Optimization Algorithms for Machine Learning

Stefano Gualandi
Università di Pavia, Dipartimento di Matematica

email: stefano.gualandi@unipv.it
twitter: @famo2spaghi
blog: http://stegua.github.com
web: http://matematica.unipv.it/gualandi/opt4ml

What is Machine Learning?

Is there a dog in the photo?

$\Rightarrow \left\{ \begin{array}{l} \text{YES} \\ \text{NO} \end{array} \right.$

Is there a dog in the photo?

$\Rightarrow \Bigg\{$ YES!

Is there a dog in the photo?

$$\Rightarrow \quad \Bigg\{$$

Is there a dog in the photo?

$\Rightarrow \Big\{$ YES!

Is there a dog in the photo?

$\Rightarrow \Bigg\{$

Is there a dog in the photo?

$\Rightarrow \Big\{$ NO!

Is there a dog in the photo?

$$\Rightarrow \left\{ \right.$$

Is there a dog in the photo?

$\Rightarrow \Big\{$ $\boxed{\text{YES!}}$

Is there a dog in the photo?

$\Rightarrow \Big\{$

Is there a dog in the photo?

$\Rightarrow \Big\{$ NO!

Example 1: Classification Problem

$\Rightarrow \left\{ \begin{array}{l} \text{YES} \\ \text{NO} \end{array} \right.$

Example 1: Classification Problem

$\Rightarrow \left\{ \right.$ YES / NO

$\boldsymbol{x} \in X$: Input data

$y \in Y$: Output/Target

Example 1: Classification Problem

$\Rightarrow$ $\left\{ \begin{array}{l} \text{YES} \\ \text{NO} \end{array} \right.$

$\boldsymbol{x} \in X$: Input data

$y \in Y$: Output/Target

GOAL: Find a function

$$f : X \to Y$$

that maps each input to the *correct* target.

Training Set and Data Set

Let us take $Y = \{-1, 1\}$, with $y = 1$ means **YES**, and $y = -1$ means **NO**.

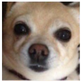
Training Set and Data Set

Let us take $Y = \{-1, 1\}$, with $y = 1$ means **YES**, and $y = -1$ means **NO**.

$\boldsymbol{x}_1 =$  $y_1 = 1$

$\boldsymbol{x}_2 =$  $y_2 = 1$

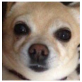
$\boldsymbol{x}_3 =$  $y_3 = -1$

$\boldsymbol{x}_4 =$  $y_4 = 1$

$\boldsymbol{x}_5 =$  $y_5 = -1$

Training Set and Data Set

Let us take $Y = \{-1, 1\}$, with $y = 1$ means **YES**, and $y = -1$ means **NO**.

$\boldsymbol{x}_1 =$  $\boldsymbol{x}_2 =$  $\boldsymbol{x}_3 =$  $\boldsymbol{x}_4 =$  $\boldsymbol{x}_5 =$ 

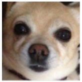
$y_1 = 1$ $\quad y_2 = 1$ $\quad y_3 = -1$ $\quad y_4 = 1$ $\quad y_5 = -1$

1. Use the TRAINING SET $\Rightarrow$ run learning algorithm $\Rightarrow$ get

$$f : X \to Y$$

Training Set and Data Set

Let us take $Y = \{-1, 1\}$, with $y = 1$ means **YES**, and $y = -1$ means **NO**.

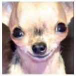
$\boldsymbol{x}_1 =$  $\boldsymbol{x}_2 =$  $\boldsymbol{x}_3 =$  $\boldsymbol{x}_4 =$  $\boldsymbol{x}_5 =$ 

$y_1 = 1$ $\quad y_2 = 1$ $\quad y_3 = -1$ $\quad y_4 = 1$ $\quad y_5 = -1$

1. Use the TRAINING SET $\Rightarrow$ run learning algorithm $\Rightarrow$ get

$$f : X \to Y$$

2. Use a TEST SET to validate f:

$f\Big(\boldsymbol{x}_6 =$  $\Big) = -1$ $\qquad f\Big(\boldsymbol{x}_7 =$  $\Big) = 1$

Open Question: More Data implies better Predictions?

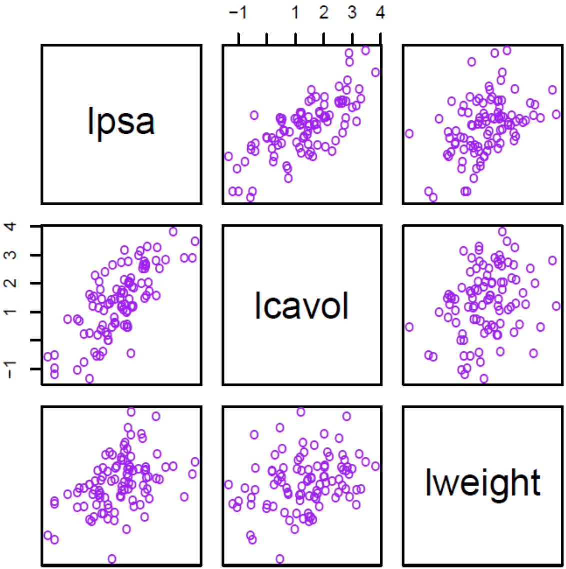
Example 2: Prostate Cancer

Goal: To predict the log of *Prostate Specific Antigen (PSA)* (`lpsa`) from a number of measurement including:

- log-cancer-volume (`lcavol`)
- log prostate weight (`lweight`)
- age
- log of benign prostatic hyperplasia amount (`lbph`)
- seminal vesicle invasion (`svi`)
- log of capsular penetratio (`lcp`)
- Gleason score (`gleason`)
- percentage of Gleason score 4 or 5 (`pgg45`)

Example 2: Prostate Cancer

Example 2: Prostate Cancer



Example 2: Regression

Goal: To predict the log of *Prostate Specific Antigen (PSA)* (`lpsa`) from a number of measurement including:

- log-cancer-volume (`lcavol`)
- log prostate weight (`lweight`)
- age
- log of benign prostatic hyperplasia amount (`lbph`)
- seminal vesicle invasion (`svi`)
- log of capsular penetratio (`lcp`)
- Gleason score (`gleason`)
- percentage of Gleason score 4 or 5 (`pgg45`)

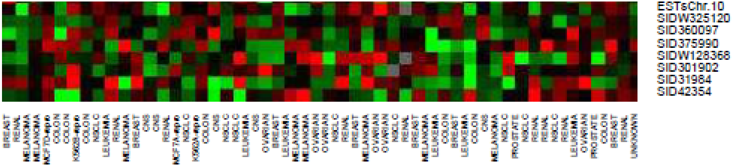
Note: In this case, the outcome of the prediction is a **quantitative measure**. This is a REGRESSION PROBLEM.

Example 3: DNA Expression Microarrays

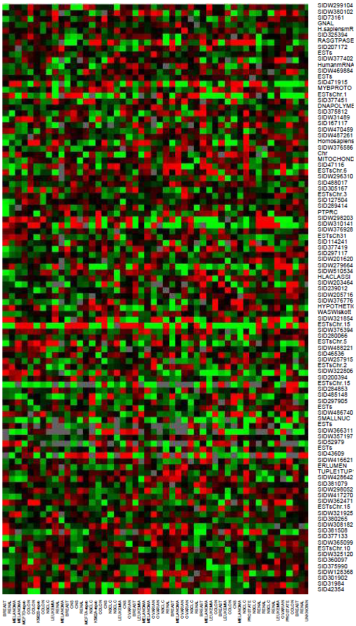
Data: DNA stands for deoxyribonucleic acid, and is the basic material that makes up human chromosomes. DNA microarrays measures the expression of a gene in a cell by measuring the amount of mRNA (messanger ribonucleic acid) present for that gene.

A gene expression dataset collects together the expression values from a series of DNA microarray experiments, with each column representing an experiments. There are several thousands rows representing individual genes, and tens of columns representing samples.

Example 3: DNA Expression Microarrays



Each color represents the expression level of each gene in the target, relative to the reference sample. Positive values (red) indicate higher expressions in the target versus the reference, and vice versa for negative values (green).



- 100 randoms rows (out of 6830) representing genes
- 64 columns representing samples
- We can think of each column as a vector of 6830 real values

Example 3: Clustering

Goal: The challenge here is to understand how the genes and samples are organized. Typical questions include:

1. Which samples are most similar to each other, in terms of their expression profiles across genes?
2. Which genes are most similar to each other, in terms of their expressions profiles across samples?
3. Do certain genes show very high (or low) expression for certain cancer samples?

Example 3: Clustering

Goal: The challenge here is to understand how the genes and samples are organized. Typical questions include:

1. Which samples are most similar to each other, in terms of their expression profiles across genes?
2. Which genes are most similar to each other, in terms of their expressions profiles across samples?
3. Do certain genes show very high (or low) expression for certain cancer samples?

Note: This is an example of **Unsupervised Learning**: we can think of the samples as points in 6830-dimensional space, which we want to **CLUSTER** together in some way.

Informal definition

"Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed"

Nature: Demo

VIDEO CLIP

Formal definition

Definition 1 (Mitchell, 1997)

A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Formal definition

Definition 1 (Mitchell, 1997)

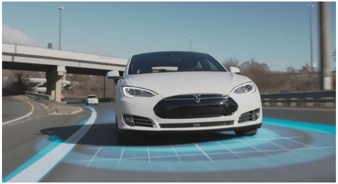
A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Example 2

A handwriting recognition learning problem:

- **Task** T: recognizing and classifying handwritten words within images
- **Performance measure** P: percent of words correctly classified
- **Training experience** E: a dataset of handwritten words with given classifications

Self-driving cars

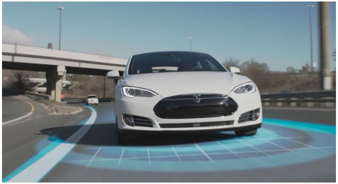


Example 3

A robot driving learning problem:

- **Task** T: driving on public streets using vision sensors
- **Performance measure** P: average distance traveled before an error (as judged by human operator)
- **Training experience** E: a sequence of images and steering commands recorded while observing a human driver

Self-driving cars



Example 3

A robot driving learning problem:

- **Task** T: driving on public streets using vision sensors
- **Performance measure** P: average distance traveled before an error (as judged by human operator)
- **Training experience** E: a sequence of images and steering commands recorded while observing a human driver

Supervised Learning vs. Unsupervised Learning

Definition 4 (Supervised Learning)

Supervised Learning is the task of learning (inferring) a function f that maps input vectors to their corresponding target vectors, by using a dataset containing a given set of pairs of $(input, output)$ samples. Examples:

- REGRESSION: the output vectors take one or more continuous values.
- CLASSIFICATION: the output vectors take one value of a finite number of discrete categories. Special case: binary classification.

Supervised Learning vs. Unsupervised Learning

Definition 5 (Unsupervised Learning)

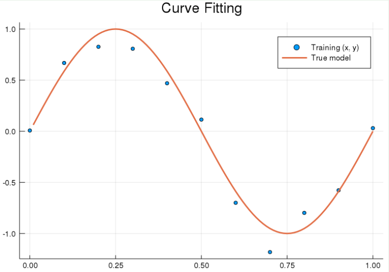
Unsupervised Learning is the task of learning (inferring) a function f that maps input vectors to their corresponding target vectors, but without any a priori knowledge about the correct mapping. Examples:

- CLUSTERING: The goal of clustering is to group or partition the input vectors (if possible) into k groups or clusters, with the vectors in each group close to each other. In this case, the input vectors represents usually features of objects.
- DENSITY ESTIMATION: The goal is to project the data from a high dimensional space down to two or three dimensions, usually for the purpose of *visualization*.

Regression Example: Curve Fitting

Example 6

We are given a training set containing m observations, written $\boldsymbol{x} = (x_1, \ldots, x_m)$, together with corresponding observations of the output values, denoted $\boldsymbol{y} = (y_i, \ldots, y_m)$. Consider the training set of m data points randomly sampled in the range [0..1], and the corresponding target values obtained as



$$y_i = \sin(2\pi x_i) + \mathcal{N}(0, 0.02)$$

Regression Example: Curve Fitting

Learning Model

We want to fit the data using a polynomial function of the form:

$$f(\boldsymbol{x};\boldsymbol{w}) = w_0 + w_1 x^1 + w_2 x^2 + \cdots + w_p x^p = \sum_{j=0}^{p} w_j x^j$$

where p is the order of the polynomial. The polynomial coefficients w_i are collectively denoted by the vector $\boldsymbol{w}$.

NOTE: While the polynomial function $f(\boldsymbol{x};\boldsymbol{w})$ is a nonlinear function of x, it is a **linear function of the coefficients** w.

Regression Example: Curve Fitting

Loss Function

The value of the coefficients will be determined (learned) by fitting the polynomial to the training data.

This can be done by minimizing a *loss function* (or *error function*) that measures the misfit between the function $f(\boldsymbol{x}, \boldsymbol{w})$, for a given value of $\boldsymbol{w}$, and the training set data points. A common loss function is the following:

$$L(\boldsymbol{y}, f(\boldsymbol{x}; \boldsymbol{w})) = \frac{1}{2} \sum_{j=1}^{m} (y_j - f(x_i; \boldsymbol{w}))^2$$

$$L(\boldsymbol{w}) = \frac{1}{2} \sum_{j=1}^{m} (y_j - \hat{y}_j)^2$$

NOTE: This is the standard sum-of-square error function.

Regression Example: Quadratic Loss

As model function $f : X \to Y$ we can start with the following model:

$$y_i = f(x_i) = w_0 + w_1 x_i + \eta_i, \qquad \forall i = 1, \dots, m$$

If m were equal to 2 and $\eta_i = 0$, then we can determine w_0 and w_1.

Regression Example: Quadratic Loss

As model function $f : X \to Y$ we can start with the following model:

$$y_i = f(x_i) = w_0 + w_1 x_i + \eta_i, \qquad \forall i = 1, \ldots, m$$

If m were equal to 2 and $\eta_i = 0$, then we can determine w_0 and w_1.

Since $m >> 2$ we can only consider the errors in our model

$$\eta_i = y_i - f(x_i) = y_i - w_0 - w_2 x_i, \qquad \forall i = 1, \ldots, m$$

Regression Example: Quadratic Loss

As model function $f : X \to Y$ we can start with the following model:

$$y_i = f(x_i) = w_0 + w_1 x_i + \eta_i, \qquad \forall i = 1, \ldots, m$$

If m were equal to 2 and $\eta_i = 0$, then we can determine w_0 and w_1.

Since $m >> 2$ we can only consider the errors in our model

$$\eta_i = y_i - f(x_i) = y_i - w_0 - w_2 x_i, \qquad \forall i = 1, \ldots, m$$

and try to **MINIMIZE** the them.

Regression Example: Quadratic Loss

In matrix notation we can write

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix},$$

Regression Example: Quadratic Loss

In matrix notation we can write

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix}, \qquad w = (w_0, w_1)^T$$

and then

$$A\,w =$$

Regression Example: Quadratic Loss

In matrix notation we can write

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix}, \qquad w = (w_0, w_1)^T$$

and then

$$A\,w = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix} \cdot \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} =$$

Regression Example: Quadratic Loss

In matrix notation we can write

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix}, \qquad w = (w_0, w_1)^T$$

and then

$$A\,w = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix} \cdot \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} w_0 + w_1 x_1 \\ w_0 + w_2 x_2 \\ \dots\dots \\ w_0 + w_1 x_m \end{bmatrix} =$$

Regression Example: Quadratic Loss

In matrix notation we can write

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix}, \qquad w = (w_0, w_1)^T$$

and then

$$A\,w = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix} \cdot \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} w_0 + w_1 x_1 \\ w_0 + w_2 x_2 \\ \dots\dots \\ w_0 + w_1 x_m \end{bmatrix} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \dots \\ f(x_m) \end{bmatrix} =$$

Regression Example: Quadratic Loss

In matrix notation we can write

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix}, \qquad w = (w_0, w_1)^T$$

and then

$$A\,w = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix} \cdot \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} w_0 + w_1 x_1 \\ w_0 + w_2 x_2 \\ \dots\dots \\ w_0 + w_1 x_m \end{bmatrix} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \dots \\ f(x_m) \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ y_m \end{bmatrix} = y$$

And finally

Regression Example: Quadratic Loss

In matrix notation we can write

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix}, \qquad w = (w_0, w_1)^T$$

and then

$$A\,w = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \dots \\ 1 & x_m \end{bmatrix} \cdot \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} w_0 + w_1 x_1 \\ w_0 + w_2 x_2 \\ \dots\dots \\ w_0 + w_1 x_m \end{bmatrix} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \dots \\ f(x_m) \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ y_m \end{bmatrix} = y$$

And finally

$$\eta = y - Ac$$

Regression Example: Quadratic Loss

Since we want to minimize a quadratic loss, we have to minimize the following objective function:

$$L(w)^2 \quad =$$

Regression Example: Quadratic Loss

Since we want to minimize a quadratic loss, we have to minimize the following objective function:

$$L(w)^2 \quad = \quad \min \frac{||\eta||^2}{2} =$$

Regression Example: Quadratic Loss

Since we want to minimize a quadratic loss, we have to minimize the following objective function:

$$
\begin{aligned}
L(w)^2 &= \min \frac{||\eta||^2}{2} = \min \left(\frac{1}{2} \sum_{i=1,\dots,m} \eta_i^2 \right) \qquad (1) \\
&=
\end{aligned}
$$

Regression Example: Quadratic Loss

Since we want to minimize a quadratic loss, we have to minimize the following objective function:

$$
\begin{aligned}
L(w)^2 &= \min \frac{||\eta||^2}{2} = \min \left(\frac{1}{2} \sum_{i=1,\ldots,m} \eta_i^2 \right) && (1) \\
&= \min \left(\frac{1}{2} \sum_{i=1,\ldots,m} (y_i - A^i\, w)^2 \right) && (2) \\
&=
\end{aligned}
$$

Regression Example: Quadratic Loss

Since we want to minimize a quadratic loss, we have to minimize the following objective function:

$$L(w)^2 = \min \frac{||\eta||^2}{2} = \min \left(\frac{1}{2} \sum_{i=1,\ldots,m} \eta_i^2 \right) \tag{1}$$

$$= \min \left(\frac{1}{2} \sum_{i=1,\ldots,m} (y_i - A^i\, w)^2 \right) \tag{2}$$

$$= \frac{1}{2} \min (y - A\, w)^T (y - A\, w) \tag{3}$$

Regression Example: Quadratic Loss

Since we want to minimize a quadratic loss, we have to minimize the following objective function:

$$
\begin{aligned}
L(w)^2 &= \min \frac{||\eta||^2}{2} = \min \left(\frac{1}{2} \sum_{i=1,\ldots,m} \eta_i^2 \right) && (1) \\
&= \min \left(\frac{1}{2} \sum_{i=1,\ldots,m} (y_i - A^i\, w)^2 \right) && (2) \\
&= \frac{1}{2} \min (y - A\, w)^T (y - A\, w) && (3)
\end{aligned}
$$

Question: Which are the variables and which are the unknown?

Regression Example: Quadratic Loss

Hence, we have to find

$$w^* = \arg\min (y - Aw)^T (y - Aw)$$

where $w = (w_0, w_1)^T$ are the unknown.

Regression Example: Quadratic Loss

Hence, we have to find

$$w^* = \arg\min (y - Aw)^T (y - Aw)$$

where $w = (w_0, w_1)^T$ are the unknown.

Since the problem is quadratic, it is enough to find the point

$$\nabla L(\boldsymbol{w}) = 0$$

and to check if the Hessian is positive definite.

Regression Example: Quadratic Loss

Hence, we have to find

$$w^* = \arg\min (y - Aw)^T (y - Aw)$$

where $w = (w_0, w_1)^T$ are the unknown.

Since the problem is quadratic, it is enough to find the point

$$\nabla L(\boldsymbol{w}) = 0$$

and to check if the Hessian is positive definite.

In this case, it is possible to prove that

$$w^* = (A^T A)^{-1} A^T y$$

Julia

```
using Plots, Random, Distributions

Random.seed!(13)

function GenerateSamples(n)
    d = Normal(0, 0.1)

    X = [(1.0/n*i) for i in 0:n]
    Y = [(sin(2*pi*x) + rand(d)) for x in X]

    X0 = [i*1/1000 for i in 0:1000]

    return X0, X, Y
end
```

Julia

```
function PlotData(X0, X, Y, Yhat=[])
    plot(X, Y, seriestype=:scatter, title="Curve Fitting",
 label="Training (x, y)")

    plot!(X0, [sin(2*pi*x) for x in X0], lw=2,
     label="True model")

    if length(Yhat) > 0
        plot!(X0, Yhat, seriestype=:line, lw=2,
      label="Fitted (x, y)")
    end
end
```

Julia

```
function LinearRegression(X, Y)
    n = length(X)
    A = ones(n, 2)
    for i in 1:n
        A[i,2] = X[i]
    end
    println(A)
    w = inv(A'*A)*A'*Y
    return w
end

function PredictLinear(X, w)
    Yhat = [(w[1] + w[2]*x) for x in X]
    return Yhat
end
```

Julia: Exercise

EXERCISE 1: Generalized the Linear Regression to polynomial of order p. What do you observe on the training and test set? How do you measure the errors?

Julia

```
println("--------- START ---------")

X0, X, Y = GenerateSamples(10)

w = LinearRegression(X, Y)
Yhat = PredictLinear(X0, w)
PlotData(X0, X, Y, Yhat)
```

Regression Example: Regularization

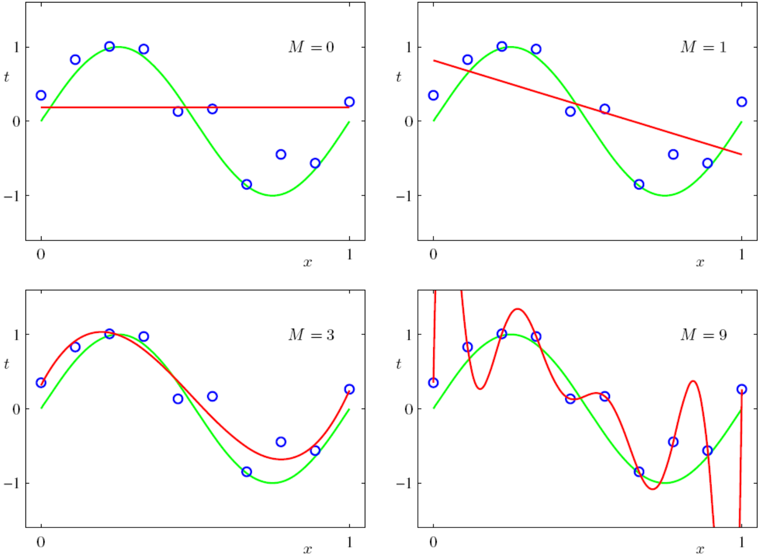
In order to avoid over-fitting, that is, to obtain weight coefficients with very large weights, the most used techniques is called **regularization**, and involves adding a penalty term to the loss function in order to discourage large coefficients.

The simplest penalty is the sum of squares of all coefficients

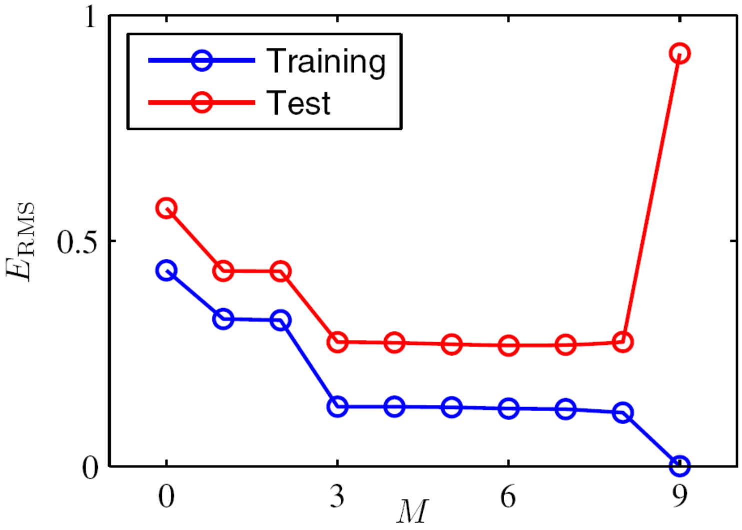
$$\tilde{L}(\boldsymbol{w}) = \frac{1}{2}\sum_{j=1}^{m}(y_j - \hat{y}_j)^2 + \frac{\lambda}{2}||\boldsymbol{w}||^2$$

where $||\boldsymbol{w}||^2 = w_0^2 + w_1^2 + \ldots w_m^2$. and the coefficient λ governs the importance of the regularization term.

Regression Example



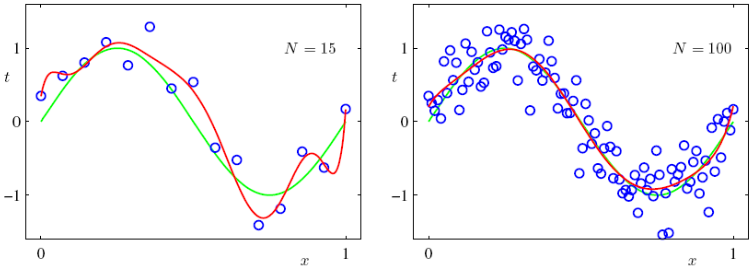
Regression Example



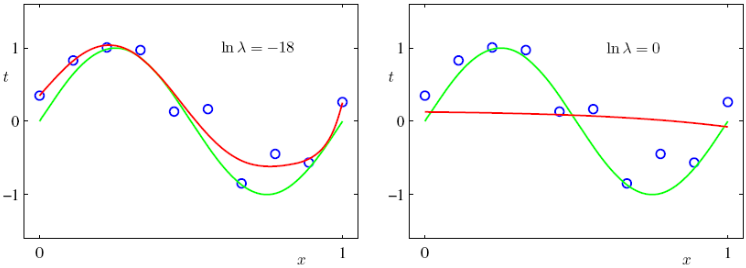
Regression Example

	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Regression Example: Order $p = 9$



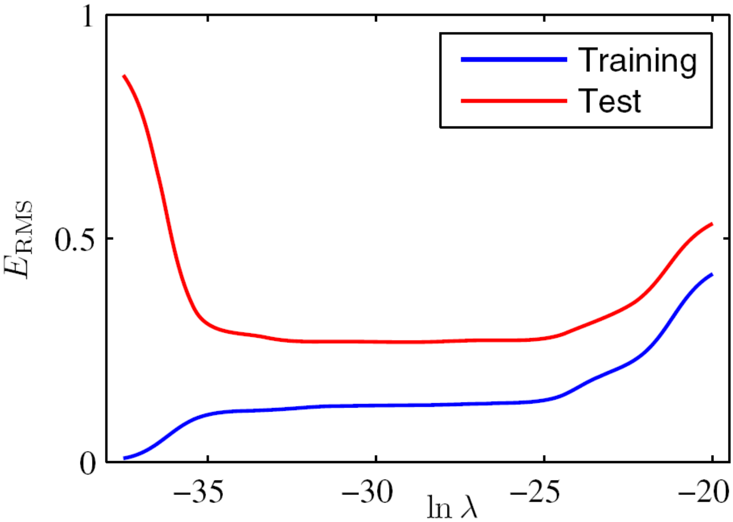
Regression Example: Order $p = 9$



Regression Example: Order $p = 9$

	$\ln\lambda = -\infty$	$\ln\lambda = -18$	$\ln\lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

Regression Example: What we want really minimize?



Pacchetti da installare in Julia

- import Pkg
- Pkg.add("Plots")
- Pkg.add("Distributions")