQVKGHGKKV
ADALTNAVAHVDDMPNALSALSDLHAHKLRV
DPVNFKLLSHCLLVTLAAHLPAEFTPAVHAS
LDKFLASVSTVLTSKYR
```



```
>1HXP:A|PDBID|CHAIN|SEQUENCE
MTQFNPVDHPHRRYNPLTGQWILVSPHRAKRPW
EGAQETPAKQVLPAHDPDCFLCAGNVRVTGDKN
PDYTGTYVFTNDFAALMSDTPDAPESHDPLMRC
QSARGTSRVICFSPDHSKTLPELSVAALTEIVK
TWQEQTAELGKTYPWVQVFENKGAAMGCSNPHP
HGQIWANSFLPNEAEREDRLQKEYFAEQKSPML
VDYVQRELADGSRTVVETEHWLAVVPYWAAWPF
ETLLLPKAHVLRITDLTDAQRSDLALALKKLTS
RYDNLFQCSFPYSMGWHGAPFNGEENQHWQLHA
HFYPPLLRSATVRKFMVGYEMLAETQRDLTAEQ
AAERLRAVSDIHFRESGV
```



```
>1HCK:A|PDBID|CHAIN|SEQUENCE
MENFQKVEKIGEGTYGVVYKARNKLTGEVVAL
KKIRLDTETEGVPSTAIREISLLKELNHPNIV
KLLDVIHTENKLYLVFEFLHQDLKKFMDASAL
TGIPLPLIKSYLFQLLQGLAFCHSHRVLHRDL
KPQNLLINTEGAIKLADFGLARAFGVPVRTYT
HEVVTLWYRAPEILLGCKYYSTAVDIWSLGCI
FAEMVTRRALFPGDSEIDQLFRIFRTLGTPDE
VVWPGVTSMPDYKPSFPKWARQDFSKVVPPLD
EDGRSLLSQMLHYDPNKRISAKAALAHPFFQD
VTKPVPHLRL
```

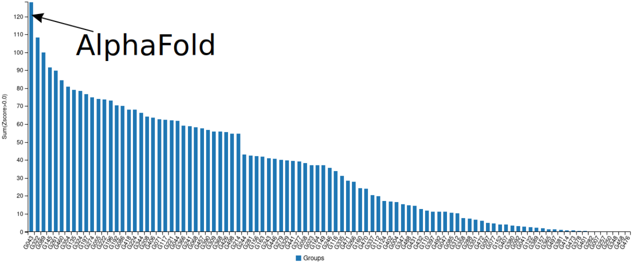


CASP competition

Blind competition. Simple principle:
- a sequence is given
- have to predict the structure.

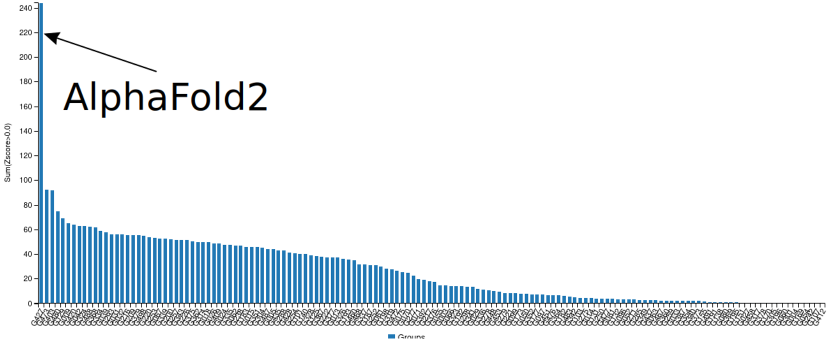
Prior to 2018 it used to be (pseudo) physical models that where best performing.

AI wins the challenge for the first time.



CASP14 (2020)

"The big leap forward"



Alphafold2: attention-based learning on protein sequence alignments
[Casp14.]
Nature's article.

Some fun and worrying examples: deepfake

link to video

Some useful and worrying examples: GPT3 examples

Poor English input: I eated the purple berries.
Good English output: I ate the purple berries.
Poor English input: Thank you for picking me as your designer. I'd appreciate it.
Good English output: Thank you for choosing me as your designer. I appreciate it.
Poor English input: The mentioned changes have done. or I did the alteration that you requested. or I changed things you wanted and did the modifications.
Good English output: The requested changes have been made. or I made the alteration that you requested. or I changed things you wanted and made the modifications.
Poor English input: I'd be more than happy to work with you in another project.
Good English output: I'd be more than happy to work with you on another project.

Poor English input: Please provide me with a short brief of the design you're looking for and that'd be nice if you could share some examples or project you did before.
Good English output: Please provide me with a brief description of the design you're looking for and that would be nice if you could share some examples or projects you have done before.

Poor English input: The patient was died.
Good English output: The patient died.

Poor English input: We think that Leslie likes ourselves.
Good English output: We think that Leslie likes us.

That's useful.👍

Some useful and worrying examples: GPT3 examples

```
Helsinki is the capital and largest city of Finland. It is in the region
of Uusimaa, in southern Finland, on the shore of the Gulf of Finland.
Helsinki has a population of , an urban population of , and a metropolitan
population of over 1.4 million, making it the most populous municipality
and urban area in Finland. Helsinki is some north of Tallinn, Estonia,
east of Stockholm, Sweden, and west of Saint Petersburg, Russia. Helsinki
has close historical connections with these three cities.

The Helsinki metropolitan area includes the urban core of Helsinki, Espoo,
Vantaa, Kauniainen, and surrounding commuter towns. It is the world's
northernmost metro area of over one million people, and the city is the
northernmost capital of an EU member state. The Helsinki metropolitan
area is the third largest metropolitan area in the Nordic countries
after Stockholm and Copenhagen, and the City of Helsinki is the third
largest after Stockholm and Oslo. Helsinki is Finland's major political,
educational, financial, cultural, and research center as well as one of
northern Europe's major cities. Approximately 75% of foreign companies
that operate in Finland have settled in the Helsinki region. The nearby
municipality of Vantaa is the location of Helsinki Airport, with frequent
service to various destinations in Europe and Asia.

Q: what is the most populous municipality in Finland?

A: Helsinki

Q: how many people live there?

A: 1.4 million in the metropolitan area

Q: what percent of the foreign companies that operate in Finland are in
Helsinki?

A: 75%
```

That's impressive. 😮

Some useful and worrying examples: GPT3 examples

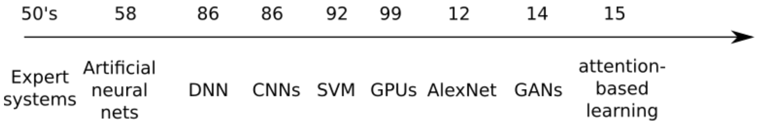
```
Title: United Methodists Agree to Historic Split
Subtitle: Those who oppose gay marriage will form their own denomination
Article: After two days of intense debate, the United Methodist Church
has agreed to a historic split - one that is expected to end in the
creation of a new denomination, one that will be "theologically and
socially conservative," according to The Washington Post. The majority of
delegates attending the church's annual General Conference in May voted to
strengthen a ban on the ordination of LGBTQ clergy and to write new rules
that will "discipline" clergy who officiate at same-sex weddings. But
those who opposed these measures have a new plan: They say they will form a
separate denomination by 2020, calling their church the Christian Methodist
denomination.
The Post notes that the denomination, which claims 12.5 million members, was
in the early 20th century the "largest Protestant denomination in the U.S.,"
but that it has been shrinking in recent decades. The new split will be the
second in the church's history. The first occurred in 1968, when roughly
10 percent of the denomination left to form the Evangelical United Brethren
Church. The Post notes that the proposed split "comes at a critical time
for the church, which has been losing members for years," which has been
"pushed toward the brink of a schism over the role of LGBTQ people in the
church." Gay marriage is not the only issue that has divided the church. In
2016, the denomination was split over ordination of transgender clergy, with
the North Pacific regional conference voting to ban them from serving as
clergy, and the South Pacific regional conference voting to allow them.
```

Figure 3.14: The GPT-3 generated news article that humans had the greatest difficulty distinguishing from a human written article (accuracy: 12%).

That's worrying! 😓

Past and future

Timeline


50's
58
86
86
92
99
12
14
15
Expert systems
Artificial neural nets
DNN
CNNs
SVM
GPUs
AlexNet
GANs
attention-based learning

Biases

- Correlation $\neq$ causality
  - Pneumonia and asthma example [Crawford and Calo 16]
- Minorities in training data
  - less accurate for minorities [demo: beard-face on ResNet50]
- Models can increase biases
  - Underfitting and overfitting
  - preceptron for hiring

AI and CO_2

AI can consumes a lot of electrical energy, having a strong environmental impact. Here are some figures showing the equivalent CO2 emission for creating some famous AI models for natural language processing:

Model	Hardware	Power (W)	Hours	kWh·PUE	CO_2e	Cloud compute cost
$\text{Transformer}_{base}$	P100x8	1415.78	12	27	26	$41–$140
Transformer_{big}	P100x8	1515.43	84	201	192	$289–$981
ELMo	P100x3	517.66	336	275	262	$433–$1472
BERT_{base}	V100x64	12,041.51	79	1507	1438	$3751–$12,571
BERT_{base}	TPUv2x16	—	96	—	—	$2074–$6912
NAS	P100x8	1515.43	274,120	656,347	626,155	$942,973–$3,201,722
NAS	TPUv2x1	—	32,623	—	—	$44,055–$146,848
GPT-2	TPUv3x32	—	168	—	—	$12,902–$43,008

Table 3: Estimated cost of training a model in terms of CO_2 emissions (lbs) and cloud compute cost (USD).[7] Power and carbon footprint are omitted for TPUs due to lack of public information on power draw for this hardware.

[Strubell et al. 19]

Conclusion

Open problems

- (Fully) understand generalization ability of deep NN
- How to improve collaboration/reuse of models in AI?
  - Model distillation (big model → small model)
  - Transfer learning (application A → application B)
- How to reduce learning hassle?
  - Unsupervised learning
  - Few-shot learning
- How to have guarantees?
  - Explainable AI
- How to reduce/remove biases (disentangle correlation and causality)
- How to regulate the creations/usages of AI[12]?

[12] Cannot rely on companies for this. See here.

To sum up: what can I do with DL?

As soon as you have data, either labeled or unlabled, you can learn a model (in particular a DNN).
If there is some *information*[13] in your training set, there is a good chance that the model will learn it.
You can use this model to predict on further data, to take decision, to estimate values, etc.

Note that now, most of the basic tasks (segmentation, image classification, etc.) can be achieved using pre-trained models. You can adapt your model to your specific dataset (few-shot learning).

[13] for instance between the data and the labels, or a structure underlying the data

The good, the bad, and the ugly

What applications would you consider as beneficial or detrimental for the society?

The good, the bad, and the ugly

What applications would you consider as beneficial or detrimental for the society?

- Backfire effect/Jevons paradox 
- Less humanistic considerations
- Biases
- "*Unresponsibilizing*"
- Pushes society toward technology
- Automation of (boring) tasks
- Prevents from human mistakes
- Allow extract (unseen) information from data

Discussion and questions?

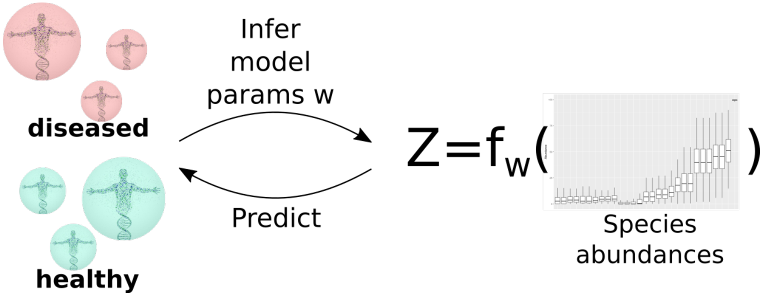
Regularization

Regularization is an important technique that aims at excluding unrealistic parameter combinations.

- Ridge regularization: avoids big values of parameters
- Lasso regularization: favor nullity of parameters (parcimonious model)
- Bayesian modeling: model a priori knowledge on each parameter

Application: example in health

We can build models to predict diseases from microbial abundances, a process known as MWAS:



MWAS as a classification problem

Let:

- $\vec{X}$ be an M-dimensional random vector of abundance of species,
- and Z binary (0/1) random variable describing the disease state of a human.

Define a predictor $f : \mathbb{R}_+^M \to [0, 1]$ such that it minimizes a *loss* on a training set $(\vec{x}_1, z_1), ..., (\vec{x}_N, z_N)$:

MWAS as a classification problem

Let:

- $\vec{X}$ be an M-dimensional random vector of abundance of species,
- and Z binary (0/1) random variable describing the disease state of a human.

Define a predictor $f : \mathbb{R}_+^M \to [0, 1]$ such that it minimizes a *loss* on a training set $(\vec{x}_1, z_1), ..., (\vec{x}_N, z_N)$:

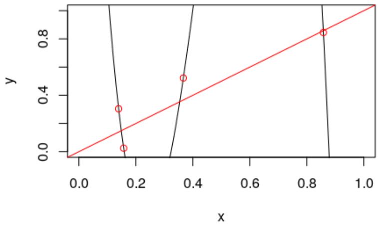
$$\min_f - \sum_{i=1}^{N} z_i . \log f(\vec{x}_i) + (1 - z_i) . \log(1 - f(\vec{x}_i))$$

Regularization

Ridge regularization example

Let's come back to the model $Y = \sum_{i=0}^{3} \beta_i x^i + \epsilon$.

The maximum likelihood with 4 points will give a $\vec{\beta}$ fitting perfectly the points:



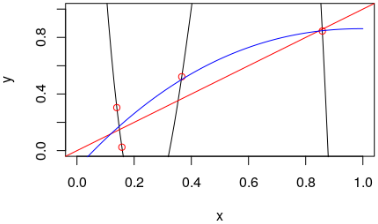
Maximum *likelihood* coefficients:

β_0	β_1	β_2	β_3
5.169	-54.388	155.755	-114.487

Ridge regularization example

Let's come back to the model $Y = \sum_{i=0}^{3} \beta_i x^i + \epsilon$.

With a prior $\mathcal{N}(0, \eta^2)$ the maximum a posteriori of the vector $\vec{\beta}$ corresponds to (blue curve):



Maximum *a posteriori* coefficients

β_0	β_1	β_2	β_3
-0.1279	2.2561	-1.5779	0.3180

Quizz

Overfitting depends on:

- Size of the training set
- Complexity of the problem
- The parametrization of the model
- The type of the model