LABORATOIRE HUBERT CURIEN

UMR • CNRS • 5516 • SAINT-ETIENNE

UNIVERSITÉ DE LYON

From Statistics to Data Mining

Master 1
COlour in Science and Industry (COSI)
Cyber-Physical Social System (CPS2)
Saint-Étienne, France

Fabrice MUHLENBACH
https://perso.univ-st-etienne.fr/muhlfabr/
e-mail: fabrice.muhlenbach@univ-st-etienne.fr

Regression Analysis

- Introduction

➢ **Definition**

- family of methods used in statistical modeling
- regression analysis = statistical processes for estimating the relationships between a **dependent** variable (often called the "outcome variable") and one or more **independent variables** (often called "predictors," "covariates," or "features")
- **example**: linear regression → finding the line (or a more complex linear combination) that most closely fits a data set

➢ **Interests:**

- prediction and forecasting (→ machine learning)
- infer causal relationships between the independent and dependent variables

Regression Analysis

- Linear Regression

➢ **Definition and notation**

○ a linear regression model (or “linear model”) is a regression model which seeks to establish a linear relationship between a variable, called “explained”, “dependent” or “outcome” and noted Y, and a (→ simple linear regression simple) or several (→ multiple linear regression) variables, called “features”, “independent” or “predictors” denoted $\mathbf{X} = \{X_1, X_2, \cdots, X_p\}$

➢ **Objective**

○ establish a link between a dependent variable Y and one (or more) independent variable(s) $\mathbf{X}$ in order to then be able to make predictions on Y when $\mathbf{X}$ is (are) measured

Regression Analysis

- Linear Regression: Introduction

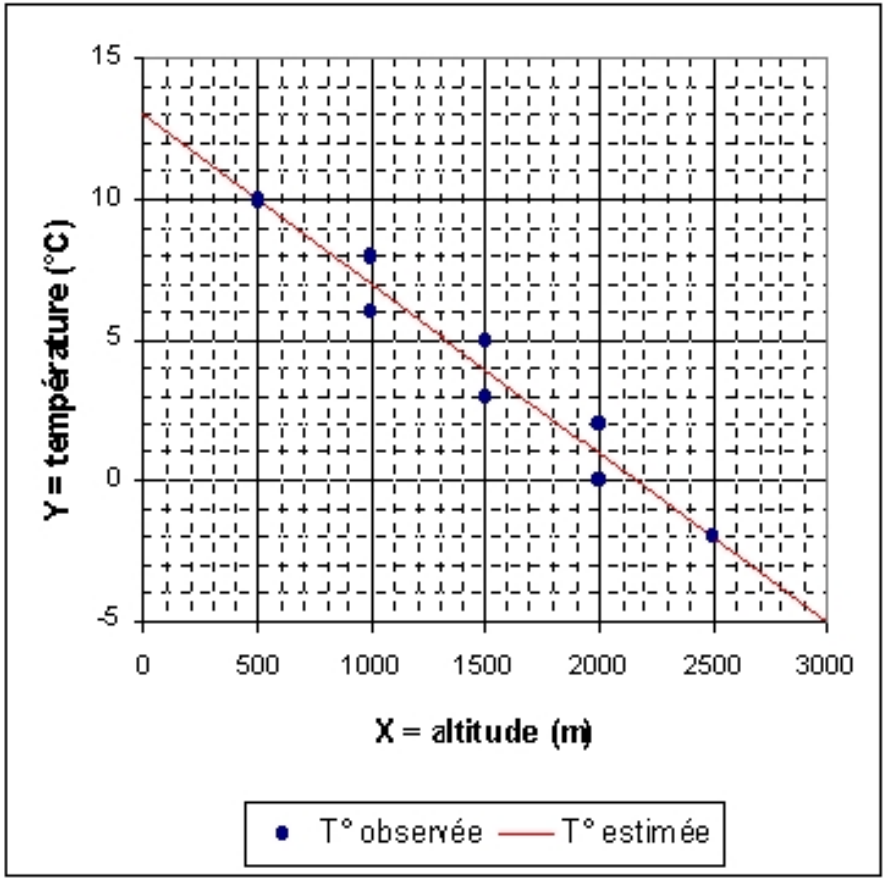
- **example**: study of the relationship between altitude (X) and temperature (Y) within a region of a sufficiently small size so that the factors of macroscopic variations in temperature can be neglected
- these fictitious data could correspond to the situation of an alpine valley of north-south direction for which one proceeded to the reading of temperatures at noon in eight stations located at different altitudes and located on each side of the valley

i	Xi	Yi
1	2000	0
2	1500	3
3	1000	6
4	500	10
5	1000	8
6	1500	5
7	2000	2
8	2500	-2

Regression Analysis

- Linear Regression: Introduction

- **example**: study of the relationship between altitude (X) and temperature (Y)
- → temperature model (in red line) as a function of altitude:
- the model (red line) of the estimated temperature follows the direction of the scatterplot of the observed temperature (blue dots)



Regression Analysis

- Linear Regression

➢ **Notations**

- m: # of training examples
- $x \in \mathbb{R}^n$: input feature vector of n variables
- y: output variable / target
- (x, y): training example
- (x_i, y_i): i^{th} training example
- h: hypothesis that maps input x to ouput y
- $h(x)$ is of the linear form: $h(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n$
  we denote $h(x) = h_\theta(x) = \sum_{i=1}^{n} \theta_i x_i = \theta^{\mathrm{T}} x$
  where $\theta = (\theta_1, \cdots, \theta_n)$ are called parameters of the linear regression (for conciseness, $h_\theta(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^{\mathrm{T}} x$ with the first θ_0 and $x_0 = 1$)

Regression Analysis

- Linear Regression

➢ **Goal**

- how do we choose the parameters θ so that our hypothesis h will make accurate predictions about all the values of y?
- in our example, how to we choose the parameters θ in order to have a function giving the temperature from the altitude?

➢ **Objective function**

$$\min_\theta \frac{1}{2}\sum_{i=1}^{n}(h_\theta(x_i)-y_i)^2$$

- let us define $J_\theta = \frac{1}{2}\sum_{i=1}^{n}(h_\theta(x_i)-y_i)^2$
- therefore, the objective function becomes: $\min_\theta J_\theta$

Regression Analysis

- Learning algorithms for minimizing J_θ

➢ How to minimize J_θ?

○ finding the parameters θ $\Leftrightarrow$ minimizing J_θ

○ several solutions exist for minimizing J_θ:

❑ batch gradient descent
❑ stochastic gradient descent
❑ closed-form solution

Regression Analysis

- Learning algorithms for minimizing J_θ

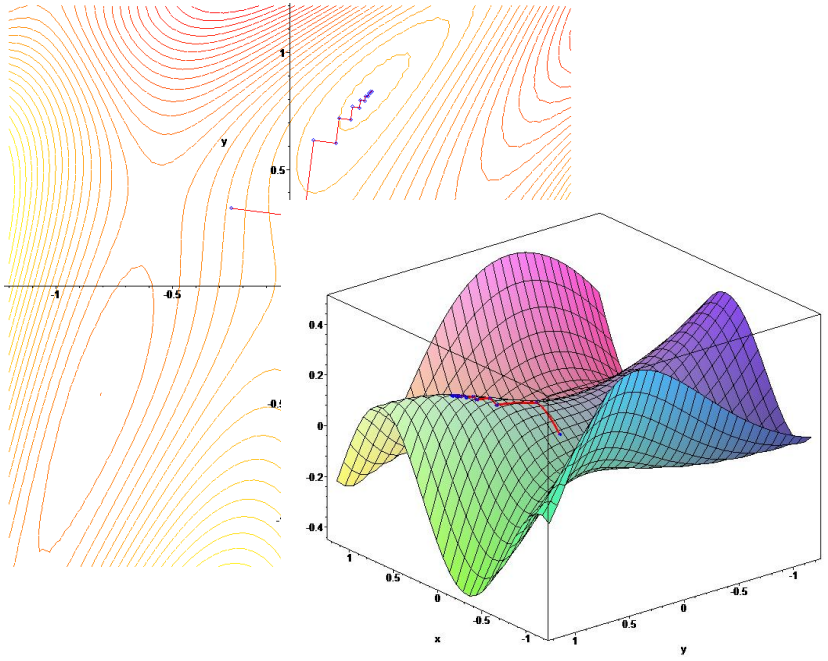
➢ Basic idea of the gradient descent

- start with some value of θ (e.g., $\vec{\theta} = 0$ that denotes a vector of all zeros or some randomly chosen vector)
- update θ's values so that to reduce J_θ (computing partial derivatives)
- repeat the process till convergence to the minimum of J_θ w.r.t. θ

Regression Analysis

- Gradient descent algorithm

- the gradient algorithm is also known as the “steepest descent algorithm” → the gradient is the slope of the linearized function at the current point and is therefore, locally, its steepest slope
- in its simplest version, the algorithm can only find or approach a stationary point of an optimization problem without constraint
- such points are global minima, if the function is convex



Regression Analysis

- Gradient descent algorithm

➢ **Update rule:**

- **gradient descent** is based on the observation that if the function $J(\theta)$ is differentiable in a neighborhood of a point x, then $J(\theta)$ decreases fastest if one goes from x in the direction of the negative gradient of $J(\theta)$
- for every step, gradient descent updates each parameter i as follows: $\theta_i \leftarrow \theta_i - \alpha \frac{\partial}{\partial \theta_i} J(\theta)$ where:

❑ $\frac{\partial}{\partial \theta_i} J(\theta)$ gives us the direction of the **deepest descent**

❑ α is the **learning rate** which controls how large a step you take in the direction of the steepest descent

Regression Analysis

- Batch gradient descent algorithm

- with m training examples, we get:

initialization of $\vec{\theta}$

repeat until convergence of $J(\theta)$:

$\forall i \in [0; n]$

$$\theta_i \leftarrow \theta_i - \alpha \frac{1}{m} \sum_{j=1}^{m} \left(h_\theta\left(x_j\right) - y_i\right)(x_i)$$

- note that we use “batch” because at each gradient descent, we are going to look at the entire training set and performing a sum over the m examples
- the convergence criterion is reached when the quantity $J(\theta)$ is lower than a determined threshold (a maximum value)
- convergence needs several runs (epochs) of the learning set

Regression Analysis

- Batch gradient descent algorithm

➢ **Limits of the general gradient descent algorithm**

- the gradient descent algorithm used by default (batch) works by updating from all of the m examples
- very often, the elementary functions constituting the sum take a simple form which allows the efficient calculation of the sum function and its gradient
- **problem 1**: in other cases, evaluating the entire gradient can become very expensive, when the calculation of the gradients of each piece is not easy
- **problem 2**: if the learning set is very large (*big data*) and no simple formula is available for the gradients, the calculations are very expensive too

Regression Analysis

- Stochastic gradient descent algorithm

➢ **Remark**

○ If m is huge –say millions of examples– then if you are running batch gradient descent, you have to perform at each step a sum over one million of examples

○ → therefore, we need an alternative algorithm

➢ **Stochastic (or incremental) gradient descent:**

initialization de $\vec{\theta}$

repeat until convergence of $J(\theta)$:

  for j in 1 to m:

    $\forall i \in [0; n]$

    $\theta_i \leftarrow \theta_i - \alpha \frac{1}{m} \left(h_\theta\left(x_j\right) - y_i\right)(x_i)$

Regression Analysis

- Stochastic gradient descent algorithm

➢ **Explanation**

- the repetition is carried out until an approximate minimum (quite precisely) is obtained we randomly mix the samples of the learning set to then update

- **advantage**: the process is much faster for large datasets than the batch gradient algorithm

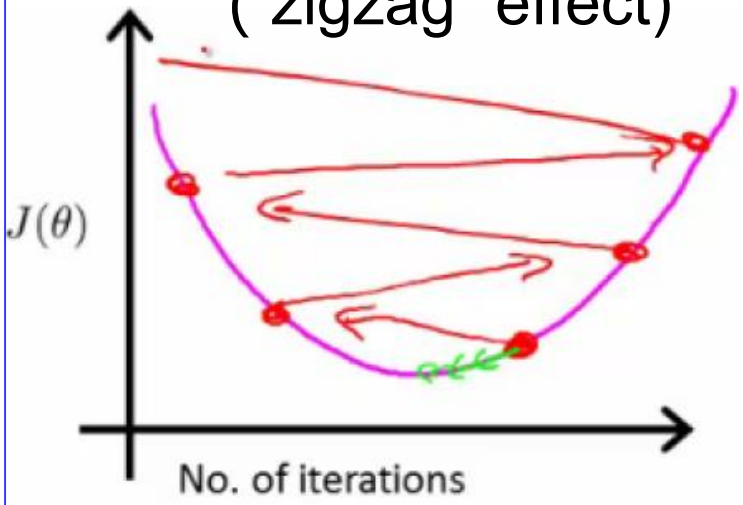
- **disadvantage**: the learning process does not converge to a global minimum exactly but it tends to oscillate around regions close to the global minimum

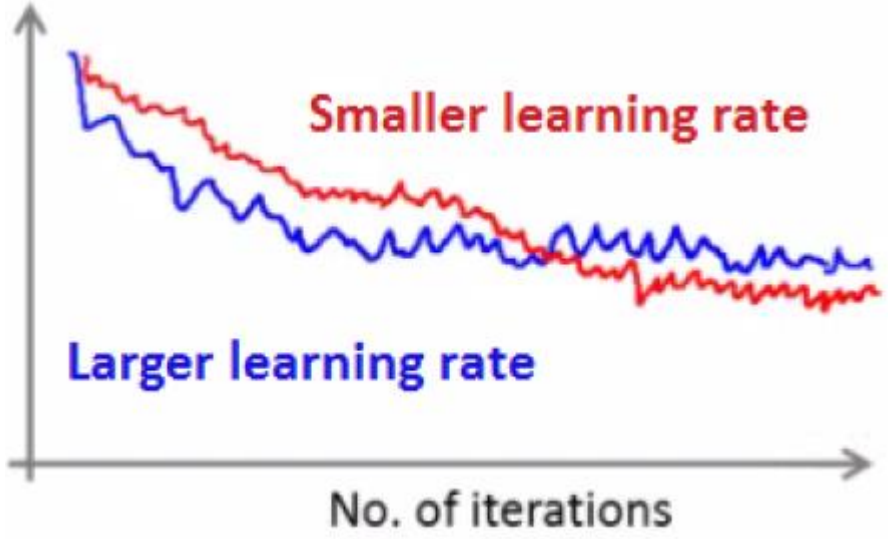
Regression Analysis

- Stochastic gradient descent algorithm

➢ **Learning rate α** → large versus small learning rate:

- for some specific examples, $J(\theta)$ may increase
- this may occur in such following situations where is large ("zigzag" effect)





- to prevent the parameters θ's from oscillating around the global minimum we can take a smaller learning rate

Regression Analysis

- Closed-form Solution

➢ **Closed-form solution**

- there is another way to perform the minimization of $J(\theta)$ that allows you to solve for the parameters θ's in closed-form, without needing to run an iterative algorithm

➢ **Definition**

- a mathematical problem is said to have a **closed-form solution** if, and only if, at least one solution of that problem can be **expressed analytically** in terms of a finite number of certain "well-known" functions (e.g., n^{th} root, exponent, logarithm, trigonometric functions, and inverse hyperbolic functions)

Regression Analysis

- Closed-form Solution

➢ **Example 1**

○ the quadratic equation $ax^2 + bx + c = 0$ is closed-form since its solutions can be expressed in terms of elementary functions:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

➢ **Example 2**

○ if A is a 2×2 matrix, then the solutions (that is the eigen-values of A) of the characteristic equation $det(A - \lambda I) = 0$ are:

$$\lambda = \frac{\mathrm{Tr}(A) \pm \sqrt{\mathrm{Tr}(A)^2 - 4\,\det(A)}}{2}$$

Regression Analysis

- Closed-form Solution

➢ **Explanation of example 2**

- let $\nabla_\theta J = \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \vdots \\ \frac{\partial J}{\partial \theta_n} \end{bmatrix} \in \mathbb{R}^{n+1}$ be the gradient of J w.r.t. θ
- we can rewrite the batch gradient descent algorithm as follows: $\theta \leftarrow \theta - \alpha \nabla_\theta J$
- where both θ and $\nabla_\theta J$ are $(n+1)$-dimensional feature vectors
- by properties on the Trace function, it is possible to reduce $\nabla_\theta J(\theta) = \vec{0}$ by solving the following equation: $X^{\mathrm{T}} X\theta - X^{\mathrm{T}} y = 0$, hence $X^{\mathrm{T}} X\theta = X^{\mathrm{T}} y$ (normal equation)
- we get $\theta = \left(X^{\mathrm{T}} X\right)^{-1} X^{\mathrm{T}} y$ (closed form solution)

Regression Analysis

- Closed-form Solution

➢ **Special case: $x \in \mathbb{R}^n$ with $n = 1$**

○ when the training set is composed of pairs (x, y) s.t. $x \in \mathbb{R}$, the closed-form solution becomes:

$$\theta_1 = \frac{\text{cov}(X, Y)}{V(X)}$$

$$\theta_0 = \bar{Y} - \theta_1 \bar{X}$$

○ therefore, the predicted output for a given new x' is

$$h(x') = \theta^{\mathrm{T}} x' = \frac{\text{cov}(X, Y)}{V(X)} x' + \bar{Y} - \theta_1 \bar{X}$$

Regression Analysis

- Closed-form Solution

➢ **What if $(X^T X)^{-1}$ is non-invertible?**

- redundant features (linearly dependent)
  → solution: perform a PCA

- too many features
  → solution: delete some features before (feature selection algorithm), or during the process (e.g., use regularization)

Regression Analysis

- Conclusion

➢ **Comparison: gradient descent *vs*. normal equation**

○ **gradient algorithm:**

☹ need to choose a parameter α (the learning step)

☹ need a lot of iterations

☺ works well even when n is large ($n \sim 10^6$)

☺ allows an online acquisition of training data

○ **normal equation:**

☺ no need to choose α

☺ no need to iterate

☹ no need to compute $\left(X^{\mathrm{T}}X\right)^{-1}$

☹ slow if n is large

Regression Analysis

- Improved versions of regression analysis methods

- **locally weighted regression** (LOESS: *LOcally weighted Scatterplot Smoother*) → strongly related nonparametric regression method that combines several multiple regression models within a meta-model that relies on the k-NN
- **learning of nonlinear models** → polynomial regression: search by (multiple linear) regression to link the variables by a polynomial of degree n:

$$y = w_0 + w_1 x + w_2 x^2 + \cdots + w_n x^n$$

- **linear regression with regularization**
  → prevent over-fitting
  → LASSO method (*Least Absolute Shrinkage and Selection Operator*): contraction of the coefficients of the regression