Analysis of Regression and Resampling Methods

Håkon Olav Torvik, Vetle Vikenes and Sigurd Sørlie Rustad

University of Oslo
Norway
October 5, 2021

CONTENTS

I. INTRODUCTION

Regression analysis is a statistical method for fitting a function to data. It is useful for building mathematical models to explain observations. There are several regression methods to achieve this, all with their strengths and weaknesses. We will in this paper study three different methods; ordinary least squares, Ridge and Lasso regression. All the code, results and instructions on running the code can be found in our GitHub repository here[1]

Larger datasets contain more information, giving more accurate models. However, real-world datasets usually have a fixed size, and getting more is practically impossible. For smaller datasets it is then useful to have tools mitigating the effects of little data. Resampling methods does this by running the same data through the regression, without over-fitting the model to the sample data. In addition to the regression methods, we will also study the effect of bootstrapping and cross-validating the data.

In order to study this, we need data to analyze. We will use two datasets to study the different methods. One dataset we will generate ourselves using the Franke function, given by equation (1). To make the data more realistic we also add normally distributed noise. The other data is real terrain data from here[2]. Our model for both datasets will be a polynomial fit with an x and y dependence of the form $[x, y, x^2, y^2, xy, \dots]$.

$$\begin{aligned} f(x,y) =& \frac{3}{4}\exp\left(-\frac{(9x-2)^2}{4}-\frac{(9y-2)^2}{4}\right)+\frac{3}{4}\exp\left(-\frac{(9x+1)^2}{49}-\frac{(9x+1)^2}{49}-\frac{(9y+1)}{10}\right) \\ &+\frac{1}{2}\exp\left(-\frac{(9x-7)^2}{4}-\frac{(9y-3)}{4}\right)-\frac{1}{5}\exp\left(-(9x-4)^2-(9y-7)^2\right) \end{aligned} \tag{1}$$

II. ORDINARY LEAST SQUARE

Using the Franke Function (equation (1)) we generate our dataset. We pick $N = 100$ randomly distributed points for x and y, where our domain is $x, y \in [0, 1]$. As mentioned in the introduction we add stochastic noise to the function, using the normal distribution $\epsilon \cdot N(0, 1)$. Here ϵ is just a variable we can use to scale the noise, we choose $\epsilon = 0.2$. This gives us the dataset visualized in figure 1.

When we perform ordinary least square (OLS) regression we assume that our data follows a model such that

$$\mathbf{z} = X\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{2}$$

Where $\mathbf{z}$ is our data, X is our design matrix given by our model, $\boldsymbol{\beta}$ are parameters in the model and $\boldsymbol{\epsilon}$ is stochastic noise. One challenge in our case is that our data is two-dimensional, i.e. $\mathbf{z}$ has dimensions (N, N). We create a workaround by mapping our two-dimensional data, to a one dimensional $(N^2, 1)$ vector. This is an important note, and we will do this throughout the report. Note that our design matrix X will have dimensions (N^2, P) where P are the number of features.

---

[1] https://github.com/sigurdru/FYS-STK4155/tree/main/project1

[2] https://github.com/CompPhysics/MachineLearning/tree/master/doc/Projects/2021/Project1/DataFiles

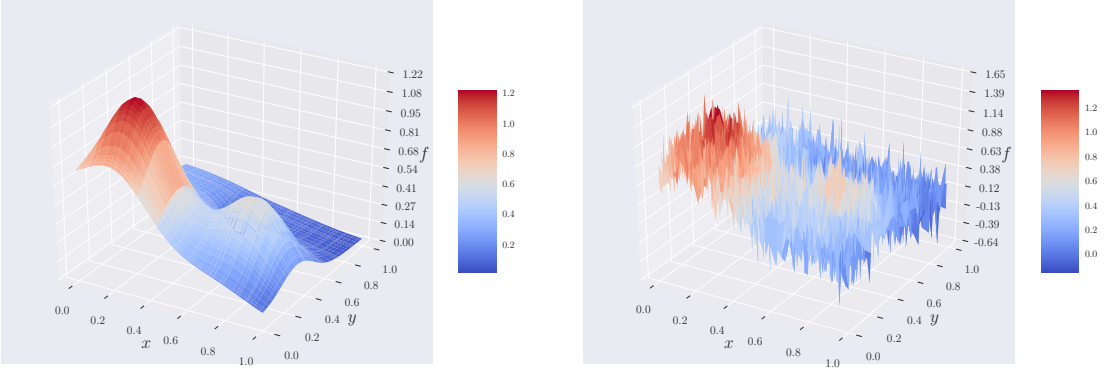

Figure 1. Here we have displayed our data along with the analytical function. The left figure shows the analytical Franke function plotted, and the right hand figure is the Franke function plotted with noise, i.e. the data we use.

As mentioned in the introduction our model will be a polynomial fit with an x and y dependence of the form $[x, y, x^2, y^2, xy, \dots]$. This gives us $P = (p+1)(p+2)/2$ number of features, where p is our maximum polynomial degree. Now for a given model

$$\tilde{\boldsymbol{z}} = X\tilde{\mathbf{y}},$$

we will have a mean square error (MSE) given by

$$C(\boldsymbol{\beta}) = \frac{1}{N}\sum_{i=0}^{N-1}(z_i - \tilde{z}_i) = \frac{1}{N}\left\{(\mathbf{z} - X\boldsymbol{\beta})^T(\mathbf{z} - X\boldsymbol{\beta})\right\}$$

III. BIAS-VARIANCE TRADE-OFF AND BOOTSTRAPPING

Bias-variance trade-off decomposition

We assume that our true data is generated from a noisy model with nromally distributed noise ϵ with a mean of zero and standard deviation σ^2, i.e.

$$\mathbf{y} = f(\mathbf{x}) + \boldsymbol{\epsilon}$$

We have approximated this function with our design matrix $\mathbf{X}$ and our parameters $\boldsymbol{\beta}$ such that our model becomes $\tilde{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta}$, where the values of $\boldsymbol{\beta}$ were obtained by optimizing the mean squared error via the cost function, given by

$$C(\mathbf{X}, \boldsymbol{\beta}) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \tilde{y}_i) = \mathbb{E}\left[(\mathbf{y} - \tilde{\mathbf{y}})^2\right]$$

where $\mathbb{E}$ is the expected value.

We want to show that the above expression can be written as

$$\mathbb{E}\left[(\mathbf{y}-\tilde{\mathbf{y}})^2\right] = \frac{1}{n}\sum_i (f_i - \mathbb{E}[\tilde{\mathbf{y}}])^2 + \frac{1}{n}\sum_i (\tilde{y}_i - \mathbb{E}[\tilde{\mathbf{y}}])^2 + \sigma^2$$

We begin by inserting our model expression for $\mathbf{y}$ and adding and subtracting $\mathbb{E}[\tilde{\mathbf{y}}]$ inside the expected value, before we square the expression.

$$\begin{aligned}\mathbb{E}\left[(\mathbf{y}-\tilde{\mathbf{y}})^2\right] &= \mathbb{E}\left[(f(\mathbf{x})+\epsilon-\tilde{\mathbf{y}}-\mathbb{E}[\tilde{\mathbf{y}}]+\mathbb{E}[\tilde{\mathbf{y}}])^2\right] = \mathbb{E}\left[((f(\mathbf{x})-\mathbb{E}[\tilde{\mathbf{y}}])+\epsilon+(\mathbb{E}[\tilde{\mathbf{y}}]-\tilde{\mathbf{y}}))^2\right] \\ &= \mathbb{E}\left[(f(\mathbf{x})-\mathbb{E}[\tilde{\mathbf{y}}])^2+\epsilon^2+(\mathbb{E}[\tilde{\mathbf{y}}]-\tilde{\mathbf{y}})^2\right] \\ &\quad + \mathbb{E}\left[2\epsilon(f(\mathbf{x})-\mathbb{E}[\tilde{\mathbf{y}}])+2\epsilon(\mathbb{E}[\tilde{\mathbf{y}}]-\tilde{\mathbf{y}})+2(f(\mathbf{x})-\mathbb{E}[\tilde{\mathbf{y}}])(\mathbb{E}[\tilde{\mathbf{y}}]-\tilde{\mathbf{y}})\right]\end{aligned}$$

where the cross terms have been written on a separate line since the expected value is linear. Next we will focus on the cross-terms. Since ϵ is normally distributed, it's expected value is simply the mean, which is zero in our case. The two cross terms involving ϵ is therefore zero, so we only need to consider

$$\mathbb{E}\left[(f(\mathbf{x})-\mathbb{E}[\tilde{\mathbf{y}}])(\mathbb{E}[\tilde{\mathbf{y}}]-\tilde{\mathbf{y}})\right] = \mathbb{E}\left[f(\mathbf{x})\mathbb{E}[\tilde{\mathbf{y}}]\right] - \mathbb{E}\left[f(\mathbf{x})\tilde{\mathbf{y}}\right] - \mathbb{E}\left[\mathbb{E}[\tilde{\mathbf{y}}]\mathbb{E}[\tilde{\mathbf{y}}]\right] + \mathbb{E}\left[\tilde{\mathbf{y}}\mathbb{E}[\tilde{\mathbf{y}}]\right]$$

Since the expected value of an expected value is just the expected value itself the last two terms in the above equation both become $\mathbb{E}[\tilde{\mathbf{y}}]^2$, canceling each other out. Using that $f(\mathbf{x})$ is a deterministic function, we have $\mathbb{E}[f(\mathbf{x})] = f(\mathbf{x})$. Expressing $f(\mathbf{x})$ in terms of its expected value, we can write the first two terms in the above equation as

$$\begin{aligned}\mathbb{E}\left[f(\mathbf{x})\mathbb{E}[\tilde{\mathbf{y}}]\right] - \mathbb{E}\left[f(\mathbf{x})\tilde{\mathbf{y}}\right] &= \mathbb{E}\left[\mathbb{E}\left[f(\mathbf{x})\right]\mathbb{E}[\tilde{\mathbf{y}}]\right] - \mathbb{E}\left[\mathbb{E}\left[f(\mathbf{x})\right]\tilde{\mathbf{y}}\right] \\ &= \mathbb{E}\left[f(\mathbf{x})\right]\mathbb{E}[\tilde{\mathbf{y}}] - \mathbb{E}\left[f(\mathbf{x})\right]\mathbb{E}[\tilde{\mathbf{y}}] = 0\end{aligned}$$

Hence, all the cross terms in the expected value cancel out, and we're left with

$$\mathbb{E}\left[(\mathbf{y}-\tilde{\mathbf{y}})^2\right] = \mathbb{E}\left[(f(\mathbf{x})-\mathbb{E}[\tilde{\mathbf{y}}])^2\right] + \mathbb{E}\left[(\mathbb{E}[\tilde{\mathbf{y}}]-\tilde{\mathbf{y}})^2\right] + \mathbb{E}\left[\epsilon^2\right]$$

Using that $\mathbb{E}[\epsilon^2] = \sigma^2$ and writing the expected values as sums we finally arrive at

$$\mathbb{E}\left[(\mathbf{y}-\tilde{\mathbf{y}})^2\right] = \frac{1}{n}\sum_i (f_i - \mathbb{E}[\tilde{\mathbf{y}}])^2 + \frac{1}{n}\sum_i (\tilde{y}_i - \mathbb{E}[\tilde{\mathbf{y}}])^2 + \sigma^2$$

which is what we wanted to show. git

IV. CROSS-VALIDATION

V. RIDGE REGRESSION

VI. LASSO REGRESSION

VII. ANALYSIS OF REAL DATA

VIII. TESTING

In order to make sure our algorithms are running correctly, it is necessary to perform tests. We did this by comparing our results to those produced by scikit-learn. First of all we generated some simpler data for testing, namely an exponential:

$$f_{\text{Test}}(x) = exp(x) + \epsilon. \tag{3}$$

Here ϵ denotes normally distributed noise, and x runs from $x_{\text{min}} = 0$ to $x_{\text{max}} = 1$ in $N = 50$ randomly distributed steps. This generates the testing data visualized in figure 2.

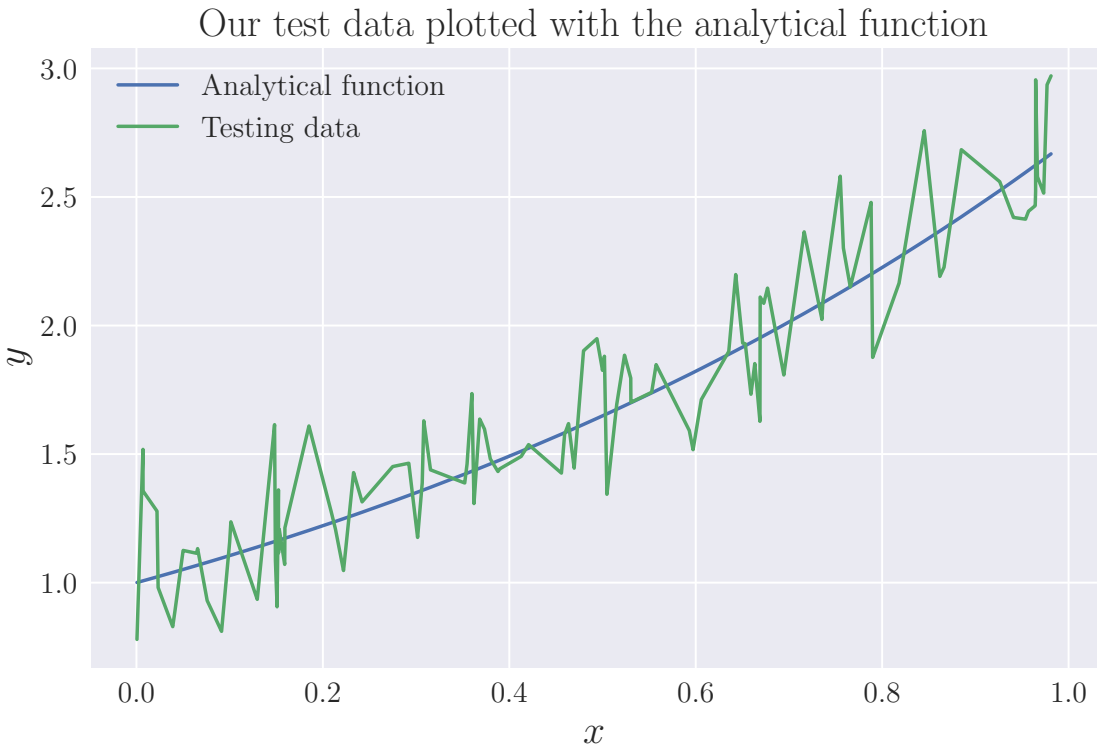

Figure 2. Here you we have plotted the testing data along with the analytical function.

First off we want to test the regression methods we have written.

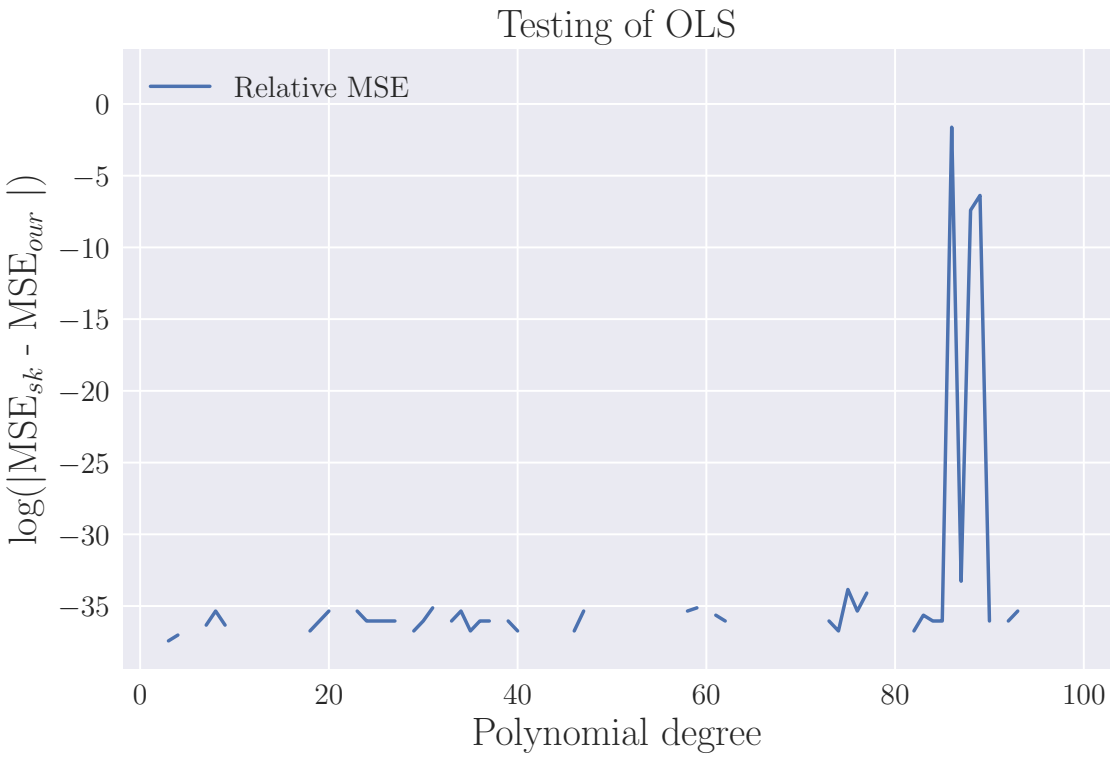

Figure 3. A really Awesome Image

---

[1] Ref.

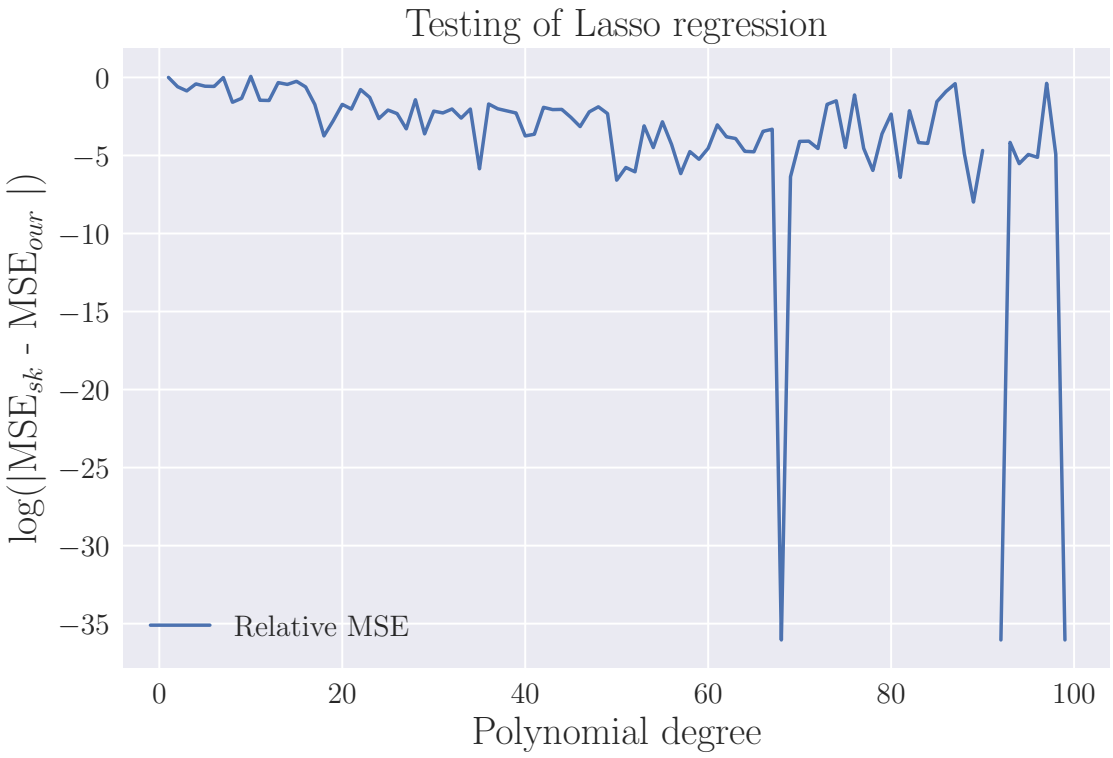

Figure 4. A really Awesome Image

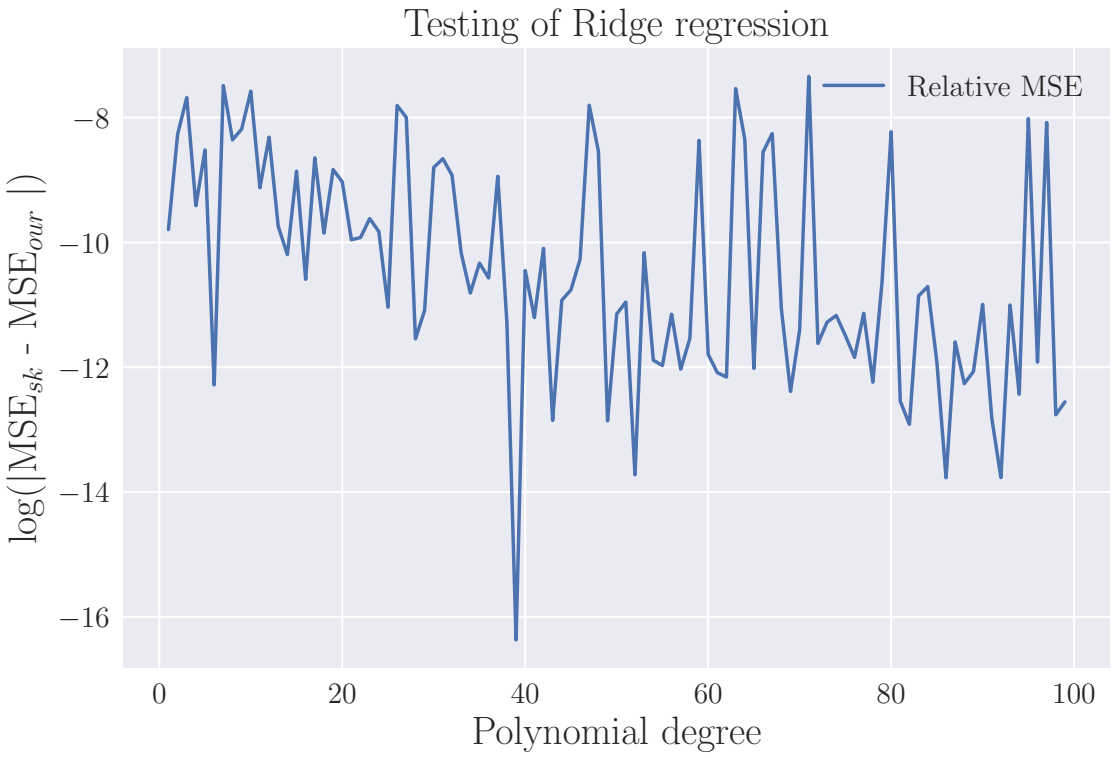

Figure 5. A really Awesome Image