University of Oslo

FYS-STK3155

Project 1

Brage Wiseth bragewi@ifi.uio.no
Felix Cameren Heyerdahl felixch@ifi.uio.no
Eirik Bjørnson Jahr eiribja@ifi.uio.no

October 10, 2023

https://github.com/bragewiseth/MachineLearningProjects

Contents

Abstract

In this paper, we delve into the realm of machine learning model optimization and evaluation. Our study encompasses various regression techniques, including Ordinary Least Squares (OLS), Ridge, and Lasso regression, to analyze their effectiveness in handling simple and more complex datasets. Additionally, we employ bootstrap resampling and cross-validation methodologies to rigorously assess model performance and enhance generalization. A significant portion of our investigation is dedicated to understanding the delicate balance between bias and variance. We explore how regularization methods like Ridge and Lasso impact bias-variance trade-offs, offering insights into the stability and predictive power of these models. Furthermore, we provide empirical evidence of the benefits of cross-validation and bootstrap techniques in mitigating overfitting and improving model robustness. We found that { ..results.. }. Additionally we verify and compare our findings with well established theory and libraris such as SKLearn.

Keywords: Linear Regression, Scaling, Bias & Variance

1. Introduction

Machine learning has emerged as a powerful tool in data analysis, providing the ability to uncover complex patterns and relationships in diverse datasets. But, at its core, machine learning is all about finding functions that capture the underlying structure of the data. The use of machine learning algorithms to approximate functions is the essence of this paper.

Our motivation for this research lies in the exploration of machine learning techniques to approximate the terrain on our planet, which can perhaps be described by such a function. Earth's terrain exhibits peaks and troughs, hills and valleys, much like some polynomial functions. Fortunately, we can employ standard linear regression techniques to approximate polynomials, but the terrain presents its own set of challenges. Firstly, the terrain's true underlying function may not be a polynomial at all, and its complexity may vary significantly from one location to another. Secondly, our landscape is teeming with small, intricate details. Some regions are characterized by flat and smooth surfaces, while others are marked by rough and uneven terrain. Focusing too much on these minute details can lead to model overfitting, making it crucial to strike a careful balance between model complexity and generalization. In this context, regularization and resampling techniques, including Ridge and Lasso regression with bootstrap and cross validation, have proven indispensable. By introducing regularization and resampling, we aim to find the sweet spot between bias and Variance. And getting the best predictions we can with our assumptions.
To embark on this exploration, we will begin with a simpler case: "Franke's function." which mimics our real terrain data. This function serves as a foundational starting point, allowing us to assess our model's performance in a controlled environment before venturing into the complexity of real-world terrain data. Through this gradual progression, we provide ourselves wih a framework that can be applied to more complex and varied real-world terrain datasets.

Data: We begin by introducing the dataset used for our analysis, highlighting data collection and preprocessing procedures. Understanding the characteristics of the terrain data is fundamental to our modeling endeavor.

Methods and Scaling: Next, we delve into the methodology, encompassing the implementation of polynomial regression models and the application of regularization techniques such as Ridge and Lasso. Additionally, we will discuss the importance of proper scaling for model stability and convergence.
Results: In this section, we will present the outcomes of our experiments, showcasing the performance of different models and regularization techniques. Through empirical evidence, we aim to provide insights into the effectiveness of our approach.
Bias-Variance Trade-off: A significant portion of our study will revolve around the critical concept of bias and variance. We'll explore how regularization methods influence this trade-off and delve into the fine balance between model complexity and generalization.
Conclusion: Finally, we will summarize the key findings and their implications for terrain modeling with machine learning. Our conclusion will underscore the importance of regularization in achieving accurate representations of complex terrains and provide a perspective on future research directions.

code for generating all figures and data can be found at /MACHINELEARNINGPROJECTS/PROJECT1/SRC

2. Data

The data we will be using consists of 100 points generated using the Franke function. The Franke function is a weighted sum of four exponentials given by

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

We will add some noise to the data to better simulate real world data. The data is generated by sampling x and y from a uniform distribution on the interval $[0, 1]$. The noise is sampled from a normal distribution with mean 0 and standard deviation $\sigma = 0.1$. ««««< HEAD The noise is sampled from a normal distribution with mean 0 and standard deviation $\sigma = 0.1$. The data is generated by sampling x and y from a uniform distribution on the interval

$$0, 1$$

. This synthetic data is divided into training and testing sets, with the testing set comprising twenty percent and the training set comprising eighty percent of the total data. In addition, we will incorporate genuine topographic data from a specific area of Norway. However, to reduce computational requirements when running the program, we've reduced the size of this real topographic dataset from 6.5 million points to 10,000 points. ======= »»»»> 8088bb782375c34c0acabd1116e5d54bc57f5250

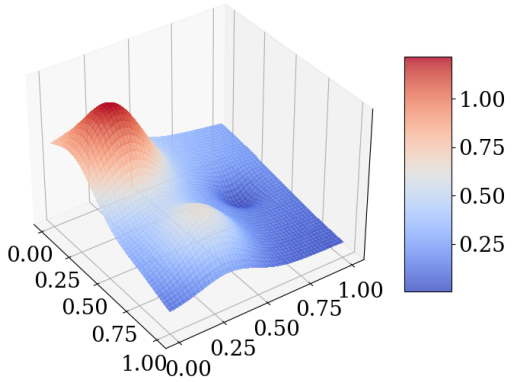


Figure 1: True Function

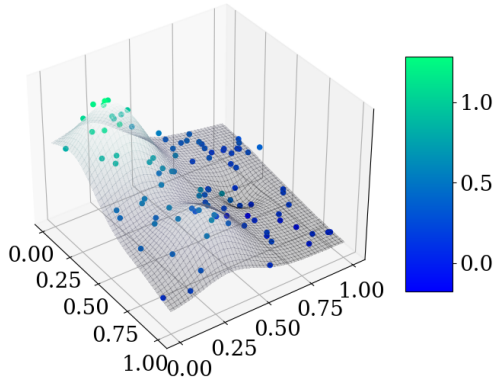


Figure 2: Our Synthetic Data

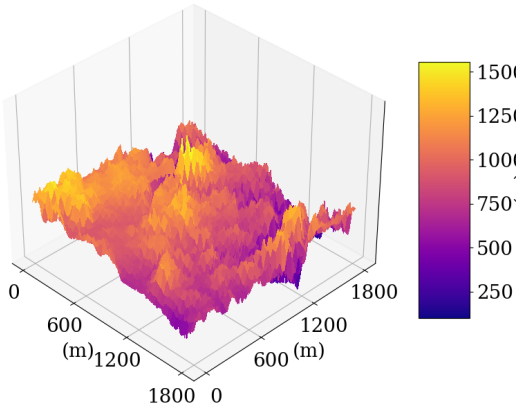


Figure 3: True Function

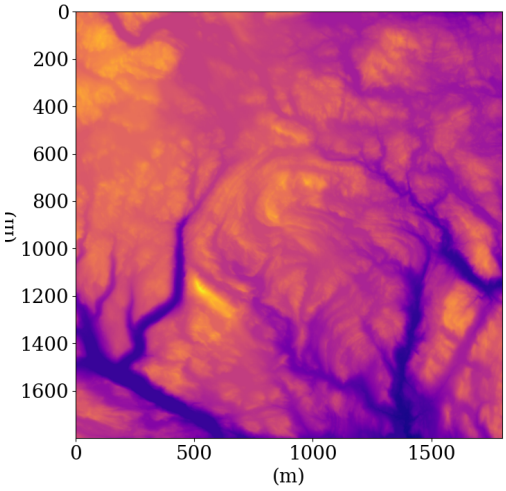


Figure 4: Our Synthetic Data

3. linear regression models

Linear models assume that the relationship between the input variables (predictors or features) and the output variable (response or target) is linear. The predictions of the model is a linear combination of the input variables with some coefficients. The coefficients are esti-

mated from the training data using a method called least squares. The least squares method is used to find the coefficients that minimizes the residual sum of squares between the observed responses in the dataset, and the responses predicted by the linear approximation. The residual sum of squares is given by

$$RSS(\beta) = \sum_{i=0}^{n-1} (y_i - \tilde{y}_i)^2$$

where y_i is the observed response for observation i, $\tilde{y}_i$ is the predicted response for observation i

$$\bar{y} = \frac{1}{n} \sum_{i=0}^{n-1} y_i$$

We can also take the mean of the residual sum of squares, which is called the mean squared error (MSE). This is what we will use to evaluate our models. The MSE is given by

$$\text{MSE}(\mathbf{y}, \tilde{\mathbf{y}}) = \frac{1}{n} \sum_{i=0}^{n-1} (y_i - \tilde{y}_i)^2$$

We wil also use the R^2 score to evaluate our models. The R^2 score is a statistical measure of how close the data are to the fitted regression line. The R^2 score is given by

$$R^2(\mathbf{y}, \tilde{\mathbf{y}}) = 1 - \frac{\sum_{i=0}^{n-1} (y_i - \tilde{y}_i)^2}{\sum_{i=0}^{n-1} (y_i - \bar{y})^2}$$

The R2 score is a number between 0 and 1, where 1 indicates that the model fits the data perfectly. A negative R2 score indicates that the model is worse than just predicting the mean of the data.

3.1 Ordinary Least Squares (OLS)

By minimizing the MSE, we find the coefficients that minimizes the residual sum of squares. This is done by taking the derivative of the MSE with respect to the coefficients β_j and setting it equal to zero. This gives us the optimal parameters

$$\hat{\boldsymbol{\beta}}_{\text{OLS}} = \left(\mathbf{X}^T \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{y}$$

The derivation of this can be found in appendix 6

We can then use these parameters to predict the response $\tilde{\mathbf{y}}$ for a new set of predictors $\mathbf{X}$. The predictions is as previously mentioned a linear combination of the predictors and the coefficients.

$$\tilde{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta}$$

Wich is a compact way of writing

$$\tilde{y}_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_{p-1} x_{i,p-1}$$

where p is the number of predictors.

3.2 Ridge & Lasso Regression

We can add add regularization to the linear regression model to help keep our parameters in check. This is done by adding a penalty term to the cost function. The penalty term is a function of the coefficients β_j. The penalty term is called the regularization term, and the parameter λ is called the regularization parameter. The regularization parameter determines how much the coefficients are penalized. The cost function for Ridge regression is given by

$$C(\beta) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \tilde{y}_i)^2 + \lambda\sum_{j=0}^{p-1}\beta_j^2 = \frac{1}{n}||\mathbf{y} - \mathbf{X}\beta||_2^2 + \lambda||\beta||_2^2$$

$$\hat{\beta}_{\text{Ridge}} = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y},$$

Lasso regression is similar to Ridge regression, with the difference being that the penalty term is the sum of the absolute values of the coefficients β_j, also called the L1 norm. The cost function for Lasso regression is given by

$$C(\beta) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i - \tilde{y}_i)^2 + \lambda\sum_{j=0}^{p-1}|\beta_j| = \frac{1}{n}||\mathbf{y} - \mathbf{X}\beta||_2^2 + \lambda||\beta||_1$$

We won't be utilizing our custom Lasso implementation and derivation; instead, we will opt for the Lasso implementation provided by the Scikit-Learn library.

3.3 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

Consider Table 1, Table 2 and Table 3. At first glance, there isn't a big difference between the unscaled data and the scaled data. This is because the original data was already close to having a mean close to zero and not being spread out too much. So, you might think that scaling the data isn't necessary. However, when we introduce polynomial terms, we notice that some values in xunscaled become extremely small, for example, 0.15

Table 3:

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

turns into 0.00001. This means that the columns of xunscaled have vastly different orders of magnitude, and this calls for scaling to bring them back to a similar scale. If we don't scale, the coefficients ($\boldsymbol{\beta}$) can range widely, from -50 to 48. But when we scale the data, this range narrows down to -10 to 13. Now, this alone might not seem like a strong reason to scale the data, especially when we use plain linear regression (OLS), as it doesn't change the Mean Squared Error (MSE) much.
However, things change when we use Ridge regression. In Ridge, the regularization cost depends on the magnitude of i. With unscaled data, where i varies a lot, some coefficients get penalized more than others. This results in a significantly higher MSE for unscaled data compared to scaled data in Ridge regression. So, to make it easier to tune the regularization parameter (λ) and to ensure fair treatment of coefficients in Ridge regression, it's a good idea to scale the data. This not only simplifies the regularization process but also leads to better predictive performance.

4. Results and Discussion

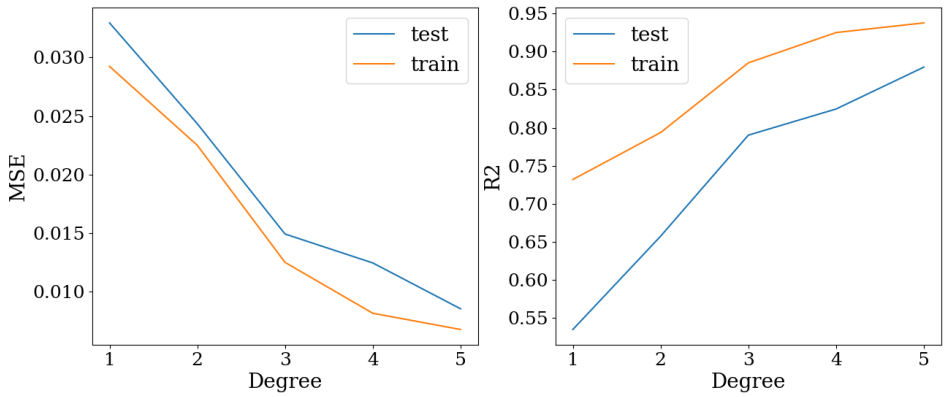
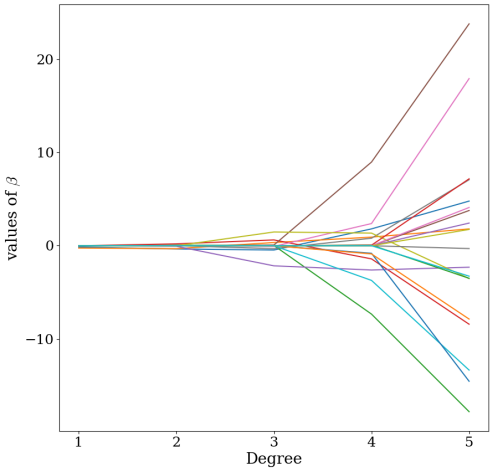
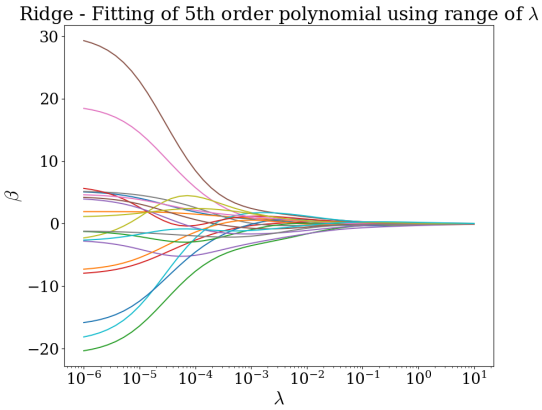
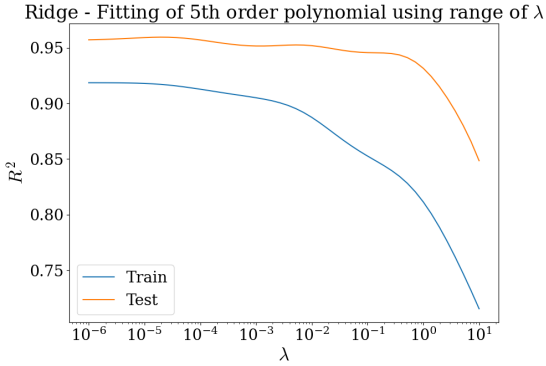


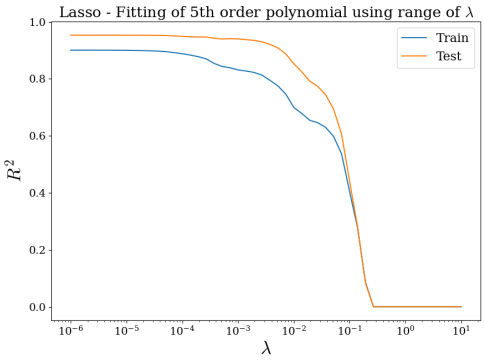
Figure 5: R^2 and MSE with increasing complexity

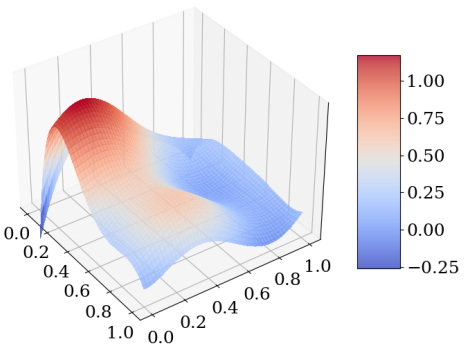
From Figure 5 we can see that the MSE and R^2 both improve as we increase the model complexity, we have yet to see any signs of overfitting which indicates that we can increase the complexity even further.

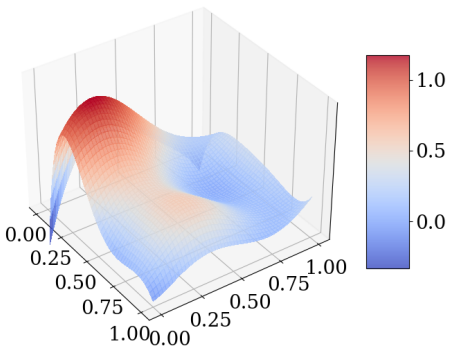

values of β
Degree

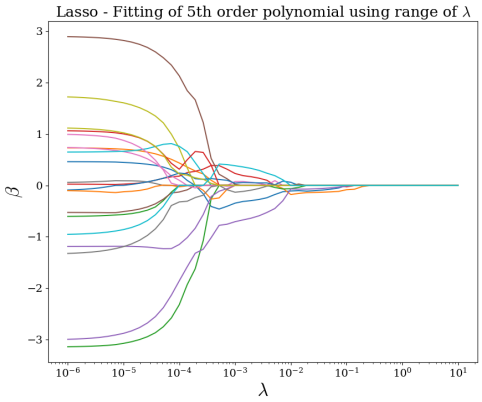

Ridge - Fitting of 5th order polynomial using range of λ
β
λ

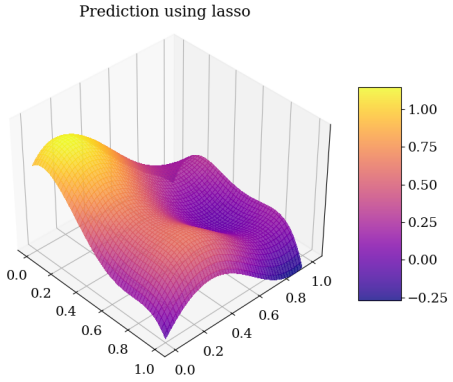

Ridge - Fitting of 5th order polynomial using range of λ
R^2
λ
Train
Test

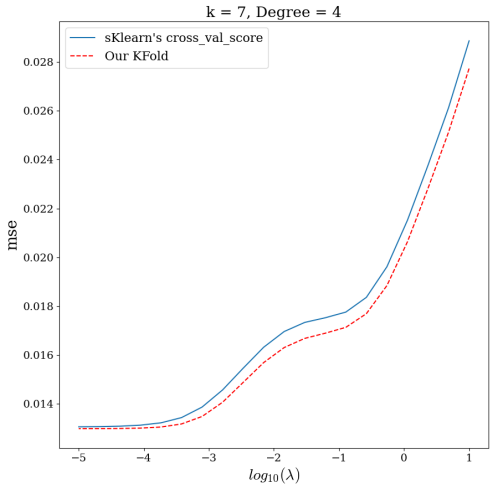

Lasso - Fitting of 5th order polynomial using range of λ
R^2
λ
Train
Test











4.1 Confidence Interval

The assumption we have made is that there exists a continuous function $f(x)$ and a normal distributed error $\epsilon \sim n(0, \sigma^2)$ which describes our data

$$y_i = f(x_i) + \epsilon_i$$

we then approximate this function $f(x)$ with our model $\tilde{y}$ from the solution of the linear regression equations (ordinary least squares ols), that is our function f is approximated by $\tilde{y}$ where we minimized $(y - \tilde{y})^2$, with $\tilde{y} = X\beta$ the matrix x is the so-called design or feature matrix. yi n (xi,∗ , 2), that is y follows a normal distribution with mean value x and variance 2. we can use this 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. 6

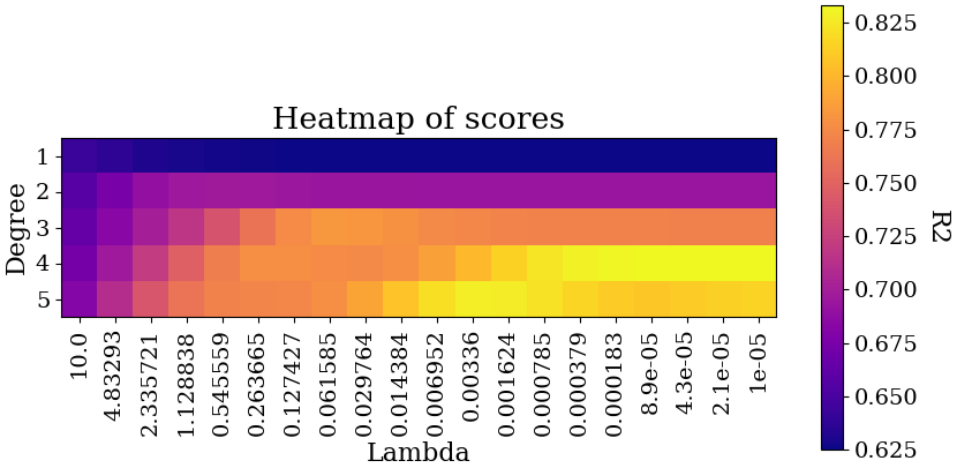


Figure 6:

5. Bias & Variance

It is possible to perform a so called bias-variance decomposition of our error, see for example Hastie et al. (2009), chapter 7.3 [3]. By doing so we can show that the mean squared error can be written as the following

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

$$\begin{aligned}\mathbb{E}[\mathbf{y}\tilde{\mathbf{y}}] &= \mathbb{E}[\mathbf{f}\tilde{\mathbf{y}} + \epsilon\tilde{\mathbf{y}}] \\ &= \mathbb{E}[\mathbf{f}\tilde{\mathbf{y}}] + \mathbb{E}[\epsilon\tilde{\mathbf{y}}] \\ &= \mathbb{E}[\mathbf{f}\tilde{\mathbf{y}}] + \mathbb{E}[\epsilon]\mathbb{E}[\tilde{\mathbf{y}}] \\ &= \mathbf{f}\mathbb{E}[\tilde{\mathbf{y}}]\end{aligned}$$

$$\begin{aligned}\mathbb{E}[\mathbf{y}^2] &= \mathbb{E}[\mathbf{f} + \epsilon]^2 = \mathbb{E}[\mathbf{f}^2 + 2\mathbf{f}\epsilon + \epsilon^2] \\ &= \mathbb{E}[\mathbf{f}^2] + 2\mathbb{E}[\mathbf{f}\epsilon] + \mathbb{E}[\epsilon^2] \\ &= \mathbb{E}[\mathbf{f}^2] + 2\mathbb{E}[\mathbf{f}]\mathbb{E}[\epsilon] + \mathbb{E}[\epsilon^2] \\ &= \mathbf{f}^2 + \sigma^2\end{aligned}$$

Bringing it all together we get

$$\begin{aligned}\mathbb{E}\left[(\mathbf{y}-\tilde{\mathbf{y}})^2\right] &= \mathbb{E}[\mathbf{y}^2] - 2\mathbb{E}[\mathbf{y}\tilde{\mathbf{y}}] + \mathbb{E}[\tilde{\mathbf{y}}^2] \\ &= \mathbf{f}^2 + \sigma^2 - 2\mathbf{f}\mathbb{E}[\tilde{\mathbf{y}}] + \mathbb{E}[\tilde{\mathbf{y}}^2] \\ &= \mathbf{f}^2 + \sigma^2 - 2\mathbf{f}\mathbb{E}[\tilde{\mathbf{y}}] + \mathbb{E}[\tilde{\mathbf{y}}]^2 + \mathrm{var}(\tilde{\mathbf{y}}) \\ &= (\mathbf{f} - \mathbb{E}[\tilde{\mathbf{y}}])^2 + \mathrm{var}(\tilde{\mathbf{y}}) + \sigma^2 \\ &= \mathrm{bias}[\tilde{\mathbf{y}}]^2 + \mathrm{var}(\tilde{\mathbf{y}}) + \sigma^2\end{aligned}$$

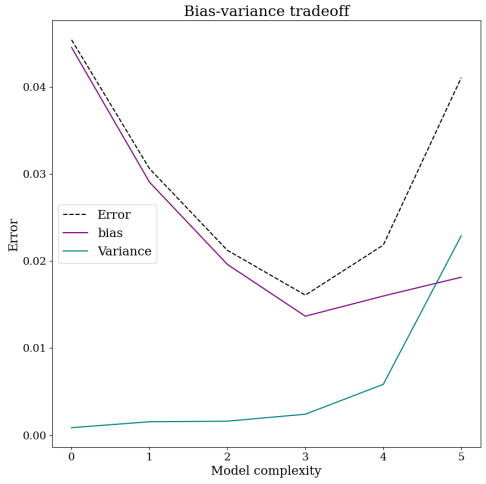


Figure 7:

To compute the expected value of $\tilde{\mathbf{y}}$, we employ the bootstrap method, which generates multiple samples for $\tilde{\mathbf{y}}$ and then calculates the mean of this distribution. In statistical terms, "variance" characterizes the spread or dispersion of our observations. In our context, each $\tilde{y}_i$ can be considered an individual prediction. Conceptually, bias resembles the mathematical definition of variance. The bias term measures the difference between the actual values in $\mathbf{y}$ and the expected values in $\tilde{\mathbf{y}}$. If we approach thison an individual observation level, a high bias would result in a broad range around $\tilde{\mathbf{y}}$ within which y_i would fall.

On the other hand, the variance term quantifies how much $\tilde{\mathbf{y}}$ deviates from its own mean. In simpler terms, a model with high variance would exhibit significant variation in the predicted values $\tilde{\mathbf{y}}$ from one test set to another, consequently leading to a substantial variance in scores. These two terms, bias and variance, are somewhat opposing forces. A model with high bias is likely to have low variance, and vice versa. As model designers, we have the flexibility to tune this trade-off. Depending on our priorities, whether we prioritize lower bias or lower variance, we can select an appropriate model to strike the right balance for our specific application.

Lorem ipsum dolor sit amet, qui minim labore adipisicing minim sint cillum sint consectetur cupidatat.

6. Conclusion

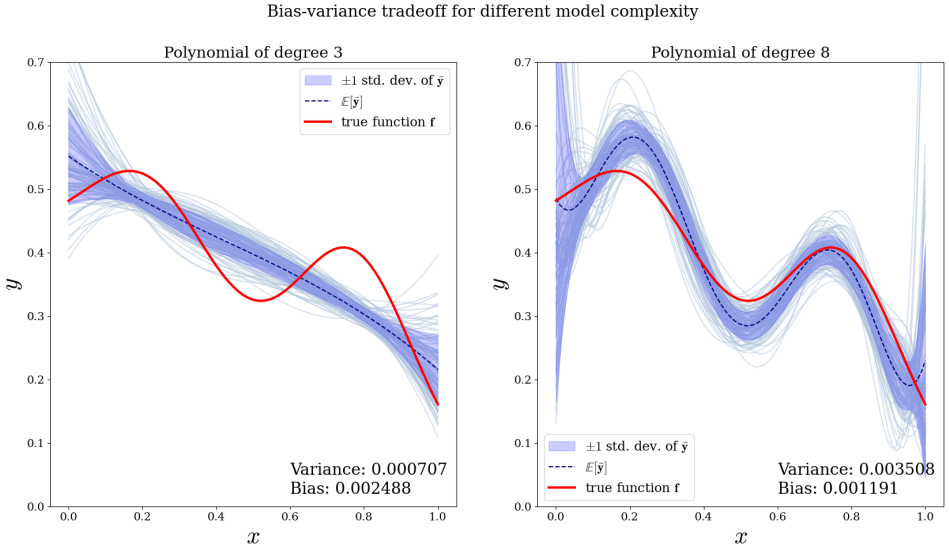


Figure 8: Bias & Variance Showcase, Using 2D Slice Of Our Data

Appendix

Derivation of The Optimal Parameters $\boldsymbol{\beta}$

In Section 3.1 we ended up with our optimal parmeters $\boldsymbol{\beta}$ using the MSE as our cost function, in this section we will go into detail about our derivations. The parameters $\boldsymbol{\beta}$ are found by minimizing our cost function (the MSE), this is done by taking the derivative of the MSE with respect to the coefficients β_j and setting it equal to zero.

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{n}\sum_{i=0}^{n-1}(y_i-\tilde{y}_i)^2$$

can also be written as

$$\begin{aligned}
&\min_{\beta\in\mathbb{R}^p}\frac{1}{n}||\mathbf{y}-\mathbf{X}\boldsymbol{\beta}||_2^2 \\
&=\frac{2}{n}\mathbf{X}^T\left(\mathbf{X}\boldsymbol{\beta}-\mathbf{y}\right)=0 \\
&=\frac{2}{n}\left(\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}-\mathbf{X}^T\mathbf{y}\right)=0 \\
\boldsymbol{\beta}&=\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}
\end{aligned}$$

$$\hat{\boldsymbol{\beta}}_{\text{OLS}}=\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y},$$

For the ridge case we have the following cost function

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p}\frac{1}{n}||\mathbf{y}-\mathbf{X}\boldsymbol{\beta}||_2^2+\lambda||\boldsymbol{\beta}||_2^2$$

We know that

$$\min_{\beta\in\mathbb{R}^p}\frac{1}{n}||\mathbf{y}-\mathbf{X}\boldsymbol{\beta}||_2^2=\frac{2}{n}\mathbf{X}^T\mathbf{X}\boldsymbol{\beta}-\frac{2}{n}\mathbf{X}^T\mathbf{y}$$

We can also clearly see that

$$\min_{\beta\in\mathbb{R}^p}\frac{1}{n}\lambda||\beta||_2^2=\frac{2}{n}\lambda\beta$$

Adding these two we get

$$\frac{\partial\left[\frac{1}{n}||\mathbf{y}-\mathbf{X}\boldsymbol{\beta}||_2^2+\lambda||\boldsymbol{\beta}||_2^2\right]}{\partial\beta}=0$$
$$=\frac{2}{n}\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)\boldsymbol{\beta}-\frac{2}{n}\mathbf{X}^T\mathbf{y}\implies\boldsymbol{\beta}=\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

$$\hat{\boldsymbol{\beta}}_{\text{Ridge}}=\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

SVD

Use the singular value decomposition of an $n\times p$ matrix $\mathbf{X}$ (our design matrix)

$$\mathbf{X}=\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices of dimensions $n\times n$ and $p\times p$, respectively, and $\boldsymbol{\Sigma}$ is an $n\times p$ matrix which contains the ingular values only. This material was discussed during the lectures of week 35.

Show that you can write the OLS solutions in terms of the eigenvectors (the columns) of the orthogonal matrix $\mathbf{U}$ as

$$\tilde{\mathbf{y}}_{\text{OLS}}=\mathbf{X}\beta=\sum_{j=0}^{p-1}\mathbf{u}_j\mathbf{u}_j^T\mathbf{y}$$

$$\mathbf{X}^T\mathbf{X}=\mathbf{V}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T=\mathbf{V}\boldsymbol{\Sigma}^T\boldsymbol{\Sigma}\mathbf{V}^T\qquad\qquad=\mathbf{V}\boldsymbol{\Sigma}^2\mathbf{V}^T$$

$$\tilde{\mathbf{y}}_{\text{OLS}}=\mathbf{X}\beta=\mathbf{X}\left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}=\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\left(\mathbf{V}\boldsymbol{\Sigma}^2\mathbf{V}^T\right)^{-1}\mathbf{V}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{y}=\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{V}^{T^{-1}}\boldsymbol{\Sigma}^{2^{-1}}\mathbf{V}^{-1}\mathbf{V}\boldsymbol{\Sigma}^T\mathbf{U}^T\mathbf{y}$$

using the orthogonality of $\mathbf{V}$ and $\mathbf{U}$ we get. Multiplying with $\boldsymbol{\Sigma}$ removes columns from $\mathbf{U}$ with eigenvalues equal to zero.

$$\tilde{\mathbf{y}}_{\text{OLS}}=\mathbf{U}\mathbf{U}^T\mathbf{y}=\sum_{j=0}^{p-1}\mathbf{u}_j\mathbf{u}_j^T\mathbf{y}$$

Math Behind the Confidence Interval, Section 4.1

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

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

$$\tilde{\mathbf{y}} = \mathbf{X}\beta.$$

$$\mathbb{E}(y_i) = \sum_j x_{ij}\beta_j = \mathbf{X}_{i,*}\,\beta$$

$$Var(y_i) = \sigma^2$$

$$Var(\hat{\beta}) = \sigma^2\,(\mathbf{X}^T\mathbf{X})^{-1}.$$

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

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

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

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

The variance is defined as

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

for the expected value of β we can insert the definition of β from earlier

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

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

ridge

$$\mathbb{E}[\hat{\beta}^{\text{Ridge}}] = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_{pp})^{-1}(\mathbf{X}^\top\mathbf{X})\beta.$$

$$Var[\hat{\beta}^{\text{Ridge}}] = \sigma^2[\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}]^{-1}\mathbf{X}^T\mathbf{X}\{[\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}]^{-1}\}^T,$$

$$\mathbb{E}[\hat{\beta}^{\text{Ridge}}] = \mathbb{E}\left[\left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_{pp}\right)^{-1}\mathbf{X}^T\mathbf{y}\right] = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_{pp}\right)^{-1}\mathbf{X}^T\mathbb{E}[\mathbf{Y}] = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_{pp}\right)^{-1}\left(\mathbf{X}^T\mathbf{X}\right)\beta$$

$$\text{Var}(\hat{\beta}^{\text{Ridge}}) = \mathbb{E}\{[\beta - \mathbb{E}(\beta)][\beta - \mathbb{E}(\beta)]^T\}$$

$$\left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_{pp}\right)^{-1} := \mathbf{A}$$

$$\begin{aligned}
&= \mathbb{E}\left\{\left[\mathbf{A}\mathbf{X}^T\mathbf{y} - \mathbf{A}\left(\mathbf{X}^T\mathbf{X}\right)\beta\right]\left[\mathbf{A}\mathbf{X}^T\mathbf{y} - \mathbf{A}\left(\mathbf{X}^T\mathbf{X}\right)\beta\right]^T\right\} \\
&= \mathbb{E}\left\{\mathbf{A}\mathbf{X}^T\,\mathbf{y}\,\mathbf{y}^T\,\mathbf{X}\,\mathbf{A} - \mathbf{A}\left(\mathbf{X}^T\mathbf{X}\right)\beta\,\beta^T\left(\mathbf{X}^T\mathbf{X}\right)\mathbf{A}\right\} \\
&= \mathbf{A}\mathbf{X}^T\,\mathbb{E}\{\mathbf{y}\,\mathbf{y}^T\}\,\mathbf{X}\,\mathbf{A} - \mathbf{A}\left(\mathbf{X}^T\mathbf{X}\right)\beta\,\beta^T\left(\mathbf{X}^T\mathbf{X}\right)\mathbf{A} \\
&= \mathbf{A}\mathbf{X}^T\left(\mathbf{X}\,\beta\,\beta^T\,\mathbf{X}^T + \sigma^2\mathbf{I}\right)\mathbf{X}\,\mathbf{A} - \mathbf{A}\left(\mathbf{X}^T\mathbf{X}\right)\beta\,\beta^T\left(\mathbf{X}^T\mathbf{X}\right)\mathbf{A} \\
&= \boldsymbol{A}\boldsymbol{X}^T\left(\mathbf{X}\,\boldsymbol{\beta}\,\boldsymbol{\beta}^T\,\mathbf{X}^T + \sigma^2\mathbf{I}\right)\mathbf{X}\,\boldsymbol{A} - \boldsymbol{A}\left(\mathbf{X}^T\mathbf{X}\right)\boldsymbol{\beta}\,\boldsymbol{\beta}^T\left(\mathbf{X}^T\mathbf{X}\right)\boldsymbol{A} \\
&= \boldsymbol{A}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta}\boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{A} + \sigma^2\boldsymbol{A}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{A} - \boldsymbol{A}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta}\boldsymbol{\beta}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{A} \\
&= \sigma^2\boldsymbol{A}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{A} \\
&= \sigma^2[\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}]^{-1}\mathbf{X}^T\mathbf{X}\{[\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}]^{-1}\}^T
\end{aligned}$$

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, Felix Cameren Heyerdahl, and Eirik Bjørnson Jahr. MachineLearningProjects. `https://github.com/bragewiseth/MachineLearningProjects`, September 2023.