Learning from data: Linear Regression

Christian Forssén[1]

Morten Hjorth-Jensen[2,3]

[1]Department of Physics, Chalmers University of Technology, Sweden
[2]Department of Physics, University of Oslo
[3]Department of Physics and Astronomy and National Superconducting Cyclotron Laboratory, Michigan State University

Sep 2, 2019

0.1 Why Linear Regression (aka Ordinary Least Squares and family)

Fitting a continuous function with linear parameterization in terms of the parameters $\boldsymbol{\theta}$.

- Method of choice for fitting a continuous function!
- Gives an excellent introduction to central Machine Learning features with **understandable pedagogical** links to other methods like **Neural Networks**, **Support Vector Machines** etc
- Analytical expression for the fitting parameters $\boldsymbol{\theta}$
- Analytical expressions for statistical propertiers like mean values, variances, confidence intervals and more
- Analytical relation with probabilistic interpretations
- Easy to introduce basic concepts like bias-variance tradeoff, cross-validation, resampling and regularization techniques and many other ML topics
- Easy to code! And links well with classification problems and logistic regression and neural networks
- Allows for **easy** hands-on understanding of gradient descent methods
- and many more features

Regression analysis, overarching aims.

Regression modeling deals with the description of the sampling distribution of a given random variable y and how it varies as function of another variable or a set of such variables $\boldsymbol{x} = [x_0, x_1, \dots, x_{n-1}]^T$. The first variable is called the **dependent**, the **outcome** or the **response** variable while the set of variables $\boldsymbol{x}$ is called the independent variable, or the predictor variable or the explanatory variable.

A regression model M aims at finding a likelihood function $p(\boldsymbol{y}|\boldsymbol{x}, M, \mathcal{D})$, that is the conditional distribution for $\boldsymbol{y}$ given the independent variable $\boldsymbol{x}$ and a model M that has been trained on a data set $\mathcal{D}$ with

- n cases $i = 0, 1, 2, \dots, n-1$
- Response (target, dependent or outcome) variable y_i with $i = 0, 1, 2, \dots, n-1$
- p so-called explanatory (independent or predictor) variables $\boldsymbol{x}_i = [x_{i0}, x_{i1}, \dots, x_{ip-1}]$ with $i = 0, 1, 2, \dots, n-1$ and explanatory variables running from 0 to $p-1$. See below for more explicit examples.

The goal of the regression analysis is to extract/exploit relationship between $\boldsymbol{y}$ and $\boldsymbol{x}$ in or to infer causal dependencies, approximations to the likelihood functions, functional relationships and to make predictions, making fits and many other things.

Consider an experiment in which p characteristics of n samples are measured. The data from this experiment, for various explanatory variables p are normally represented by a matrix $\mathbf{X}$.

The matrix $\mathbf{X}$ is called the *design matrix*. Additional information of the samples is available in the form of $\boldsymbol{y}$ (also as above). The variable $\boldsymbol{y}$ is generally referred to as the *response variable*. The aim of regression analysis is to explain $\boldsymbol{y}$ in terms of $\boldsymbol{X}$ through a functional relationship like $y_i = f(\mathbf{X}_{i,*})$. When no prior knowledge on the form of $f(\cdot)$ is available, it is common to assume a linear relationship between $\boldsymbol{X}$ and $\boldsymbol{y}$. This assumption gives rise to the *linear regression model* where $\boldsymbol{\theta} = [\theta_0, \dots, \theta_{p-1}]^T$ are the *regression parameters*.

Linear regression gives us a set of analytical equations for the parameters θ_j.

Example: Liquid-drop model for nuclear binding energies.

In order to understand the relation among the predictors p, the set of data n and the target (outcome, output etc) $\boldsymbol{y}$, consider the model we discussed for describing nuclear binding energies.

There we assumed that we could parametrize the data using a polynomial approximation based on the liquid drop model. Assuming

$$BE(A) = a_0 + a_1 A + a_2 A^{2/3} + a_3 A^{-1/3} + a_4 A^{-1},$$

we have five predictors, that is the intercept, the A dependent term, the $A^{2/3}$ term and the $A^{-1/3}$ and A^{-1} terms. This gives $p = 0, 1, 2, 3, 4$. Furthermore we have n entries for each predictor. It means that our design matrix is a $p \times n$ matrix $\boldsymbol{X}$.

0.2 General linear models

Polynomial basis functions.

Before we proceed let us study a case from linear algebra where we aim at fitting a set of data $\boldsymbol{y} = [y_0, y_1, \dots, y_{n-1}]$. We could think of these data as a result of an experiment or a complicated numerical experiment. These data are functions of a series of variables $\boldsymbol{x} = [x_0, x_1, \dots, x_{n-1}]$, that is $y_i = y(x_i)$ with $i = 0, 1, 2, \dots, n-1$. The variables x_i could represent physical quantities like time, temperature, position etc. We assume that $y(x)$ is a smooth function.

Since obtaining these data points may not be trivial, we want to use these data to fit a function which can allow us to make predictions for values of y which are not in the present set. The perhaps simplest approach is to assume we can parametrize our function in terms of a polynomial of degree $n-1$ with n points, that is

$$y = y(x) \rightarrow y(x_i) = \tilde{y}_i + \epsilon_i = \sum_{j=0}^{n-1} \theta_j x_i^j + \epsilon_i,$$

where ϵ_i is the error in our approximation.

For every set of values y_i, x_i we have thus the corresponding set of equations

$$
\begin{aligned}
y_0 &= \theta_0 + \theta_1 x_0^1 + \theta_2 x_0^2 + \dots + \theta_{n-1} x_0^{n-1} + \epsilon_0 \\
y_1 &= \theta_0 + \theta_1 x_1^1 + \theta_2 x_1^2 + \dots + \theta_{n-1} x_1^{n-1} + \epsilon_1 \\
y_2 &= \theta_0 + \theta_1 x_2^1 + \theta_2 x_2^2 + \dots + \theta_{n-1} x_2^{n-1} + \epsilon_2 \\
\dots & \dots \\
y_{n-1} &= \theta_0 + \theta_1 x_{n-1}^1 + \theta_2 x_{n-1}^2 + \dots + \theta_{n-1} x_{n-1}^{n-1} + \epsilon_{n-1}.
\end{aligned}
$$

Defining the vectors

$$\boldsymbol{y} = [y_0, y_1, y_2, \dots, y_{n-1}]^T,$$

and

$$\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \dots, \theta_{n-1}]^T,$$

and

$$\boldsymbol{\epsilon} = [\epsilon_0, \epsilon_1, \epsilon_2, \dots, \epsilon_{n-1}]^T,$$

and the design matrix

$$\boldsymbol{X} = \begin{bmatrix} 1 & x_0^1 & x_0^2 & \dots & \dots & x_0^{n-1} \\ 1 & x_1^1 & x_1^2 & \dots & \dots & x_1^{n-1} \\ 1 & x_2^1 & x_2^2 & \dots & \dots & x_2^{n-1} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & x_{n-1}^1 & x_{n-1}^2 & \dots & \dots & x_{n-1}^{n-1} \end{bmatrix}$$

we can rewrite our equations as

$$\boldsymbol{y} = \boldsymbol{X\theta} + \boldsymbol{\epsilon}.$$

The above design matrix is called a Vandermonde matrix.

General basis functions.

We are obviously not limited to the above polynomial expansions. We could replace the various powers of x with elements of Fourier series or instead of x_i^j we could have $\cos{(jx_i)}$ or $\sin{(jx_i)}$, or time series or other orthogonal functions. For every set of values y_i, x_i we can then generalize the equations to

$$
\begin{aligned}
y_0 &= \theta_0 x_{00} + \theta_1 x_{01} + \theta_2 x_{02} + \cdots + \theta_{n-1} x_{0n-1} + \epsilon_0 \\
y_1 &= \theta_0 x_{10} + \theta_1 x_{11} + \theta_2 x_{12} + \cdots + \theta_{n-1} x_{1n-1} + \epsilon_1 \\
y_2 &= \theta_0 x_{20} + \theta_1 x_{21} + \theta_2 x_{22} + \cdots + \theta_{n-1} x_{2n-1} + \epsilon_2 \\
\dots&\dots \\
y_i &= \theta_0 x_{i0} + \theta_1 x_{i1} + \theta_2 x_{i2} + \cdots + \theta_{n-1} x_{in-1} + \epsilon_i \\
\dots&\dots \\
y_{n-1} &= \theta_0 x_{n-1,0} + \theta_1 x_{n-1,2} + \theta_2 x_{n-1,2} + \cdots + \theta_{n-1} x_{n-1,n-1} + \epsilon_{n-1}.
\end{aligned}
$$

Note that we have $p = n$ here. The matrix is symmetric. This is generally not the case!

We redefine in turn the matrix $\boldsymbol{X}$ as

$$
\boldsymbol{X} = \begin{bmatrix}
x_{00} & x_{01} & x_{02} & \dots & \dots & x_{0,n-1} \\
x_{10} & x_{11} & x_{12} & \dots & \dots & x_{1,n-1} \\
x_{20} & x_{21} & x_{22} & \dots & \dots & x_{2,n-1} \\
\dots & \dots & \dots & \dots & \dots & \dots \\
x_{n-1,0} & x_{n-1,1} & x_{n-1,2} & \dots & \dots & x_{n-1,n-1}
\end{bmatrix}
$$

and without loss of generality we rewrite again our equations as

$$
\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\theta} + \boldsymbol{\epsilon}.
$$

The left-hand side of this equation is kwown. Our error vector $\boldsymbol{\epsilon}$ and the parameter vector $\boldsymbol{\theta}$ are our unknow quantities. How can we obtain the optimal set of θ_i values?

We have defined the matrix $\boldsymbol{X}$ via the equations

$$\begin{aligned}
y_0 &= \theta_0 x_{00} + \theta_1 x_{01} + \theta_2 x_{02} + \cdots + \theta_{n-1} x_{0n-1} + \epsilon_0 \\
y_1 &= \theta_0 x_{10} + \theta_1 x_{11} + \theta_2 x_{12} + \cdots + \theta_{n-1} x_{1n-1} + \epsilon_1 \\
y_2 &= \theta_0 x_{20} + \theta_1 x_{21} + \theta_2 x_{22} + \cdots + \theta_{n-1} x_{2n-1} + \epsilon_1 \\
&\dots\dots \\
y_i &= \theta_0 x_{i0} + \theta_1 x_{i1} + \theta_2 x_{i2} + \cdots + \theta_{n-1} x_{in-1} + \epsilon_1 \\
&\dots\dots \\
y_{n-1} &= \theta_0 x_{n-1,0} + \theta_1 x_{n-1,2} + \theta_2 x_{n-1,2} + \cdots + \theta_{n-1} x_{n-1,n-1} + \epsilon_{n-1}.
\end{aligned}$$

As we noted above, we stayed with a system with the design matrix $\boldsymbol{X} \in \mathbb{R}^{n\times n}$, that is we have $p = n$. For reasons to come later (algorithmic arguments) we will hereafter define our matrix as $\boldsymbol{X} \in \mathbb{R}^{n\times p}$, with the predictors refering to the column numbers and the entries n being the row elements.

With the above we use the design matrix to define the approximation $\boldsymbol{\tilde{y}}$ via the unknown quantity $\boldsymbol{\theta}$ as

$$\boldsymbol{\tilde{y}} = \boldsymbol{X}\boldsymbol{\theta},$$

and in order to find the optimal parameters θ_i instead of solving the above linear algebra problem, we define a function which gives a measure of the spread between the values y_i (which represent hopefully the exact values) and the parameterized values $\tilde{y}_i$, namely

$$C(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \tilde{y}_i)^2 = \frac{1}{n}\left\{(\boldsymbol{y}-\boldsymbol{\tilde{y}})^T(\boldsymbol{y}-\boldsymbol{\tilde{y}})\right\},$$

or using the matrix $\boldsymbol{X}$ and in a more compact matrix-vector notation as

$$C(\boldsymbol{\theta}) = \frac{1}{n}\left\{(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\theta})^T(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\theta})\right\}.$$

This function is one possible way to define the so-called cost function.

It is also common to define the function Q as

$$C(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=0}^{n-1}(y_i - \tilde{y}_i)^2,$$

since when taking the first derivative with respect to the unknown parameters θ, the factor of 2 cancels out.

The function

$$C(\boldsymbol{\theta}) = \frac{1}{n}\left\{\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\theta}\right)^T\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\theta}\right)\right\},$$

can be linked to the variance of the quantity y_i if we interpret the latter as the mean value. When linking (see the discussion below) with the maximum likelihood approach below, we will indeed interpret y_i as a mean value

$$y_i = \langle y_i \rangle = \theta_0 x_{i,0} + \theta_1 x_{i,1} + \theta_2 x_{i,2} + \dots + \theta_{n-1} x_{i,n-1} + \epsilon_i,$$

where $\langle y_i \rangle$ is the mean value. Keep in mind also that till now we have treated y_i as the exact value. Normally, the response (dependent or outcome) variable y_i the outcome of a numerical experiment or another type of experiment and is thus only an approximation to the true value. It is then always accompanied by an error estimate, often limited to a statistical error estimate given by the standard deviation discussed earlier. In the discussion here we will treat y_i as our exact value for the response variable.

In order to find the parameters θ_i we will then minimize the spread of $C(\boldsymbol{\theta})$, that is we are going to solve the problem

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^p}\frac{1}{n}\left\{\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\theta}\right)^T\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\theta}\right)\right\}.$$

In practical terms it means we will require

$$\frac{\partial C(\boldsymbol{\theta})}{\partial \theta_j} = \frac{\partial}{\partial \theta_j}\left[\frac{1}{n}\sum_{i=0}^{n-1}\left(y_i - \theta_0 x_{i,0} - \theta_1 x_{i,1} - \theta_2 x_{i,2} - \dots - \theta_{n-1} x_{i,n-1}\right)^2\right] = 0,$$

which results in

$$\frac{\partial C(\boldsymbol{\theta})}{\partial \theta_j} = -\frac{2}{n}\left[\sum_{i=0}^{n-1} x_{ij}\left(y_i - \theta_0 x_{i,0} - \theta_1 x_{i,1} - \theta_2 x_{i,2} - \dots - \theta_{n-1} x_{i,n-1}\right)\right] = 0,$$

or in a matrix-vector form as

$$\frac{\partial C(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0 = \boldsymbol{X}^T\left(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\theta}\right).$$

We can rewrite

$$\frac{\partial C(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0 = \boldsymbol{X}^T \left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta} \right),$$

as

$$\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\theta},$$

and if the matrix $\boldsymbol{X}^T\boldsymbol{X}$ is invertible we have the solution

$$\boldsymbol{\theta} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}.$$

We note also that since our design matrix is defined as $\boldsymbol{X} \in \mathbb{R}^{n\times p}$, the product $\boldsymbol{X}^T\boldsymbol{X} \in \mathbb{R}^{p\times p}$. In the liquid drop model example from the Intro lecture, we had $p = 5$ ($p \ll n$) meaning that we end up with inverting a small 5×5 matrix. This is a rather common situation, in many cases we end up with low-dimensional matrices to invert. The methods discussed here and for many other supervised learning algorithms like classification with logistic regression or support vector machines, exhibit dimensionalities which allow for the usage of direct linear algebra methods such as **LU** decomposition or **Singular Value Decomposition** (SVD) for finding the inverse of the matrix $\boldsymbol{X}^T\boldsymbol{X}$.

Small question: What kind of problems can we expect when inverting the matrix $\boldsymbol{X}^T\boldsymbol{X}$?

0.3 Adding error analysis and training set up

We can easily test our fit by computing various **cost functions** (or **training scores**). Several such cost functions are used in machine learning applications. First we have the **Mean-Squared Error** (MSE)

$$\mathrm{MSE}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\left(y_{\mathrm{data},i} - y_{\mathrm{model},i}(\boldsymbol{\theta})\right)^2,$$

where we have n training data and our model is a function of the parameter vector $\boldsymbol{\theta}$.

Furthermore, we have the **mean absolute error** (MAE) defined as.

$$\mathrm{MAE}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\left|y_{\mathrm{data},i} - y_{\mathrm{model},i}(\boldsymbol{\theta})\right|,$$

And the $R2$ score, also known as *coefficient of determination* is

$$\mathrm{R2}(\boldsymbol{\theta}) = 1 - \frac{\sum_{i=1}^{n}\left(y_{\mathrm{data},i} - y_{\mathrm{model},i}(\boldsymbol{\theta})\right)^2}{\sum_{i=1}^{n}\left(y_{\mathrm{data},i} - \bar{y}_{\mathrm{model}}(\boldsymbol{\theta})\right)^2},$$

where $\bar{y}_{\mathrm{model}}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} y_{\mathrm{model},i}(\boldsymbol{\theta})$ is the mean of the model predictions.

The χ^2 function.

Normally, the response (dependent or outcome) variable y_i is the outcome of a numerical experiment or another type of experiment and is thus only an approximation to the true value. It is then always accompanied by an error estimate, often limited to a statistical error estimate given by the standard deviation discussed earlier. In the discussion here we will treat y_i as our exact value for the response variable.

Introducing the standard deviation σ_i for each measurement y_i (assuming uncorrelated errors), we define the χ^2 function as

$$\chi^2(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=0}^{n-1}\frac{(y_i-\tilde{y}_i)^2}{\sigma_i^2} = \frac{1}{n}\left\{(\boldsymbol{y}-\tilde{\boldsymbol{y}})^T\,\boldsymbol{\Sigma}^{-1}\,(\boldsymbol{y}-\tilde{\boldsymbol{y}})\right\},$$

where the matrix $\boldsymbol{\Sigma}$ is a diagonal $n\times n$ matrix with σ_i^2 as matrix elements.

In order to find the parameters θ_i we will then minimize the spread of $\chi^2(\boldsymbol{\theta})$ by requiring

$$\frac{\partial\chi^2(\boldsymbol{\theta})}{\partial\theta_j} = \frac{\partial}{\partial\theta_j}\left[\frac{1}{n}\sum_{i=0}^{n-1}\left(\frac{y_i-\theta_0x_{i,0}-\theta_1x_{i,1}-\theta_2x_{i,2}-\dots-\theta_{n-1}x_{i,n-1}}{\sigma_i}\right)^2\right]=0,$$

which results in

$$\frac{\partial\chi^2(\boldsymbol{\theta})}{\partial\theta_j} = -\frac{2}{n}\left[\sum_{i=0}^{n-1}\frac{x_{ij}}{\sigma_i}\left(\frac{y_i-\theta_0x_{i,0}-\theta_1x_{i,1}-\theta_2x_{i,2}-\dots-\theta_{n-1}x_{i,n-1}}{\sigma_i}\right)\right]=0,$$

or in a matrix-vector form as

$$\frac{\partial\chi^2(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}} = 0 = \boldsymbol{A}^T\left(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{\theta}\right).$$

where we have defined the matrix $\boldsymbol{A}=\boldsymbol{X}\boldsymbol{\Sigma}^{-1/2}$ with matrix elements $a_{ij}=x_{ij}/\sigma_i$ and the vector $\boldsymbol{b}$ with elements $b_i=y_i/\sigma_i$.

We can rewrite

$$\frac{\partial\chi^2(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}} = 0 = \boldsymbol{A}^T\left(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{\theta}\right),$$

as

$$\boldsymbol{A}^T\boldsymbol{b} = \boldsymbol{A}^T\boldsymbol{A}\boldsymbol{\theta},$$

and if the matrix $\boldsymbol{A}^T\boldsymbol{A}$ is invertible we have the solution

$$\boldsymbol{\theta} = \left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}\boldsymbol{A}^T\boldsymbol{b}.$$

If we then introduce the matrix

$$\boldsymbol{H} = \left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1},$$

we have then the following expression for the parameters θ_j (the matrix elements of $\boldsymbol{H}$ are h_{ij})

$$\theta_j = \sum_{k=0}^{p-1} h_{jk}\sum_{i=0}^{n-1}\frac{y_i}{\sigma_i}\frac{x_{ik}}{\sigma_i} = \sum_{k=0}^{p-1} h_{jk}\sum_{i=0}^{n-1} b_i a_{ik}$$

We state without proof the expression for the uncertainty in the parameters θ_j as (we leave this as an exercise)

$$\sigma^2(\theta_j) = \sum_{i=0}^{n-1}\sigma_i^2\left(\frac{\partial\theta_j}{\partial y_i}\right)^2,$$

resulting in

$$\sigma^2(\theta_j) = \left(\sum_{k=0}^{p-1} h_{jk}\sum_{i=0}^{n-1} a_{ik}\right)\left(\sum_{l=0}^{p-1} h_{jl}\sum_{m=0}^{n-1} a_{ml}\right) = h_{jj}!$$