# Deep Learning & Applied AI

Overfitting and going nonlinear

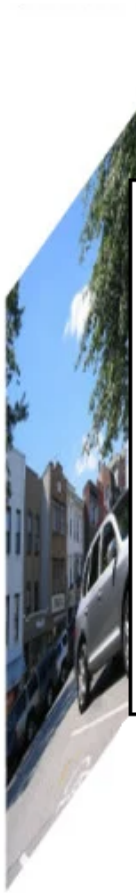
Emanuele Rodolà
rodola@di.uniroma1.it

SAPIENZA
UNIVERSITÀ DI ROMA

2nd semester a.y. 2022/2023 · March 15, 2023

# A glimpse into neural networks

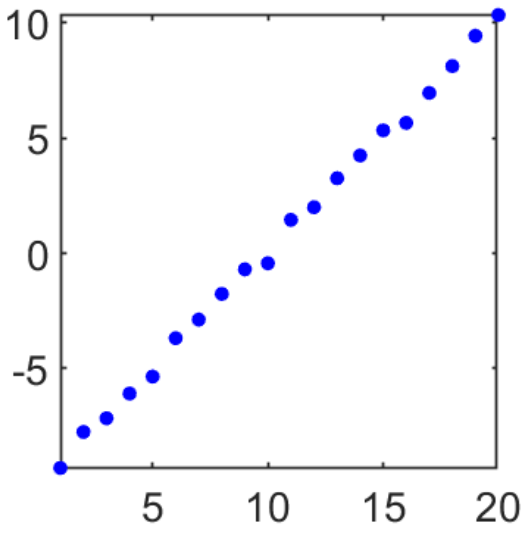
In deep learning, we deal with highly parametrized models called deep neural networks:



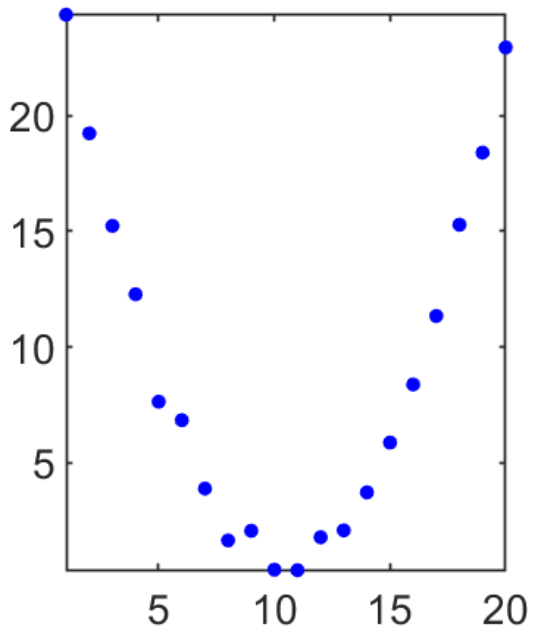
$$f_{\boldsymbol{\Theta}}(\mathbf{x}) = \mathbf{y}$$

# Parametrized models

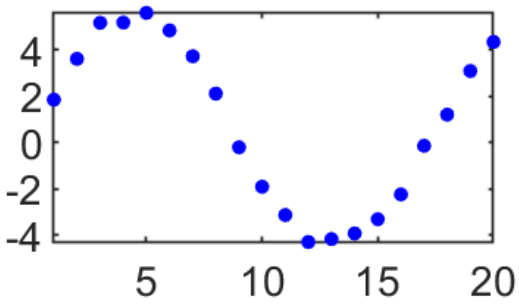
The parameters describe the behavior of the network, and must be solved for.




$y = ax + b$



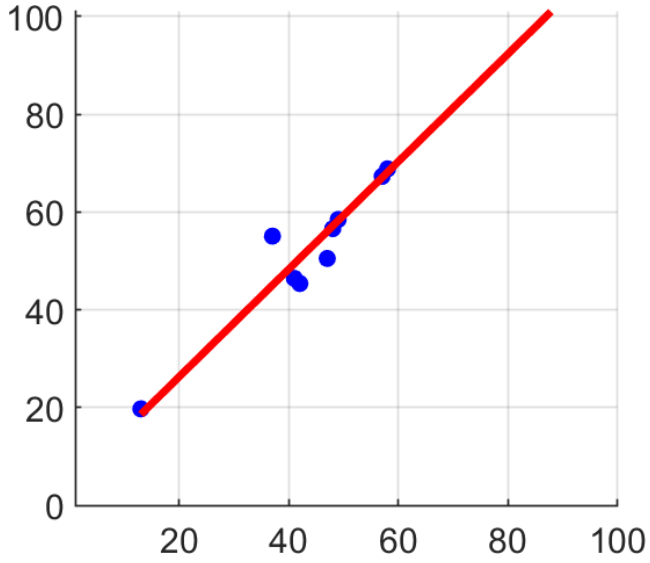

$y = ax^2 + bx + c$




$y = a\sin(x) + bx + c$

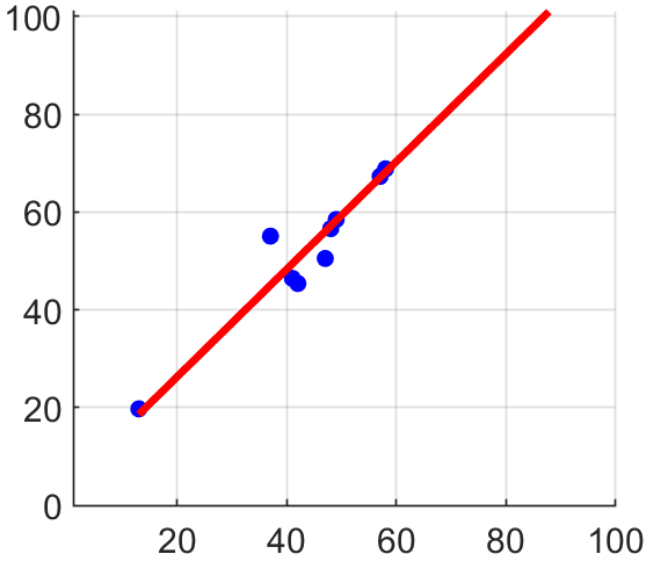
From a technical standpoint, our task is to determine the parameters $\Theta$.

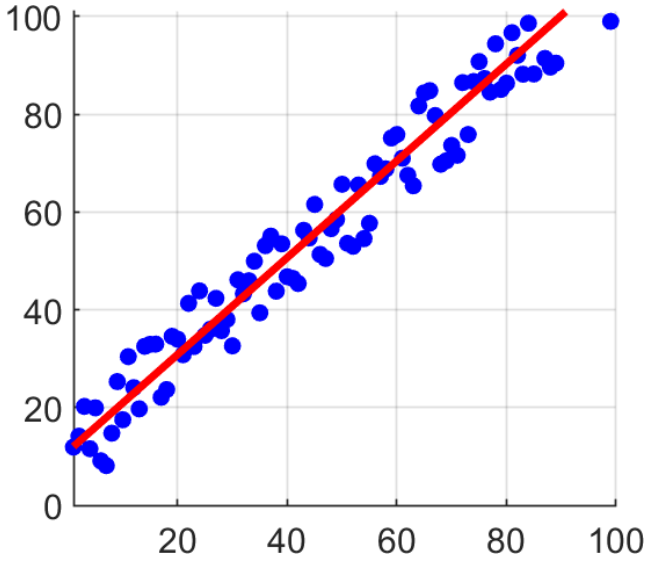
# Data distribution




Assumption: linear model

# Data distribution

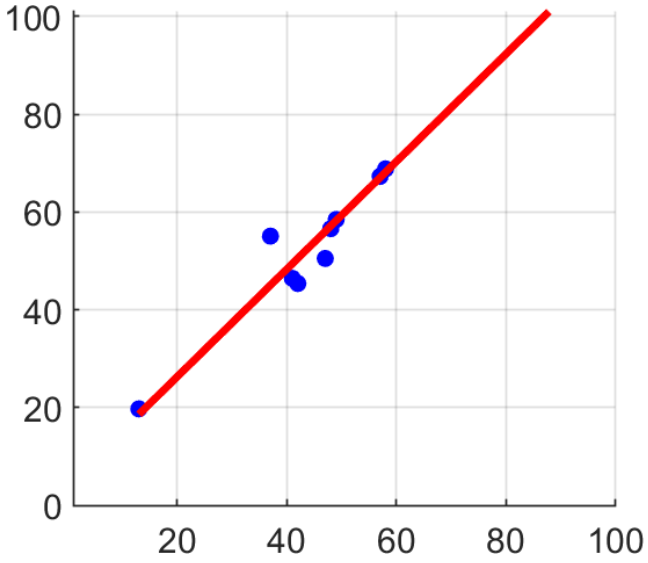


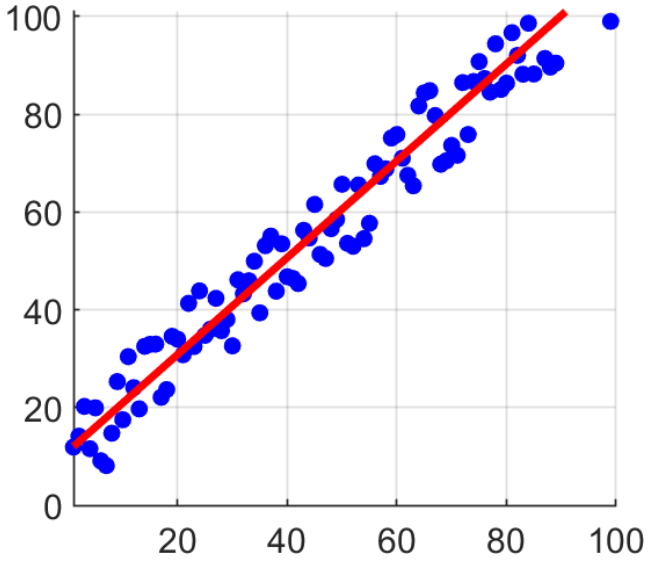




Assumption: linear model

More data allows us to improve our prediction

# Data distribution



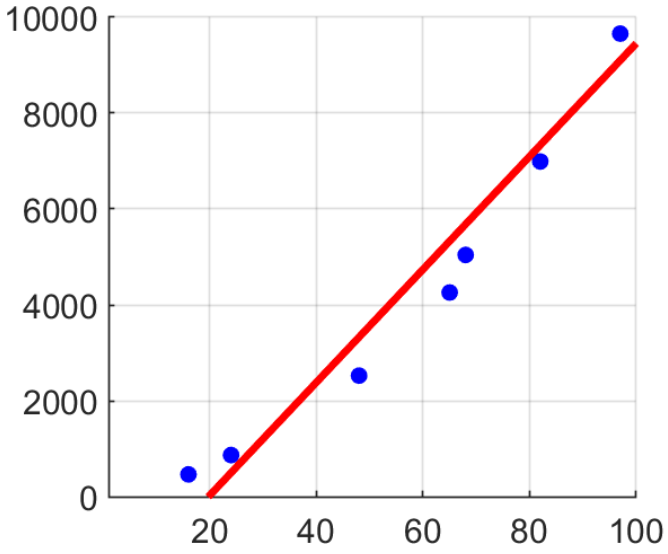




Assumption: linear model

More data allows us to improve our prediction

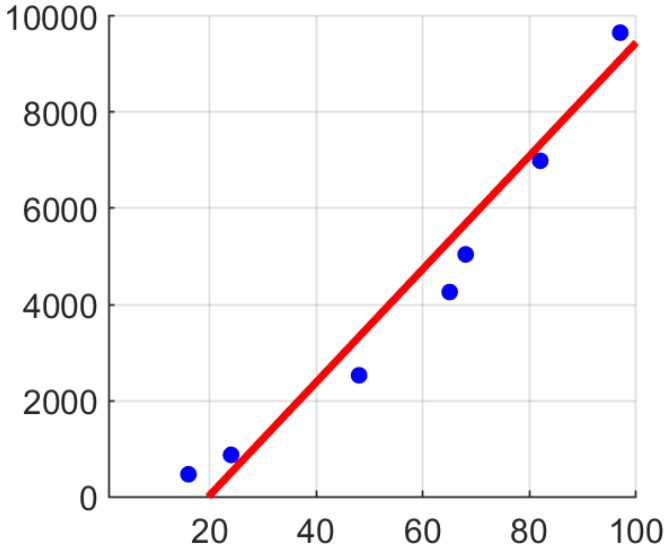
What if the assumption (i.e. linear prior here) is **wrong**?

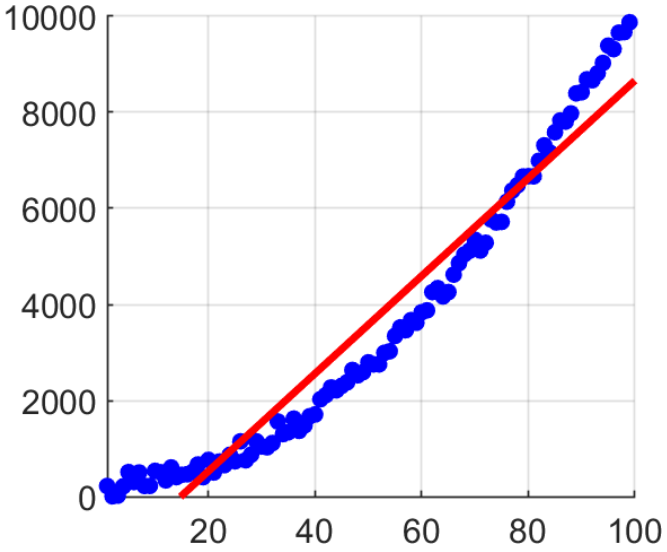
# Data distribution




Assumption: linear model

# Data distribution

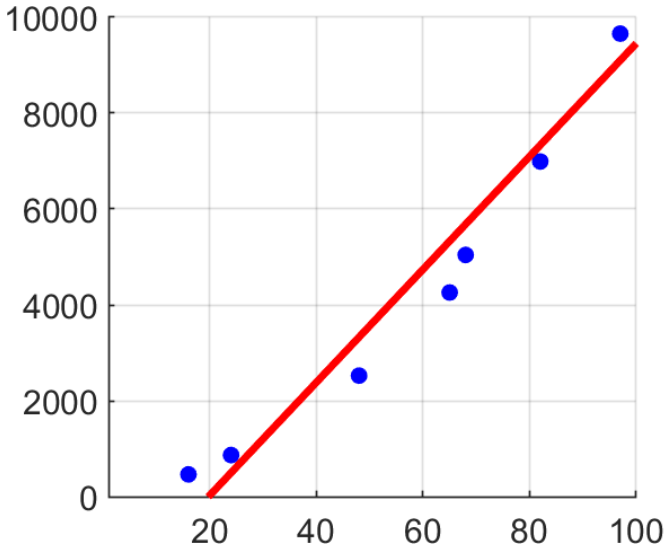


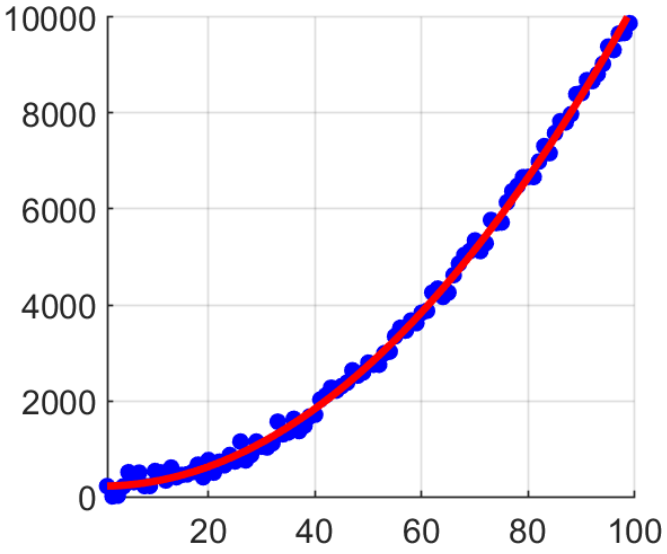




Assumption: linear model

More data confutes our assumptions

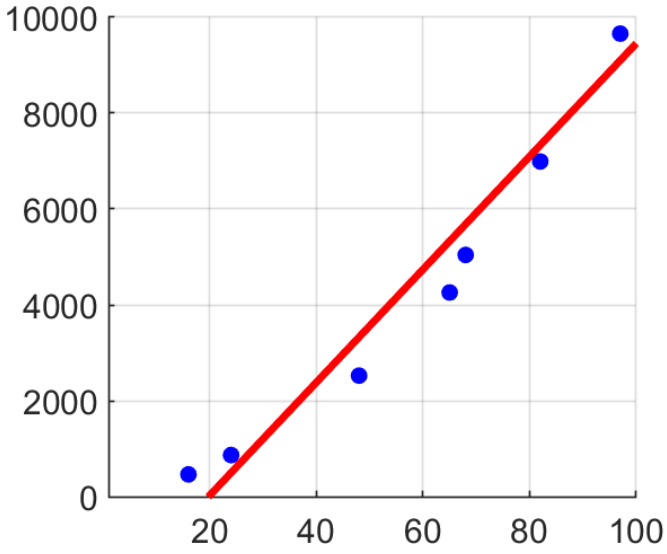
# Data distribution

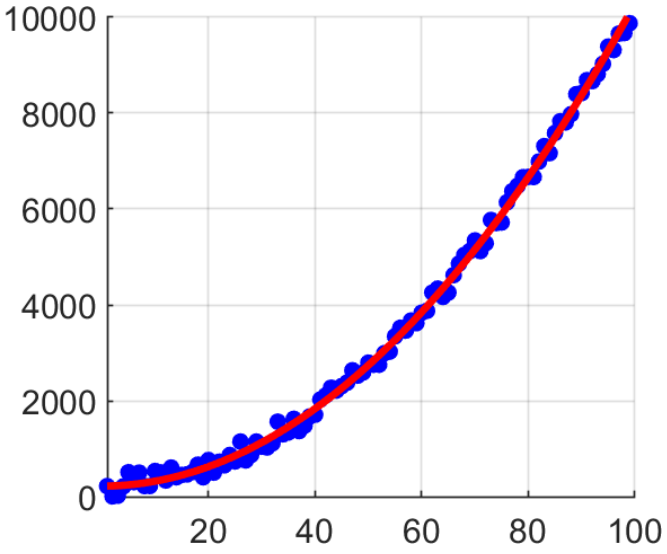



Assumption: quadratic model

# Data distribution



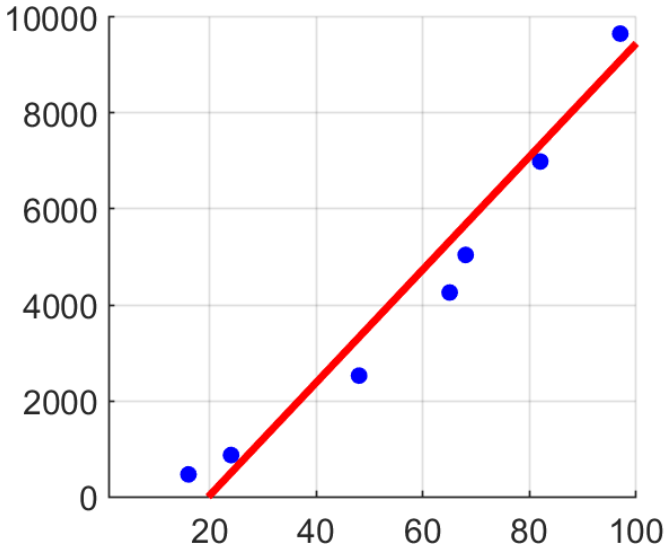



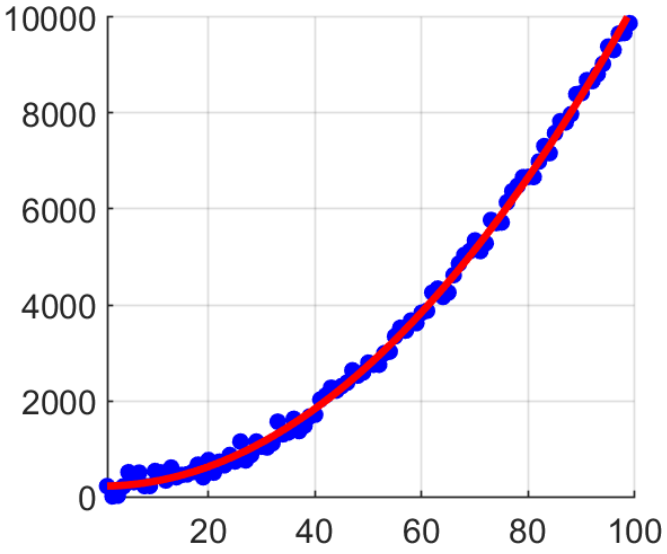

Assumption: quadratic model

Key questions:

- How to select the correct distribution?

# Data distribution



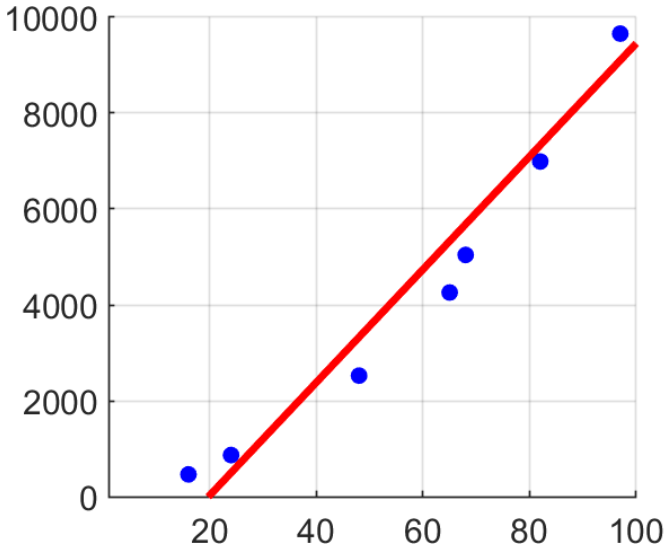



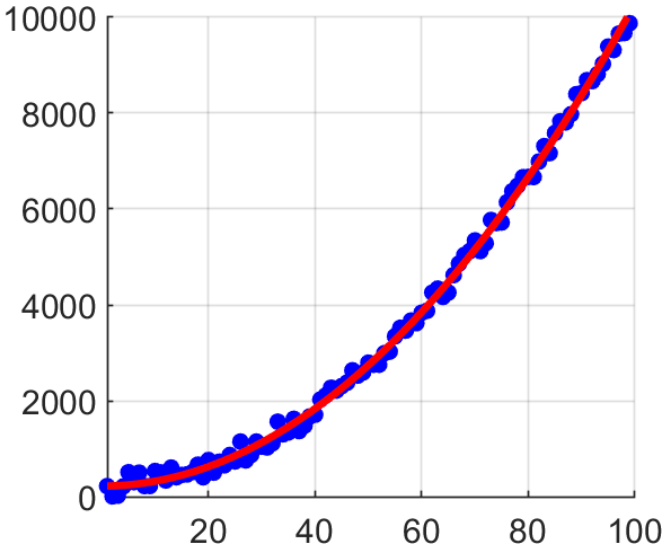

Assumption: quadratic model

Key questions:

- How to select the correct distribution?
- How much data do we need?

# Data distribution




Assumption: quadratic model

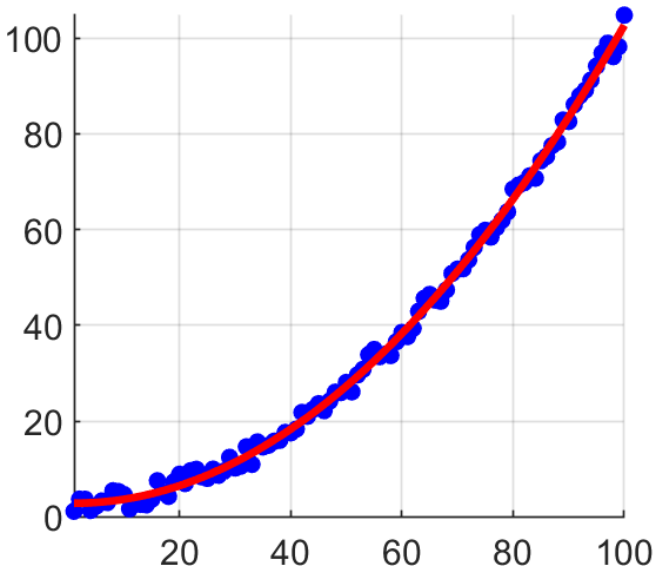
Key questions:

- How to select the correct distribution?
- How much data do we need?
- What if the correct distribution does not admit a simple expression?

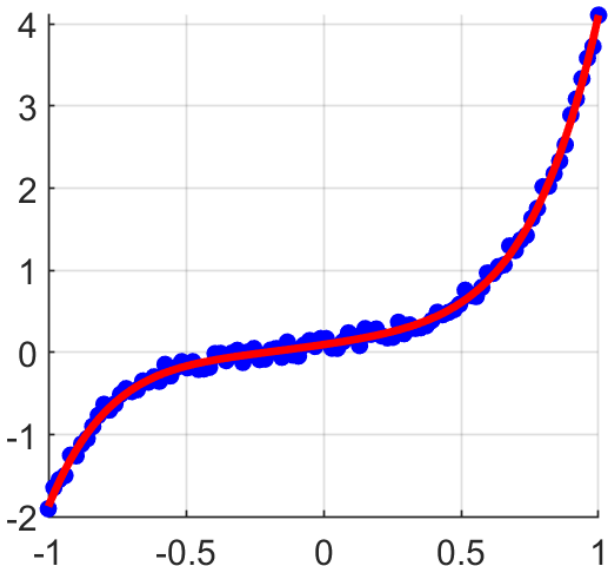
questo ultimo punto è riferito particolarmente al DL

# Polynomial regression

After the linear model, the simplest thing is a polynomial model.



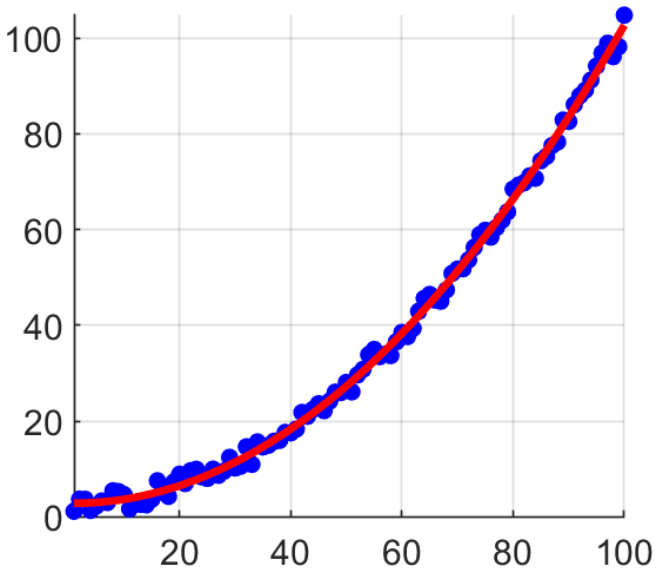

**2nd**-order



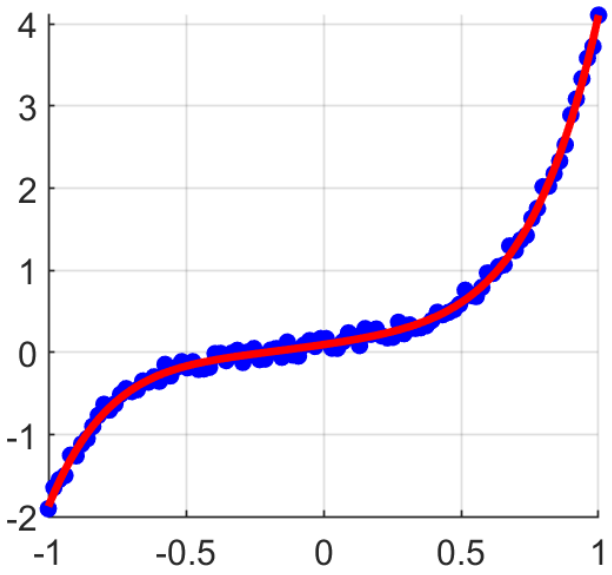

**5th**-order

# Polynomial regression

After the linear model, the simplest thing is a polynomial model.




**2nd**-order

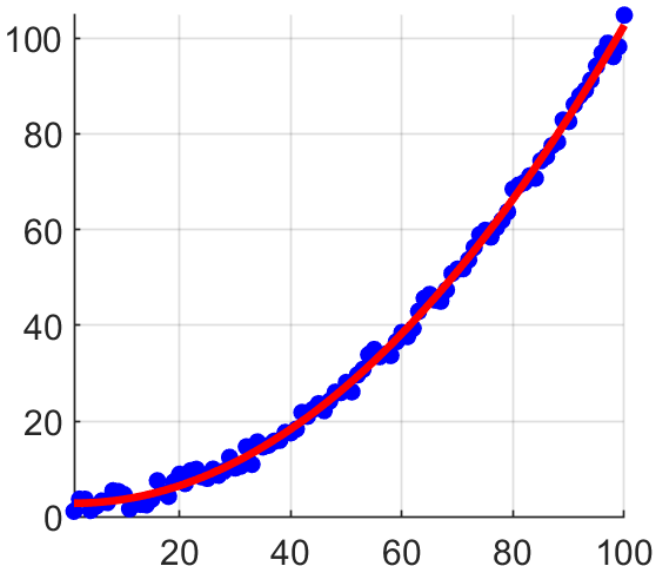



**5th**-order

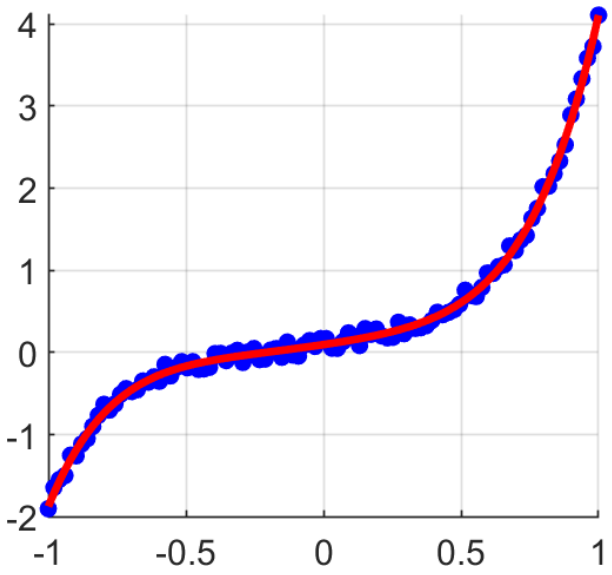
The number of parameters grows with the order.

# Polynomial regression

After the linear model, the simplest thing is a polynomial model.




**2nd**-order




**5th**-order

The number of parameters grows with the order.

More data are needed to make an informed decision on the order.

sembra che sia possibile trovare una curva polinomiale che si adatta bene a qualsiasi problema. In realtà questo è in parte vero, ma le curve polinomiali hanno alcune limitazioni...

# Polynomial regression

attenzione al nome: chiamata così sembra che la funzione non sia lineare, ma invece lo è! Per evitare confusione, si può chiamare...

**Remark:** Despite the name, polynomial regression is still linear in the parameters. It is polynomial with respect to the data.

# Linear regression with polynomial features

**Remark:** Despite the name, polynomial regression is still linear in the parameters. It is polynomial with respect to the data.

# Linear regression with polynomial features

**Remark:** Despite the name, polynomial regression is still linear in the parameters. It is polynomial with respect to the data.

$$y_i = a_3 x_i^3 + a_2 x_i^2 + a_1 x_i + b \quad \text{for all data points } i = 1, \ldots, n$$

# Linear regression with polynomial features

**Remark:** Despite the name, polynomial regression is still linear in the parameters. It is polynomial with respect to the data.

$$y_i = b + \sum_{j=1}^{k} a_j x_i^j \quad \text{for all data points } i = 1, \ldots, n$$

# Linear regression with polynomial features

**Remark:** Despite the name, polynomial regression is still linear in the parameters. It is polynomial with respect to the data.

$$y_i = b + \sum_{j=1}^{k} a_j x_i^j \quad \text{for all data points } i = 1, \ldots, n$$

# Linear regression with polynomial features

**Remark:** Despite the name, polynomial regression is still linear in the parameters. It is polynomial with respect to the data.

$$y_i = b + \sum_{j=1}^{k} a_j x_i^j \quad \text{for all data points } i = 1, \ldots, n$$

In matrix notation:

$$\underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}}_{\mathbf{y}} = \underbrace{\begin{pmatrix} x_1^k & x_1^{k-1} & \cdots & x_1 & 1 \\ x_2^k & x_2^{k-1} & \cdots & x_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^k & x_n^{k-1} & \cdots & x_n & 1 \end{pmatrix}}_{\mathbf{X}} \underbrace{\begin{pmatrix} a_k \\ a_{k-1} \\ \vdots \\ a_1 \\ b \end{pmatrix}}_{\boldsymbol{\theta}}$$

# Linear regression with polynomial features

**Remark:** Despite the name, polynomial regression is still linear in the parameters. It is polynomial with respect to the data.

$$y_i = b + \sum_{j=1}^{k} a_j x_i^j \quad \text{for all data points } i = 1, \ldots, n$$

In matrix notation:

$$\underbrace{\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}}_{\mathbf{y}} = \underbrace{\begin{pmatrix} x_1^k & x_1^{k-1} & \cdots & x_1 & 1 \\ x_2^k & x_2^{k-1} & \cdots & x_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^k & x_n^{k-1} & \cdots & x_n & 1 \end{pmatrix}}_{\mathbf{X}} \underbrace{\begin{pmatrix} a_k \\ a_{k-1} \\ \vdots \\ a_1 \\ b \end{pmatrix}}_{\boldsymbol{\theta}}$$

The same exact least-squares solution as with linear regression applies, with the requirement that $k < n$. la condizione in specifica che bisogna avere sempre almeno più data points del grado della polinomiale

# Polynomial fitting

An application of the Stone-Weierstrass theorem tells us:

If $f$ is continuous on the interval $[a, b]$, then for every $\epsilon > 0$ there exists a polynomial $p$ such that $|f(x) - p(x)| < \epsilon$ for all $x$.

# Polynomial fitting

An application of the Stone-Weierstrass theorem tells us:

If $f$ is continuous on the interval $[a, b]$, then for every $\epsilon > 0$ there exists a polynomial $p$ such that $|f(x) - p(x)| < \epsilon$ for all $x$.

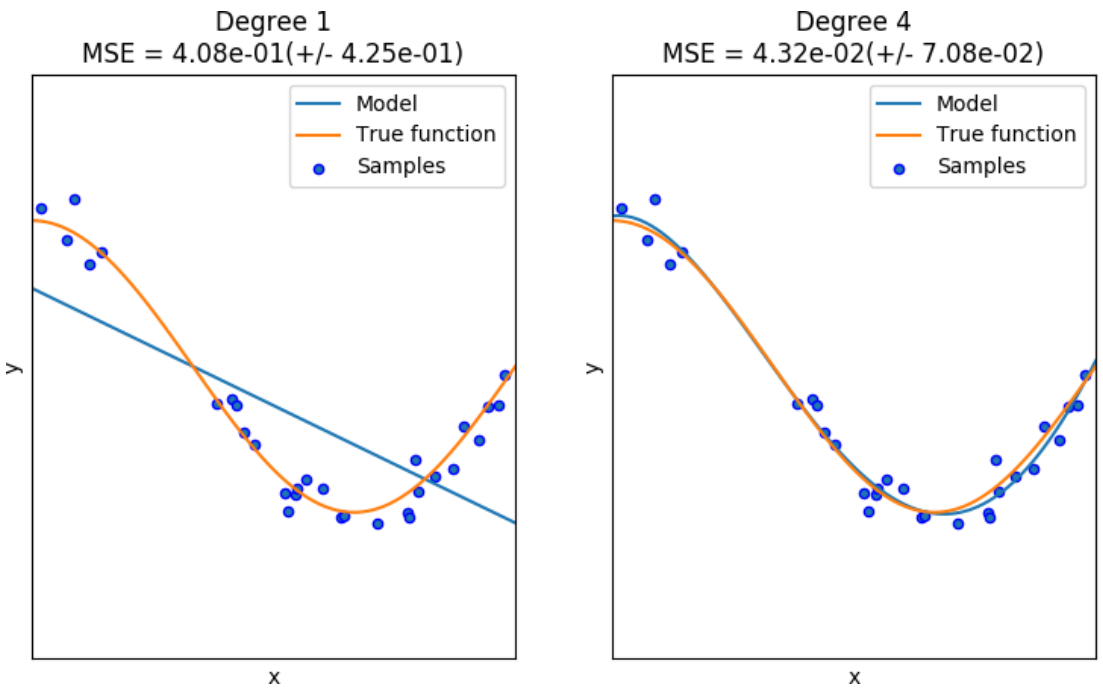
Thus, we can try to fit a polynomial in many cases.

# Polynomial fitting

An application of the Stone-Weierstrass theorem tells us:

If $f$ is continuous on the interval $[a, b]$, then for every $\epsilon > 0$ there exists a polynomial $p$ such that $|f(x) - p(x)| < \epsilon$ for all $x$.

Thus, we can try to fit a polynomial in many cases.



# Polynomial fitting

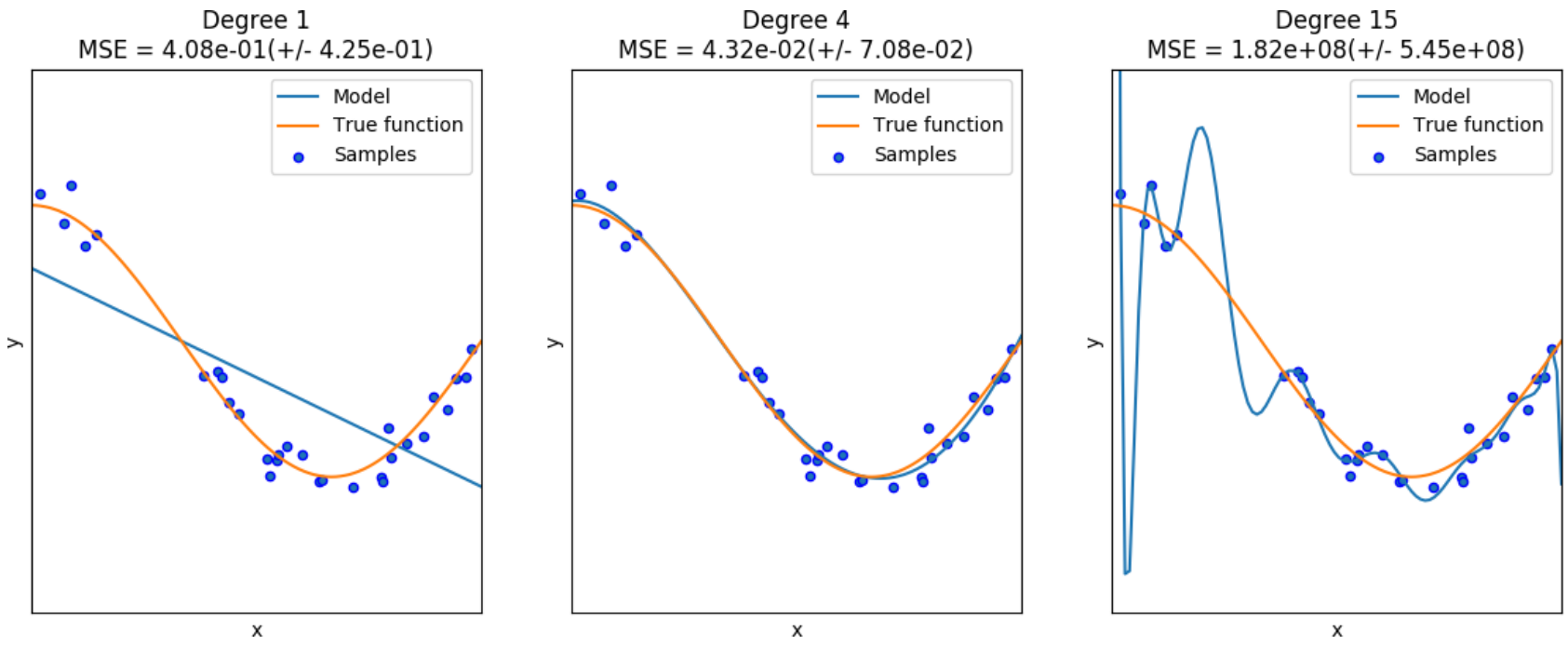
teorema potentissimo

An application of the Stone-Weierstrass theorem tells us:

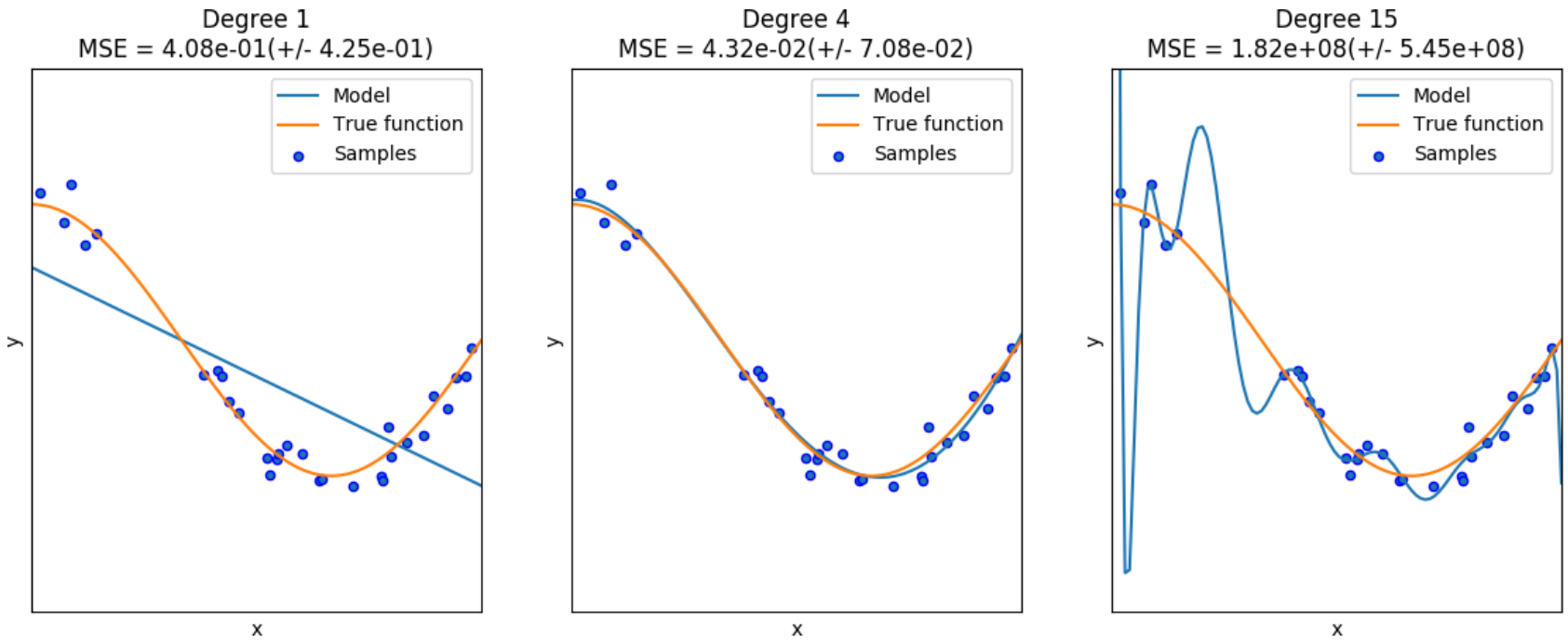
errore

If $f$ is continuous on the interval $[a, b]$, then for every $\epsilon > 0$ there exists a polynomial $p$ such that $|f(x) - p(x)| < \epsilon$ for all $x$.

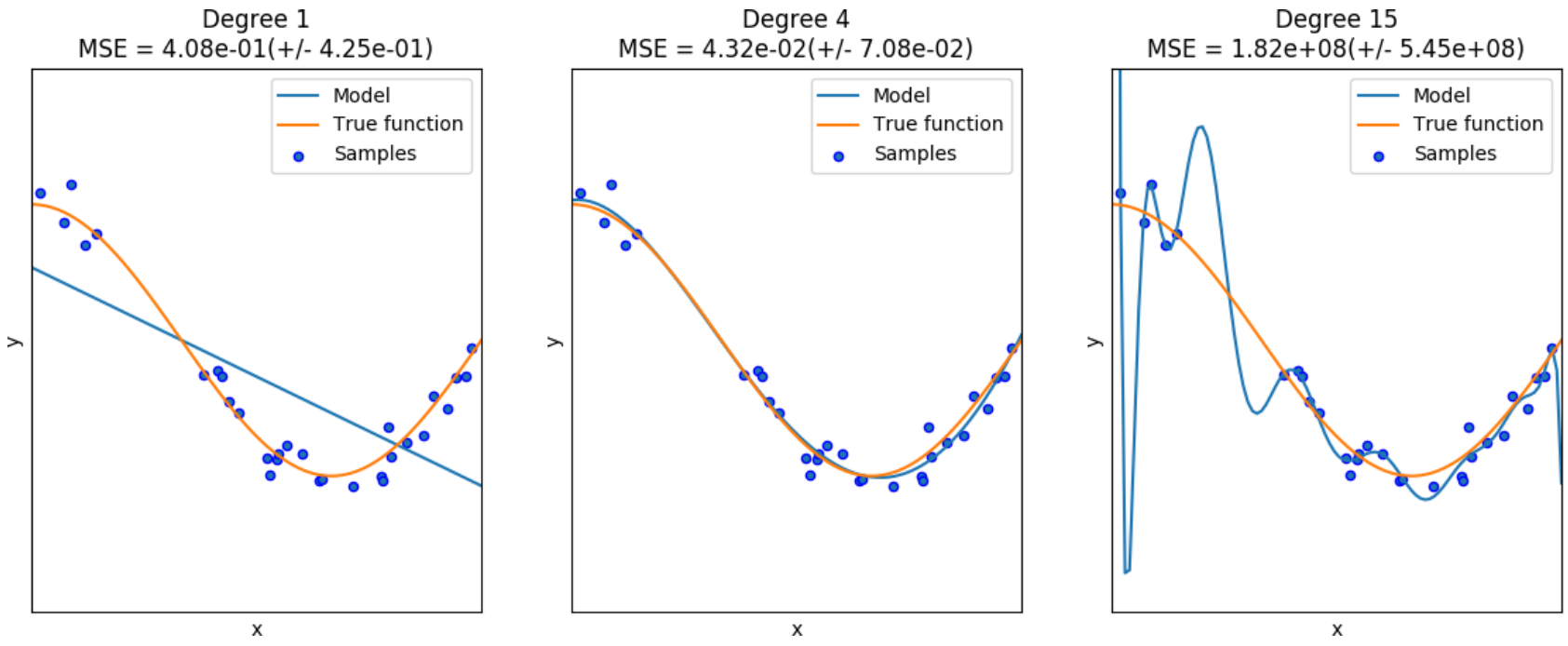
Thus, we can try to fit a polynomial in many cases.



# Underfitting vs. Overfitting

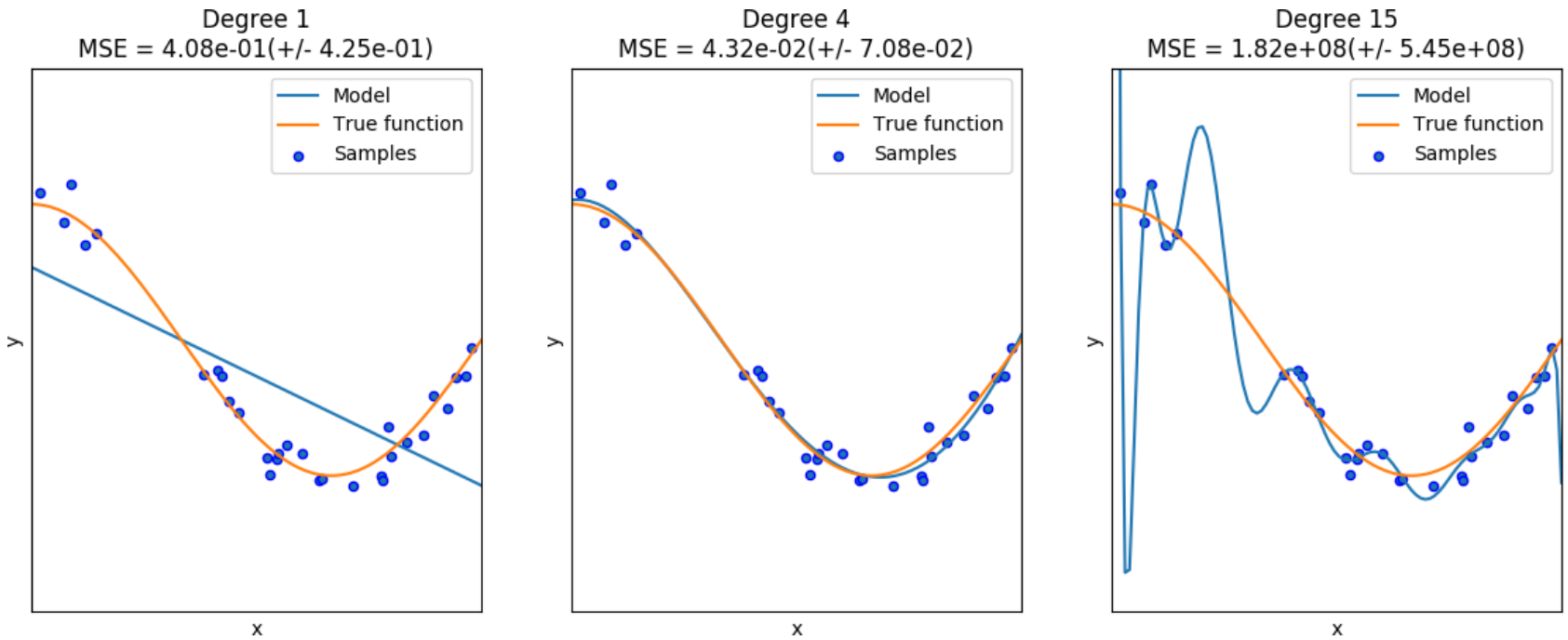


# Underfitting vs. Overfitting



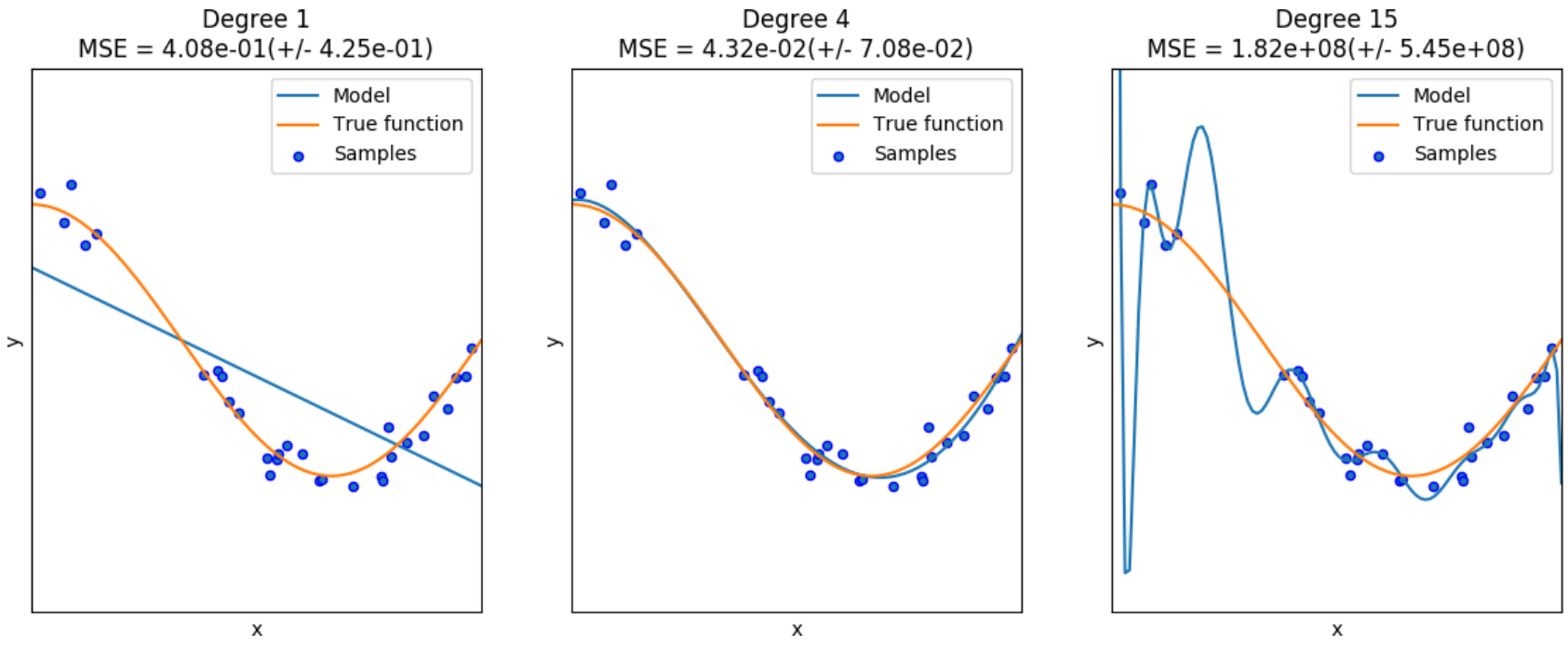

- Underfitting: Not sufficiently fitting the data (large MSE).

# Underfitting vs. Overfitting




- Underfitting: Not sufficiently fitting the data (large MSE).
- Overfitting: We are "learning the noise" (small MSE).

# Underfitting vs. Overfitting




- Underfitting: Not sufficiently fitting the data (large MSE).
- Overfitting: We are "learning the noise" (small MSE).

Adding complexity can lead to overfitting and thus worse generalization.

# Underfitting vs. Overfitting

This trade-off is always present, and still an open problem.

Different mechanisms defend us from under- and overfitting.

# Underfitting vs. Overfitting

This trade-off is always present, and still an open problem.

Different mechanisms defend us from under- and overfitting.

Detection is relatively easier:

1. Estimate the model parameters on a training set.
   (the MSE is minimized on example data)

# Underfitting vs. Overfitting

This trade-off is always present, and still an open problem.

Different mechanisms defend us from under- and overfitting.

Detection is relatively easier:

1. Estimate the model parameters on a training set.
   (the MSE is minimized on example data)
2. Large MSE on the training $\Rightarrow$ underfitting

# Underfitting vs. Overfitting

This trade-off is always present, and still an open problem.

Different mechanisms defend us from under- and overfitting.

Detection is relatively easier:

1. Estimate the model parameters on a training set.
   (the MSE is minimized on example data)
2. Large MSE on the training $\Rightarrow$ underfitting
3. Small MSE on the training $\Rightarrow$
   Apply the model parameters to a validation set.
   (the MSE is computed on different example data)

# Underfitting vs. Overfitting

This trade-off is always present, and still an open problem.

Different mechanisms defend us from under- and overfitting.

Detection is relatively easier:

1. Estimate the model parameters on a training set.
   (the MSE is minimized on example data)
2. Large MSE on the training $\Rightarrow$ underfitting
3. Small MSE on the training $\Rightarrow$
   Apply the model parameters to a validation set.
   (the MSE is computed on different example data)
4. Large MSE on the validation $\Rightarrow$ overfitting

# Underfitting vs. Overfitting

This trade-off is always present, and still an open problem.

Different mechanisms defend us from under- and overfitting.

Detection is relatively easier:

1. Estimate the model parameters on a training set.
   (the MSE is minimized on example data)
2. Large MSE on the training $\Rightarrow$ underfitting
3. Small MSE on the training $\Rightarrow$
   Apply the model parameters to a validation set.
   (the MSE is computed on different example data)
4. Large MSE on the validation $\Rightarrow$ overfitting $\Rightarrow$ bad generalization

# Underfitting vs. Overfitting

Underfitting:
large training error, large validation error

# Underfitting vs. Overfitting

Underfitting:
large training error, large validation error

Overfitting:
(very) small training error, large validation error

# Not done yet

"If $f$ is continuous on the interval $[a, b]$, then for every $\epsilon > 0$ there exists a polynomial $p$ such that $|f(x) - p(x)| < \epsilon$ for all $x$."

So is polynomial regression all we need?

# Not done yet

"If $f$ is continuous on the interval $[a, b]$, then for every $\epsilon > 0$ there exists a polynomial $p$ such that $|f(x) - p(x)| < \epsilon$ for all $x$."

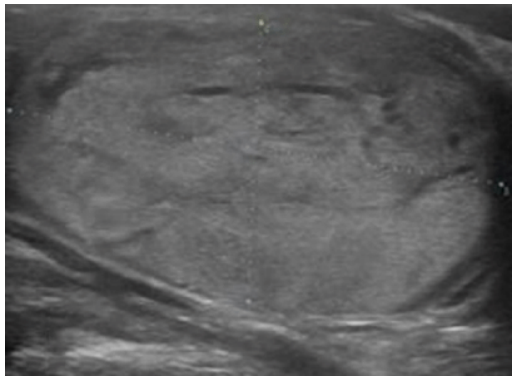
So is polynomial regression all we need?

Not really!

- Different loss than MSE
- Regularization
- Additional priors se si conoscono altre informazioni sui dati a priori, queste rendono difficile l'individuazione della soluzione polinomiale
- Intermediate features le reti neurali, durante il processo di allenamento producono le feature intermedie che sono molto utili. I modelli polinomiali non permettono ciò
- Flexibility
- Regression (predict a value) vs. classification (predict a category)

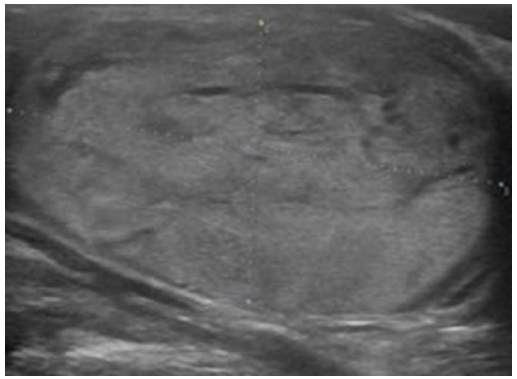
# Classification

What if we want to predict a category instead of a value?

$f_\Theta($  $) = \{0, 1\}$

# Classification

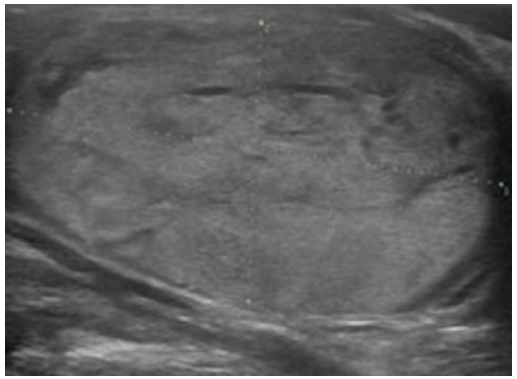
What if we want to predict a category instead of a value?

$f_\Theta($  $) = \{0, 1\}$

Possible solution: Do post-processing (e.g. thresholding) to convert linear regression to a binary output.

# Classification

What if we want to predict a category instead of a value?

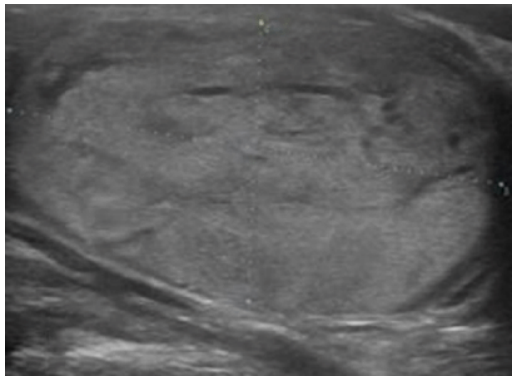
$f_\Theta($  $) = \{0, 1\}$

Possible solution: Do post-processing (e.g. thresholding) to convert linear regression to a binary output.

$\Rightarrow$ The solution is not necessarily an optimum anymore.

# Classification

What if we want to predict a category instead of a value?

$f_\Theta($  $) = \{0, 1\}$

Possible solution: Do post-processing (e.g. thresholding) to convert linear regression to a binary output.

$\Rightarrow$ The solution is not necessarily an optimum anymore.

Instead: Modify the loss to minimize over categorical values directly.

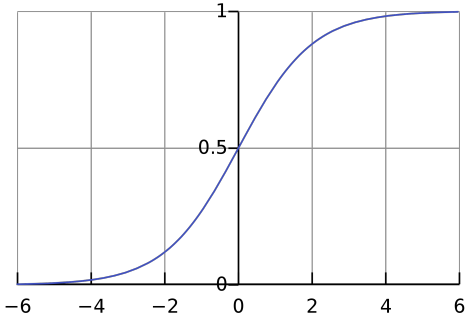
# Logistic regression

New loss:

$$\ell_\Theta(\{x_i, y_i\}) = \sum_{i=1}^{n} (y_i - \sigma(\underbrace{ax_i + b}_{\text{linear}}))^2$$

Here, $\sigma$ is the nonlinear logistic sigmoid:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



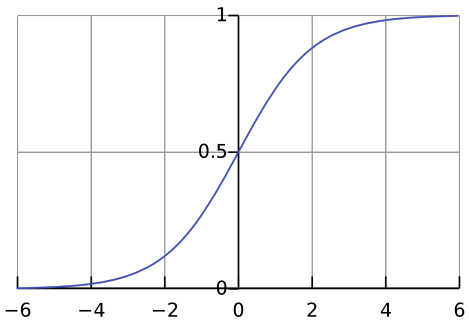
# Logistic regression

New loss:

$$\ell_\Theta(\{x_i, y_i\}) = \sum_{i=1}^{n} (y_i - \sigma(\underbrace{ax_i + b}_{\text{linear}}))^2$$

Here, $\sigma$ is the nonlinear logistic sigmoid:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$




$\sigma$ has a saturation effect as it maps $\mathbb{R} \mapsto (0, 1)$.

# Logistic regression

New loss:

in pratica applica il tresholding (dato dalla sigmoide)
all'interno della loss function stessa

$$\ell_\Theta(\{x_i, y_i\}) = \sum_{i=1}^{n} (y_i - \sigma(\underbrace{ax_i + b}_{\text{linear}}))^2$$

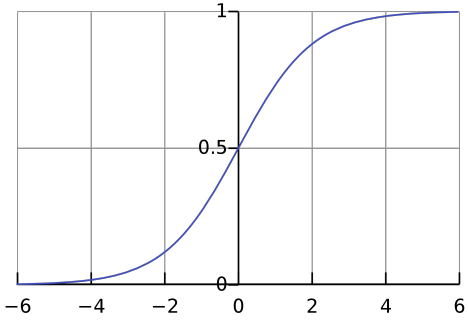
non-convex

anche detta **saturation function** o **equalizer**

Here, $\sigma$ is the nonlinear logistic sigmoid:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

se si moltiplica x per un coeff,
la curva diventa più ripida
(quindi cambia più in fretta)




$\sigma$ has a saturation effect as it maps $\mathbb{R} \mapsto (0, 1)$.

# Logistic regression

New loss: quindi ora la loss function è così

$$\ell_\Theta(\{x_i, y_i\}) = \sum_{i=1}^{n} c(x_i, y_i)\,, \quad \text{with}$$

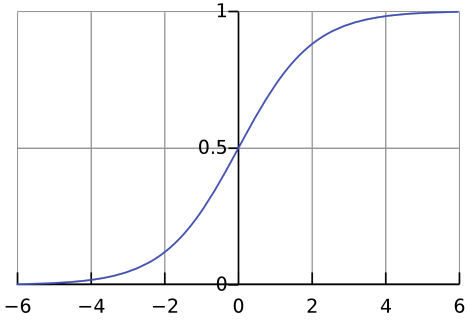
$$c(x_i, y_i) = \begin{cases} -\ln(\sigma(ax_i + b)) & y_i = 1 \\ -\ln(1 - \sigma(ax_i + b)) & y_i = 0 \end{cases}$$

convex

scritta in formula...

Here, $\sigma$ is the nonlinear logistic sigmoid:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$




$\sigma$ has a saturation effect as it maps $\mathbb{R} \mapsto (0, 1)$.

# Logistic regression

New loss:

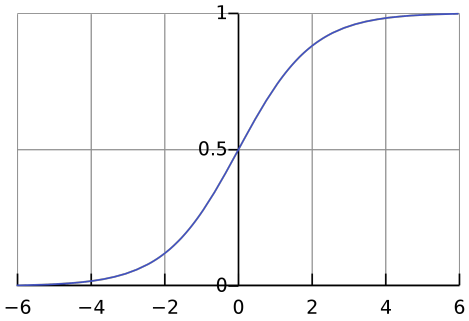
$$\ell_\Theta(\{x_i, y_i\}) = \sum_{i=1}^{n} c(x_i, y_i)\,, \quad \text{with}$$

$$c(x_i, y_i) = -y_i \ln(\sigma(ax_i + b)) - (1 - y_i)\ln(1 - \sigma(ax_i + b))$$ nvex

1 se giusto 0 se è giusto ...è così

Here, $\sigma$ is the nonlinear logistic sigmoid:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$




$\sigma$ has a saturation effect as it maps $\mathbb{R} \mapsto (0, 1)$.

# Logistic regression

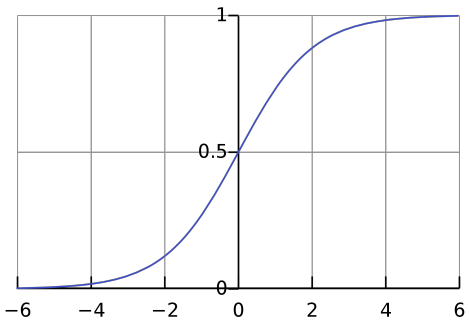
New convex loss:

è la **cross-entropy loss**

$$\ell_\Theta(\{x_i, y_i\}) = -\sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i)\ln(1 - \sigma(ax_i + b))$$

Here, $\sigma$ is the nonlinear logistic sigmoid:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$




$\sigma$ has a saturation effect as it maps $\mathbb{R} \mapsto (0, 1)$.

# Logistic regression: Finding a solution

(taylor?)

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \ell_\Theta = 0$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

ci va il -

where $\Theta = \{a, b\}$.

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$\nabla_\Theta \left( y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) \right)$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$\nabla_\Theta y_i \ln(\sigma(ax_i + b)) + \nabla_\Theta (1 - y_i) \ln(1 - \sigma(ax_i + b))$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \ln(\sigma(ax_i + b)) + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial h} \cdot \frac{\partial h}{\partial a}$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial h} \cdot \frac{\partial}{\partial a} a x_i + b$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \frac{\partial g}{\partial h} \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \frac{\partial \sigma(ax_i + b)}{\partial (ax_i + b)} \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \frac{\partial}{\partial(ax_i + b)} \frac{1}{1 + e^{-(ax_i + b)}} \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \frac{e^{-(ax_i + b)}}{(1 + e^{-(ax_i + b)})^2} \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \frac{1}{1 + e^{-(ax_i + b)}} \frac{e^{-(ax_i + b)}}{1 + e^{-(ax_i + b)}} \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \frac{1}{1 + e^{-(ax_i + b)}} \frac{(1 + e^{-(ax_i + b)}) - 1}{1 + e^{-(ax_i + b)}} \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \frac{1}{1 + e^{-(ax_i + b)}} (1 - \frac{1}{1 + e^{-(ax_i + b)}}) \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \sigma(ax_i + b)(1 - \sigma(ax_i + b)) \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial f}{\partial g} \cdot \sigma(ax_i + b)(1 - \sigma(ax_i + b)) \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{\partial \ln(\sigma(ax_i + b))}{\partial \sigma(ax_i + b)} \cdot \sigma(ax_i + b)(1 - \sigma(ax_i + b)) \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{1}{\sigma(ax_i + b)} \cdot \sigma(ax_i + b)(1 - \sigma(ax_i + b)) \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} f(g(h(a, b))) = \frac{1}{\sigma(ax_i + b)} \cdot \sigma(ax_i + b)(1 - \sigma(ax_i + b)) \cdot x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} \ln(\sigma(ax_i + b)) = (1 - \sigma(ax_i + b))x_i$$

# Logistic regression: Finding a solution

Since the loss is convex, the first-order conditions apply:

$$\nabla_\Theta \sum_{i=1}^{n} y_i \ln(\sigma(ax_i + b)) + (1 - y_i) \ln(1 - \sigma(ax_i + b)) = 0$$

where $\Theta = \{a, b\}$.

Consider the gradient of each term in the summation:

$$y_i \nabla_\Theta \underbrace{\ln(\sigma(ax_i + b))}_{f(g(h(\Theta)))} + (1 - y_i) \nabla_\Theta \ln(1 - \sigma(ax_i + b))$$

Apply the chain rule to each partial derivative:

$$\frac{\partial}{\partial a} \ln(\sigma(ax_i + b)) = (1 - \sigma(ax_i + b))x_i$$

And similarly for the second term and for parameter $b$.

# Logistic regression: Finding a solution

By looking at the partial derivative:

$$\frac{\partial}{\partial a} \ln(\sigma(ax_i + b)) = (1 - \sigma(ax_i + b))x_i$$

we see that the parameters enter the gradient in a nonlinear way.

# Logistic regression: Finding a solution

By looking at the partial derivative:

$$\frac{\partial}{\partial a} \ln(\sigma(ax_i + b)) = (1 - \sigma(ax_i + b))x_i$$

we see that the parameters enter the gradient in a nonlinear way.

Thus:

- $\nabla \ell_\Theta = 0$ is not a linear system that we can solve easily.

# Logistic regression: Finding a solution

By looking at the partial derivative:

$$\frac{\partial}{\partial a} \ln(\sigma(ax_i + b)) = (1 - \sigma(ax_i + b))x_i$$

we see that the parameters enter the gradient in a nonlinear way.

Thus:

- $\nabla \ell_\Theta = 0$ is not a linear system that we can solve easily.
- $\nabla \ell_\Theta = 0$ is a transcendental equation $\Rightarrow$ no analytical solution.

# Logistic regression: Finding a solution

By looking at the partial derivative:

$$\frac{\partial}{\partial a} \ln(\sigma(ax_i + b)) = (1 - \sigma(ax_i + b))x_i$$

we see that the parameters enter the gradient in a nonlinear way.

Thus:

- $\nabla \ell_\Theta = 0$ is not a linear system that we can solve easily.
- $\nabla \ell_\Theta = 0$ is a transcendental equation $\Rightarrow$ no analytical solution.

| **model** | **loss** | **solution** |
|---|---|---|
| linear regression<br>linear regression + Tikhonov<br>logistic regression | | |

# Logistic regression: Finding a solution

By looking at the partial derivative:

$$\frac{\partial}{\partial a} \ln(\sigma(ax_i + b)) = (1 - \sigma(ax_i + b))x_i$$

we see that the parameters enter the gradient in a nonlinear way.

Thus:

- $\nabla \ell_\Theta = 0$ is not a linear system that we can solve easily.
- $\nabla \ell_\Theta = 0$ is a transcendental equation $\Rightarrow$ no analytical solution.

| **model** | **loss** | **solution** |
|---|---|---|
| linear regression | convex | |
| linear regression + Tikhonov | convex | |
| logistic regression | convex | |

# Logistic regression: Finding a solution

By looking at the partial derivative:

$$\frac{\partial}{\partial a} \ln(\sigma(ax_i + b)) = (1 - \sigma(ax_i + b))x_i$$

we see that the parameters enter the gradient in a nonlinear way.

Thus:

- $\nabla \ell_\Theta = 0$ is not a linear system that we can solve easily.
- $\nabla \ell_\Theta = 0$ is a transcendental equation $\Rightarrow$ no analytical solution.

| **model** | **loss** | **solution** |
|---|---|---|
| linear regression | convex | least squares |
| linear regression + Tikhonov | convex | |
| logistic regression | convex | |

# Logistic regression: Finding a solution

By looking at the partial derivative:

$$\frac{\partial}{\partial a} \ln(\sigma(ax_i + b)) = (1 - \sigma(ax_i + b))x_i$$

we see that the parameters enter the gradient in a nonlinear way.

Thus:

- $\nabla \ell_\Theta = 0$ is not a linear system that we can solve easily.
- $\nabla \ell_\Theta = 0$ is a transcendental equation $\Rightarrow$ no analytical solution.

| **model** | **loss** | **solution** |
|---|---|---|
| linear regression | convex | least squares |
| linear regression + Tikhonov | convex | least squares |
| logistic regression | convex | |

# Logistic regression: Finding a solution

By looking at the partial derivative:

$$\frac{\partial}{\partial a} \ln(\sigma(ax_i + b)) = (1 - \sigma(ax_i + b))x_i$$

we see that the parameters enter the gradient in a nonlinear way.

Thus:

- $\nabla \ell_\Theta = 0$ is not a linear system that we can solve easily.
- $\nabla \ell_\Theta = 0$ is a transcendental equation $\Rightarrow$ no analytical solution.

quindi non c'è una soluzione per a e b

| **model** | **loss** | **solution** |
|---|---|---|
| linear regression | convex | least squares |
| linear regression + Tikhonov | convex | least squares |
| logistic regression | convex | nonlinear optimization |

matrice di regolarizzazione nella minimizzazione del MSE (anche detta **ridge regression**)

che sarebbe GD

# Suggested reading

On polynomial regression vs. neural nets:
https://arxiv.org/pdf/1806.06850

Proof that the logistic loss is convex:
https://math.stackexchange.com/questions/1582452/logistic-regression-prove-that-the-cost-function-is-convex