University of Oslo

FYS-STK3155

Project 1

Eirik eirik@ifi.uio.no
Felix felix@ifi.uio.no
Brage Wiseth bragewi@ifi.uio.no

September 20, 2023

https://github.com/bragewiseth/MachineLearningProjects

Contents

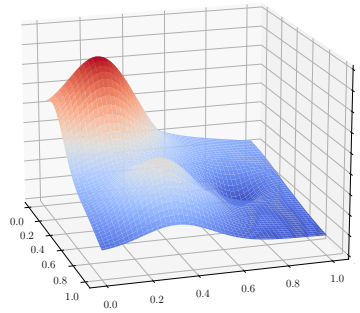
Abstract

In this project, we explored the use of different methods for solving linear regres- sion on topological data. We emplyed OLS, ridge and lasso methods for linear regression on simulated height data using *Franke's Function*, Where the goal was to fit polynomials to minimize the mean square error. As well as real topological data with the same goal.

Keywords: Linear Regression, Fitting Polynomials, Scaling, Bias-Variance

1. Introduction

Figure 1: Franke's Function



To get a smooth start on this project we begin to construct some fake topological data. This can be constructed with *Franke's Function*. This gives us a nice dataset to start linear regression. A natrual start would be to use ordinary least squares to try to fit a polynomial to some degree. This is done by constructing a design matrix X with the polynomial terms and then solving the linear regression equation

$$\boldsymbol{\beta} = (X^T X)^{-1} X^T \boldsymbol{y} \tag{1}$$

where $\boldsymbol{\beta}$ is the coefficients of the polynomial and $\boldsymbol{y}$ is the data. This is a nice start, but we can do better. We can add a penalty term to the cost function, this is called ridge regression.

$$\boldsymbol{\beta} = (X^T X + \lambda I)^{-1} X^T \boldsymbol{y} \tag{2}$$

where λ is a hyperparameter that we can tune to get a better fit. This is a nice start, but we can do better. We can add a penalty term to the cost function, this is called ridge regression.

$$\boldsymbol{\beta} = (X^T X + \lambda I)^{-1} X^T \boldsymbol{y} \tag{3}$$

where λ is a hyperparameter that we can tune to get a better fit. This is a nice start, but we can do better. We can add a penalty term to the cost function, this is called ridge regression.

$$\boldsymbol{\beta} = (X^T X + \lambda I)^{-1} X^T \boldsymbol{y} \tag{4}$$

where λ is a hyperparameter that we can tune to get a better fit. This is a nice start, but we can do better. We can add a penalty term to the cost function, this is called ridge regression.

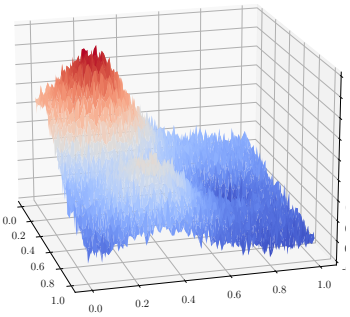


Figure 2: Franke's Function with noise

$$\boldsymbol{\beta} = (X^T X + \lambda I)^{-1} X^T \boldsymbol{y} \tag{5}$$

where λ is a hyperparameter that we can tune to get a better fit. This is a nice start, but we can do better. We can add a penalty term to the cost function, this is called ridge regression.

$$\boldsymbol{\beta} = (X^T X + \lambda I)^{-1} X^T \boldsymbol{y} \tag{6}$$

2. Ordinary Least Squares

2.1 Discussion on Scaling

Table 1: Unscaled sample design matrix fitting one-dimensional polynomial of degree 5

1.	0.	0.	0.	0.	0.
1.	0.25	0.0625	0.01562	0.00391	0.00098
1.	0.5	0.25	0.125	0.0625	0.03125
1.	0.75	0.5625	0.42188	0.31641	0.2373
1.	1.	1.	1.	1.	1.

Table 2: Scaled sample design matrix fitting one-dimensional polynomial of degree 5

0.	-1.41421	-1.0171	-0.83189	-0.728	-0.66226
0.	-0.70711	-0.84758	-0.7903	-0.71772	-0.65971
0.	0.	-0.33903	-0.49913	-0.56348	-0.58075
0.	0.70711	0.50855	0.29116	0.10488	-0.0433
0.	1.41421	1.69516	1.83016	1.90431	1.94603

MSE for OLS on unscaled data:	0.010349396022903145
MSE for OLS on scaled data:	0.010349396024145656
MSE for Ridge on unscaled data:	0.02106077418650843
MSE for Ridge on scaled data:	0.01782525371566323

the code for genertating this output [1]

First x_{unscaled} and x_{scaled} is not that different, the original data was close to zero-centerd and not that spread out, which means that initaially by just looking at the data scaling is not that nececary. When we add the polynomial terms we can now see that some of the entries of X_{unscaled} get really small as an exsample $0.1^5 = 0.00001$ this makes the collums of X_{unscaled} live in their own order of magnitude and scaling should be considerd to bring them back to the same ish order of magnitude [2]. The act of not scaling results in $\boldsymbol{\beta}$ spanning from -50 to 48 while scaling gives a smaller span from -10 to 13. Now this alone may not justify why we should scale this dataset, as scaled and unscaled OLS yields the same MSE. However when doing ridge regression the cost function is directly dependent on the magnitude of β_i. Now with each β_i variying alot, some are getting more penaliesed than others. As we can see the MSE for unscaled data is much higher than for scaled data in the ridge case. This leads us to conclude that we should scale the data, making it easier to tweek λ and giving us nicer numbers to work with.

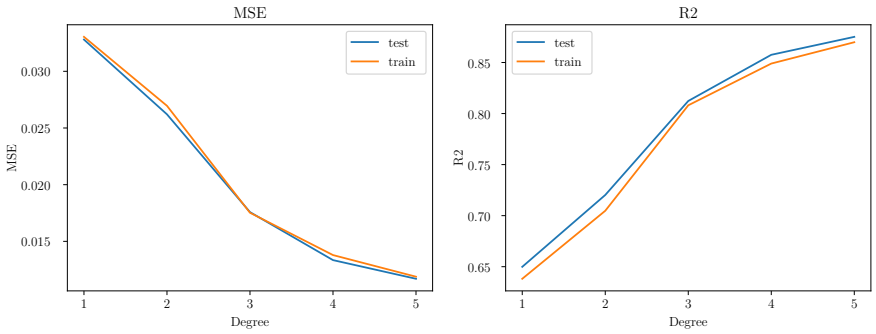


Figure 3: Franke's Function

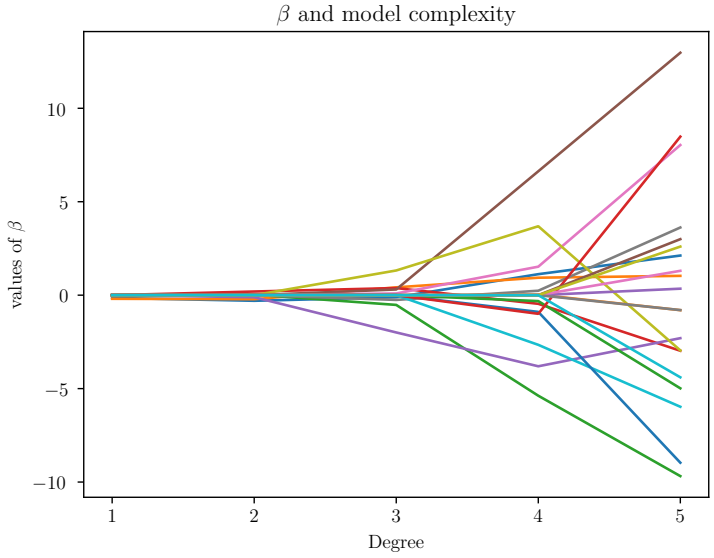


Figure 4: Franke's Function

3

2.2 Analysis

3. Ridge

4. Lasso

5. Paper & Pencil

The assumption we have made is that there exists a continuous function f (x) and a normal distributed error N (0, 2) which describes our data y = f (x) + We then approximate this function f (x) with our model ˜y from the solution of the linear regression equations (ordinary least squares OLS), that is our function f is approximated by ˜y where we minimized (y − ˜y)2, with ˜y = X . The matrix X is the so-called design or feature matrix. Show that the expectation value of y for a given element i E(yi) = j xij j = Xi,∗ , and that its variance is Var(yi) = 2. Hence, yi N (Xi,∗ , 2), that is y follows a normal distribution with mean value X and variance 2. With the OLS expressions for the optimal parameters ˆ show that E(ˆ) = . Show finally that the variance of is Var(ˆ) = 2 (XT X)−1. We can use the last expression when we define a so-called confidence interval for the parameters . A given parameter j is given by the diagonal matrix element of the above matrix.

We can assume that y follows some function f with some noise ϵ

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

$$\mathbb{E}[\mathbf{y}] = \mathbb{E}[f(\mathbf{x}) + \epsilon] = \mathbb{E}[f(\mathbf{x})] + \mathbb{E}[\epsilon_i]$$

The expected value of ϵ_i is 0, $f(x)$ is a non-stochastic variable and is approimated by $\boldsymbol{X\beta}$

$$\mathbb{E}[y_i] = \boldsymbol{X_{i,*}\beta}$$

The variance is defined as

$$\begin{aligned}
Var(y_i) &= \mathbb{E}\big[(y_i - \mathbb{E}[y_i])^2\big] = \mathbb{E}\left[y_i^2 - 2y_i\mathbb{E}[y_i] + \mathbb{E}[y_i]^2\right] = \mathbb{E}[y_i^2] - \mathbb{E}[y_i]^2 \\
Var(y_i) &= \mathbb{E}\big[\left(\boldsymbol{X}_{i,*}\boldsymbol{\beta}\right)^2 + 2\epsilon\boldsymbol{X}_{i,*}\boldsymbol{\beta} + \epsilon^2\big] - (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 \\
Var(y_i) &= (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 + 2\mathbb{E}[\epsilon]\boldsymbol{X}_{i,*}\boldsymbol{\beta} + \mathbb{E}[\epsilon^2] - (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 \\
Var(y_i) &= \mathbb{E}[\epsilon^2] = \sigma^2
\end{aligned}$$

for the expected value of $\boldsymbol{\beta}$ we can insert the definition of $\boldsymbol{\beta}$ from earlier

$$\mathbb{E}[\hat{\boldsymbol{\beta}}] = \mathbb{E}[(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}] = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbb{E}[\mathbf{Y}] = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} = \boldsymbol{\beta}.$$

$$\begin{aligned}
Var(\hat{\boldsymbol{\beta}}) &= \mathbb{E}\{[\boldsymbol{\beta} - \mathbb{E}(\boldsymbol{\beta})][\boldsymbol{\beta} - \mathbb{E}(\boldsymbol{\beta})]^T\} \\
&= \mathbb{E}\{[(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} - \boldsymbol{\beta}]\,[(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} - \boldsymbol{\beta}]^T\} \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\,\mathbb{E}\{\mathbf{y}\,\mathbf{y}^T\}\,\mathbf{X}\,(\mathbf{X}^T\mathbf{X})^{-1} - \boldsymbol{\beta}\,\boldsymbol{\beta}^T \\
&= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\,\{\mathbf{X}\,\boldsymbol{\beta}\,\boldsymbol{\beta}^T\,\mathbf{X}^T + \sigma^2\}\,\mathbf{X}\,(\mathbf{X}^T\mathbf{X})^{-1} - \boldsymbol{\beta}\,\boldsymbol{\beta}^T \\
&= \boldsymbol{\beta}\,\boldsymbol{\beta}^T + \sigma^2\,(\mathbf{X}^T\mathbf{X})^{-1} - \boldsymbol{\beta}\,\boldsymbol{\beta}^T \;=\; \sigma^2\,(\mathbf{X}^T\mathbf{X})^{-1},
\end{aligned}$$

1. https://github.com/bragewiseth/MachineLearningProjects/blob/main/project1/src/linearRegression.py[5]
2. We would imagine that keeping everything in the same 0 order of magnitude is something that the computer likes, perhaps improving performance

6. Bias & Variance

Acknowledgments

Appendix A.

Bibliography

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1 edition, 2007.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. Book in preparation for MIT Press.

[3] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[4] Wessel N. van Wieringen. Lecture notes on ridge regression, 2023.

[5] Brage Wiseth, Eirik ?, and Felix ? MachineLearningProjects. `https://github.com/bragewiseth/MachineLearningProjects`, September 2023.