FYS-STK4155 Machine Learning Project 1

Qian Jianheng Oscar[1]

[1]University of Oslo

October 7, 2024

Abstract

Purpose - The goal of this project is to evaluate and compare different regression techniques, including Ordinary Least Squares (OLS), Ridge regression, and Lasso regression. Additionally, the project explores the impact of resampling methods, such as bootstrapping and K-fold cross-validation, on these regression techniques. The first part of the analysis involves using a two-dimensional Franke's function to simulate data, while the second part applies the same methods to real-world terrain data.

Findings - The analysis revealed that among the regression techniques tested, OLS using a polynomial of degree 7 provided the best fit for the data generated with Franke's function.

1 Introduction

This paper is prepared for Project 1 in the course FYS-STK4155 at the University of Oslo. The primary objective of this study is to explore regression techniques in Machine Learning (ML).

Machine Learning techniques play a crucial role in numerous aspects of modern society. In this project, we focus on analyzing and comparing three fundamental regression methods:
1. Ordinary Least Squares (OLS)
2. Ridge Regression
3. Lasso Regression

The main objective of this study is to learn and evaluate the performance of various models at different polynomial degrees.

2 Method

2.1 Franke's Function

We will first study how to fit polynomials to a specific two-dimensional function called Franke's function. This is a function which has been widely used when testing various interpolation and fitting algorithms. Furthermore, after having established the model and the method, we will employ resampling techniques such as cross-validation and bootstrap in order to perform proper assessment of our models. We will also study in detail the so-called Bias-Variance trade off.

The Franke function is given by the following equation:

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

$$+ \frac{1}{2}\exp\left(-\frac{(9x-7)^2}{4} - \frac{(9y-3)^2}{4}\right) - \frac{1}{5}\exp\left(-(9x-4)^2 - (9y-7)^2\right). \tag{2}$$

The function will be defined for $x, y \in [0,1]$. In a sense, our data are thus scaled to a particular domain for the input values.

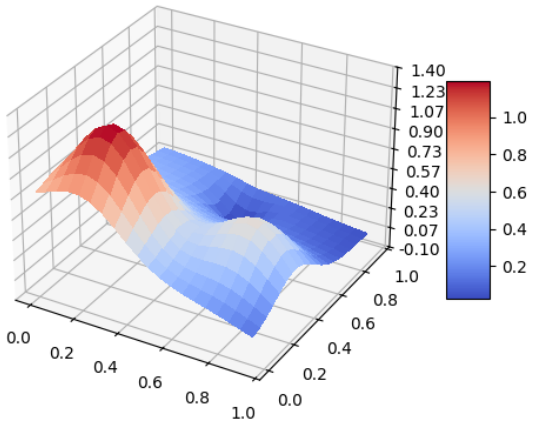

Figure 1: Visual representation of Franke's function

2.2 Evaluation Metrics

In this project, we will be using 2 main evaluation metrics - Mean Squared Error (MSE) and the R2 Score.

MSE is given by:

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

and, the R2 score is given by:

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

where we have defined the mean value of y as:

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

3 Main Section

Our first step will be to perform an OLS regression analysis of this function, trying out a polynomial fit with an x and a y dependence of the form $[x, y, x^2, y^2, xy, \dots]$. We will also include bootstrap first as a resampling technique. After that we will include the cross-validation techniques.

Most code for regression techniques are written by myself, and is benchmarked against Scikit-learn's functionality.

3.1 OLS on Franke's Function

We start of with the simplest OLS regression method which is about optimising beta given by:

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

I first observed the MSE, R2 score and beta values of OLS at different polynomial degrees. The result can be seen in Figure 2.

From Figure 2. we can see that the data is best fitted by OLS at polynomial degree of 7.

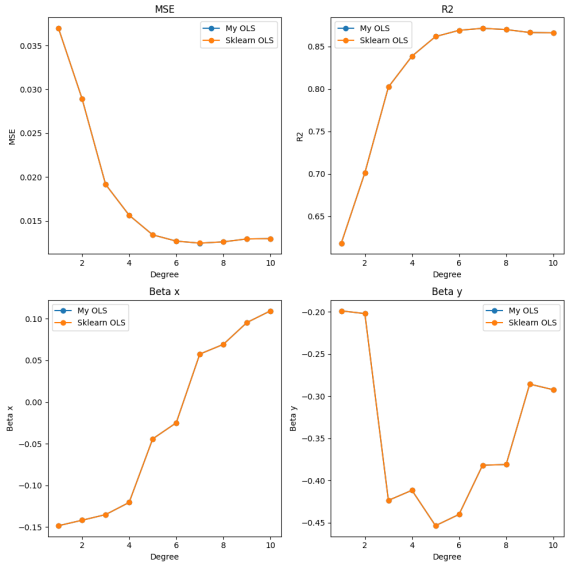

Figure 2: MSE, R2 score and beta values for different polynomial degrees fitted using OLS

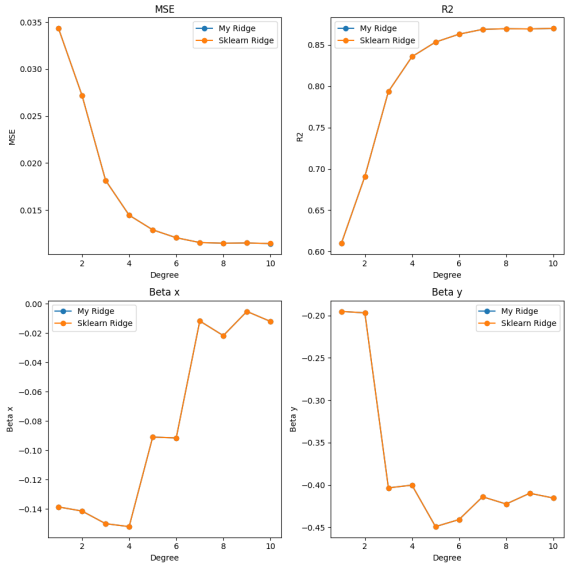

Figure 3: MSE, R2 score and beta values for different polynomial degrees fitted using Ridge regression

3.2 Ridge Regression on Franke's Function

Next, for ridge regression, it is about optimising the beta given by:

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

Then, I did my analysis on Ridge regression at different polynomial degrees. The result can be seen in Figure 3.

From Figure 3, we can see that the data is best fitted at polynomial degree of 10, however, I stopped at polynomial degree of 10 due to performance issue.

I also did an analysis on the effect of the regularisation term, alpha on the MSE and R2 score. From Figure 4, we can see that using alpha=0.375 is optimal.

3.3 Lasso Regression on Franke's Function

Lastly, for Lasso regression, it is optimising the beta given by:

$$\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta} + \lambda sgn(\boldsymbol{\beta}) = 2\boldsymbol{X}^T\boldsymbol{y}.$$

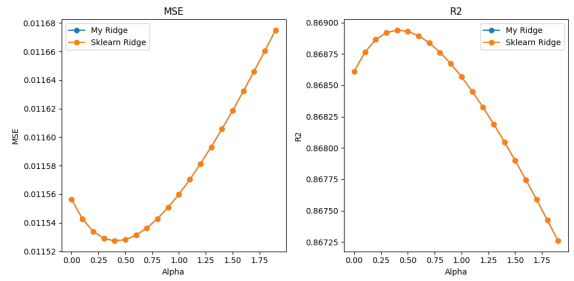

Figure 4: MSE and R2 score with varying alphas using Ridge regression

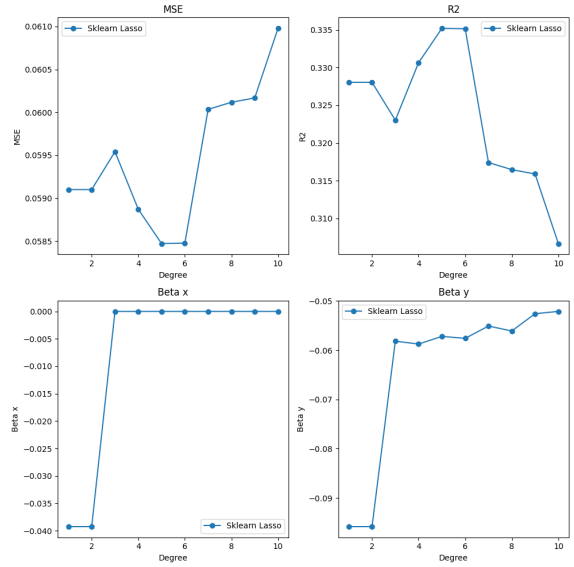

Figure 5: MSE, R2 score and beta values for different polynomial degrees fitted using Lasso regression

I did my analysis on Lasso regression at differenrt polynomial degrees, and the results can be seen in Figure 5. We can see from Figure 5 that the best polynomial degree for Lasso regression is 5 or 6.

Then, I also analysed the effect of values of alpha on the MSE and R2 score for Lasso regression. The result can be seen in Figure 6.

3.4 Analytical Analysis

The expectation value of $\boldsymbol{y}$ is

$$\mathbb{E}[\boldsymbol{y}] = \mathbb{E}[\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}] = \boldsymbol{X}\boldsymbol{\beta} + \mathbb{E}[\boldsymbol{\varepsilon}] = \boldsymbol{X}\boldsymbol{\beta}.$$

We can do the above simplification since $\boldsymbol{\sigma}$ follows $\mathcal{N}(0, \sigma^2)$, we have $\mathbb{E}(\boldsymbol{\varepsilon}) = 0$ and because $f = \boldsymbol{X}\boldsymbol{\beta}$ is stochastic.

The expectation value of $\boldsymbol{y}$ for a given element i is

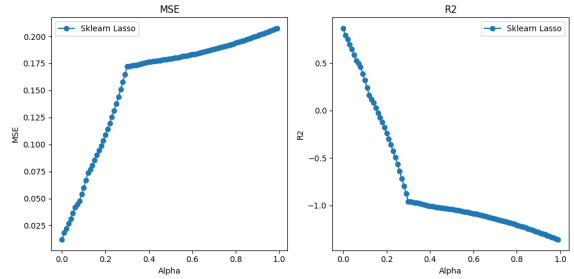

Figure 6: MSE and R2 score with varying alphas using Ridge regression

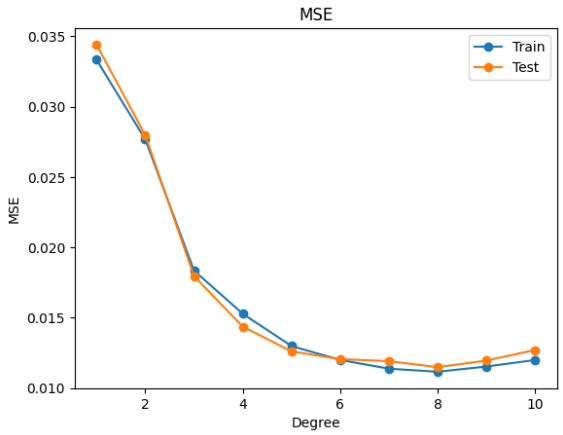

Figure 7: Comparing MSE at varying polynomial degrees between train and test set

$$\mathbb{E}[y_i] = \sum_j x_{ij}\beta_j = \mathbf{X}_{i,*}\boldsymbol{\beta}.$$

The variance of $\boldsymbol{y_i}$ is

$$Var(\boldsymbol{y_i}) = \mathbb{E}[y_i^2] - \mathbb{E}[y_i]^2$$

$$= \mathbb{E}[(\boldsymbol{X}_{i,*}\boldsymbol{\beta} + \varepsilon_i)^2] - (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2$$
$$= \mathbb{E}[(\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 + 2\boldsymbol{X}_{i,*}\boldsymbol{\beta}\varepsilon_i + \varepsilon_i^2] - (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2$$
$$= (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 + 2\boldsymbol{X}_{i,*}\boldsymbol{\beta}\mathbb{E}[\varepsilon_i] + \mathbb{E}[\varepsilon_i^2] - (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2$$

since $\mathbb{E}[\varepsilon_i] = 0$ we have

$$= (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2 + \mathbb{E}[\varepsilon_i^2] - (\boldsymbol{X}_{i,*}\boldsymbol{\beta})^2$$

$$= \mathbb{E}[\varepsilon_i^2] = Var(\varepsilon_i) = \sigma^2.$$

For the expectation value of $\hat{\boldsymbol{\beta}}$, since $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$, we have

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

For variance of $\hat{\boldsymbol{\beta}}$ we have

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

since $\mathbb{E}[\boldsymbol{Y}\boldsymbol{Y}^T] = \mathbb{E}[\boldsymbol{Y}]\mathbb{E}[\boldsymbol{Y}^T] + Cov(\boldsymbol{Y}) = \boldsymbol{X}\boldsymbol{\beta}\boldsymbol{\beta}^{\boldsymbol{T}}\boldsymbol{X}^{\boldsymbol{T}} + \sigma^2\boldsymbol{I}_{\boldsymbol{nn}}$ we have

$$= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{\beta}\boldsymbol{\beta}^{\boldsymbol{T}}\boldsymbol{X}^{\boldsymbol{T}} + \sigma^2)\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} - \boldsymbol{\beta}\boldsymbol{\beta}^T$$
$$= (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{\beta}\boldsymbol{\beta}^{\boldsymbol{T}}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} + \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1} - \boldsymbol{\beta}\boldsymbol{\beta}^T$$
$$= \boldsymbol{\beta}\boldsymbol{\beta}^{\boldsymbol{T}} + \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1} - \boldsymbol{\beta}\boldsymbol{\beta}^{\boldsymbol{T}} = \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}.$$

3.5 Bias-Variance Trade-Off with Bootstrap

Before I perform an analysis of the bias-variance trade-off on the test data, I made first a figure similar to Fig. 2.11 of Hastie, Tibshirani, and Friedman. Figure 2.11 of this reference displays only the test and training MSEs. The test MSE can be used to indicate possible regions of low/high bias and variance. The result can be seen in Figure 7.

Next, I analysed mathematically the bias-variance trade-off. Assuming a function is given by:

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

Here ϵ is normally distributed with mean zero and standard deviation σ^2.

In our derivation of the ordinary least squares method we defined then an approximation to the function f in terms of the parameters $\boldsymbol{\beta}$ and the design matrix $\boldsymbol{X}$ which embody our model, that is $\tilde{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{\beta}$.

The parameters $\boldsymbol{\beta}$ are in turn found by optimizing the mean squared error via the so-called cost function

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

Here the expected value $\mathbb{E}$ is the sample value.

We can rewrite this as

$$\begin{aligned}
\mathbb{E}\left[(\boldsymbol{y} - \tilde{\boldsymbol{y}})^2\right] &= \mathbb{E}[\boldsymbol{y}^2 - 2\boldsymbol{y}\tilde{\boldsymbol{y}} + \tilde{\boldsymbol{y}}^{\boldsymbol{2}}] && (3)\\
&= \mathbb{E}[\boldsymbol{y}^2] - 2\mathbb{E}[\boldsymbol{y}\tilde{\boldsymbol{y}}] + \mathbb{E}[\tilde{\boldsymbol{y}}^{\boldsymbol{2}}] && (4)\\
& && (5)
\end{aligned}$$

We look at the first term $\mathbb{E}[\boldsymbol{y}^2]$, which can be written as

$$\begin{aligned}
\mathbb{E}[\boldsymbol{y}^2] &= \mathbb{E}[(f(\boldsymbol{x}) + \boldsymbol{\epsilon})^2] && (6)\\
&= \mathbb{E}[f^2 + 2f\boldsymbol{\epsilon} + \boldsymbol{\epsilon}^2] && (7)\\
&= \mathbb{E}[f^2] + 2\mathbb{E}[f \cdot \boldsymbol{\epsilon}] + \mathbb{E}[\boldsymbol{\epsilon}^2] && (8)\\
& && (9)
\end{aligned}$$

Since f is deterministic, we have $\mathbb{E}[f^2] = f^2$ and $\mathbb{E}[f \cdot \boldsymbol{\epsilon}] = f\mathbb{E}[\boldsymbol{\epsilon}] = 0$. Thus, we have

$$\mathbb{E}[\boldsymbol{y}^2] = f^2 + \mathbb{E}[\boldsymbol{\epsilon}^2] \tag{10}$$

Since $\boldsymbol{\epsilon} \sim N(0, \sigma^2)$, we have $\mathbb{E}[\boldsymbol{\epsilon}^2] = Var[\boldsymbol{\epsilon}] - \mathbb{E}[\boldsymbol{\epsilon}]^2 = \sigma^2 - 0$, so

$$\mathbb{E}[\boldsymbol{y}^2] = f^2 + \sigma^2 \tag{11}$$

Now we look at the second term $2\mathbb{E}[\boldsymbol{y}\tilde{\boldsymbol{y}}]$, which can be written as

$$\begin{aligned}
2\mathbb{E}[\boldsymbol{y}\tilde{\boldsymbol{y}}] &= 2\mathbb{E}[(f + \boldsymbol{\epsilon})\tilde{\boldsymbol{y}}] && (12)\\
&= 2\mathbb{E}[f\tilde{\boldsymbol{y}} + \boldsymbol{\epsilon}\tilde{\boldsymbol{y}}] && (13)\\
&= 2\mathbb{E}[f\tilde{\boldsymbol{y}}] + 2\mathbb{E}[\boldsymbol{\epsilon}\tilde{\boldsymbol{y}}] && (14)\\
& && (15)
\end{aligned}$$

Since ϵ is independent of $\tilde{\boldsymbol{y}}$, we have $\mathbb{E}[\boldsymbol{\epsilon}\tilde{\boldsymbol{y}}] = \mathbb{E}[\boldsymbol{\epsilon}]\mathbb{E}[\tilde{\boldsymbol{y}}] = 0$. Thus, we have

$$\begin{aligned}
2\mathbb{E}[\boldsymbol{y}\tilde{\boldsymbol{y}}] &= 2\mathbb{E}[f\tilde{\boldsymbol{y}}] && (16)\\
&= 2f\mathbb{E}[\tilde{\boldsymbol{y}}] && (17)\\
& && (18)
\end{aligned}$$

Finally, we look at the third term $\mathbb{E}[\tilde{\boldsymbol{y}}^{\boldsymbol{2}}]$, which can be written as

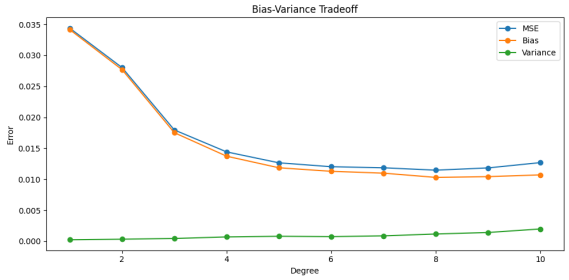

Figure 8: Bias-Variance trade-off analysis

$$\mathbb{E}[\tilde{\boldsymbol{y}}^{\boldsymbol{2}}] = Var[\tilde{\boldsymbol{y}}] + \mathbb{E}[\tilde{\boldsymbol{y}}]^2 \tag{19}$$

Putting everything together, we have

$$\begin{aligned}
\mathbb{E}\left[(\boldsymbol{y}-\tilde{\boldsymbol{y}})^2\right] &= \mathbb{E}[\boldsymbol{y}^2] - 2\mathbb{E}[\boldsymbol{y}\tilde{\boldsymbol{y}}] + \mathbb{E}[\tilde{\boldsymbol{y}}^{\boldsymbol{2}}] \\
&= \overbrace{f^2 - 2f\mathbb{E}[\tilde{\boldsymbol{y}}] + \mathbb{E}[\tilde{\boldsymbol{y}}]^2}^{Bias} + Var[\tilde{\boldsymbol{y}}] + \sigma^2 \\
&= \overbrace{\mathbb{E}\left[(\boldsymbol{f}-\mathbb{E}[\tilde{\boldsymbol{y}}])^2\right]}^{Bias} + \overbrace{\mathbb{E}\left[(\tilde{\boldsymbol{y}}-\mathbb{E}\left[\tilde{\boldsymbol{y}}\right])^2\right]}^{Var} + \sigma^2 \\
&= \overbrace{\frac{1}{n}\sum_i (y_i - \mathbb{E}[\tilde{\boldsymbol{y}}])^2}^{Bias} + \overbrace{\frac{1}{n}\sum_i (\tilde{y}_i - \mathbb{E}[\tilde{\boldsymbol{y}}])^2}^{Var} + \sigma^2 \\
&= \text{Bias}[\tilde{y}] + \text{Var}[\tilde{y}] + \sigma^2
\end{aligned}$$

We can see that the first term represents the square of the bias of the learning method, which is the error caused by simplifying the assumptions built into the method. The second term represents the variance of the chosen model. The last term represents the variance of the noise in the data.

Having done this analysis, I we can expect the MSE to be always bigger than the sum of bias and variance. This can be seen in Figure 8 where I tested this theory on the actually data.

From Figure 8, we can see that the error is always bigger than the sum of variance and bias. This is inline with our above analysis where $\mathbb{E}\left[(\boldsymbol{y}-\tilde{\boldsymbol{y}})^2\right] = \text{Bias}[\tilde{y}] + \text{var}[\tilde{y}] + \sigma^2$. We can see that there is always a trade off between bias and variance. As the model complexity increases, the bias decreases while the variance increases. This is because a more complex model can fit the data better, but it is also more sensitive to noise. The number of data points also affects the bias-variance trade off. With more data points, the bias decreases and the variance increases. This is because with more data points, the model can better capture the underlying structure of the data, but it is also more sensitive to noise. So the goal is to find the model complexity that minimizes the error, which is the sum of bias and variance.

3.6 K-fold Cross Validation

Next, I implemented the k-fold cross-validation technique and compared the MSE with the bootstrap method.

We can see the result in Figures 9, 10, and 11 for the comparison of MSE using Bootstrap and K-fold cross-validation for OLS, Ridge, and Lasso regression respectively.

3.7 Analysis on Real Data

With our codes functioning and having been tested properly on a simpler function we are now ready to look at real data. We will essentially repeat in this exercise what was done in exercises a-f. However, we need first to download the data and prepare properly the inputs to our codes. We are going to download digital terrain data from the website ¡https://earthexplorer.usgs.gov/¿.

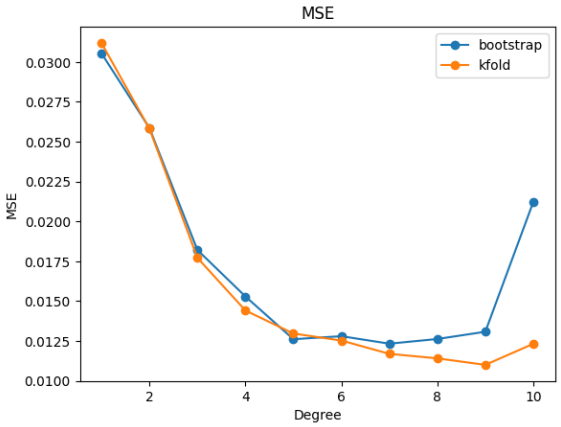

Figure 9: MSE of Bootstrap versus K-fold using OLS

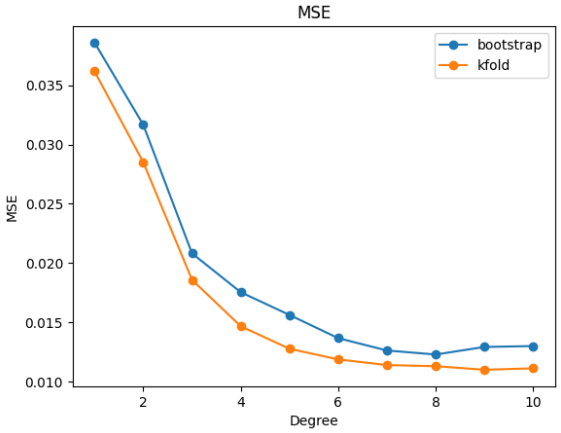

Figure 10: MSE of Bootstrap versus K-fold using Ridge regression

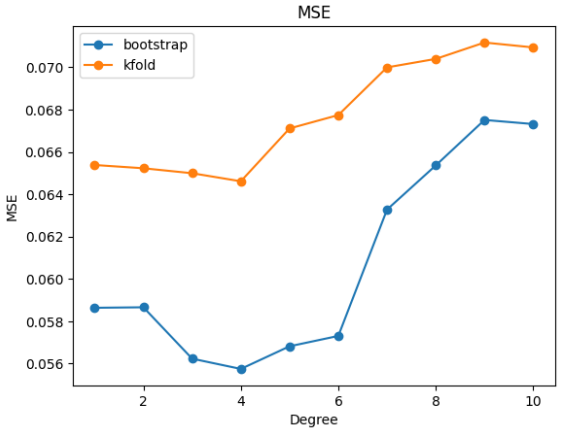

Figure 11: MSE of Boostrap versus K-fold using Lasso regression

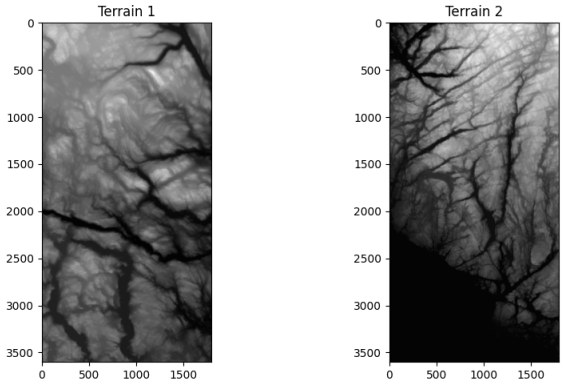

Figure 12: Visualisation of terrain data

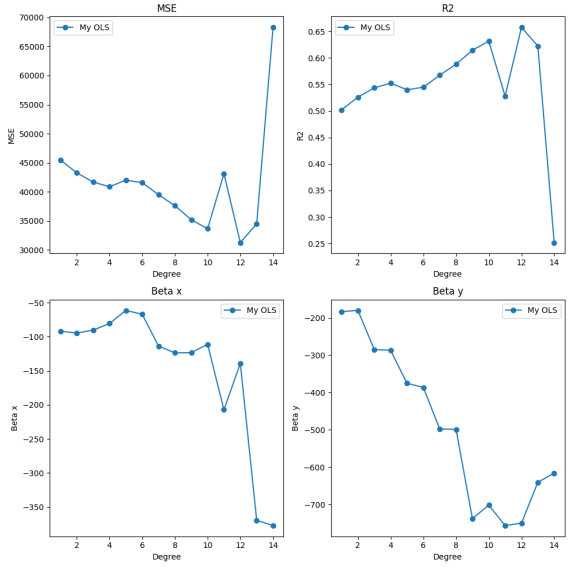

Figure 13: MSE, R2 and Betas on terrain data using OLS

A visualisation of the dataset downloaded can be seen in Figure 12.

We will then perform the same analysis done on Franke's function on this new terrain data.

3.8 OLS on Terrain Data

From Figure 13, we can see that the best fit occur at polynomial degree equals to 12.

3.9 Ridge regression on Terrain Data

From Figure 14, we can see that the best fit occurs at polynomial degree equal to 11.

3.10 Lasso regression on Terrain Data

From Figure 15, we can see that the best fit occurs at polynomial degree equal to 13.

3.11 Bias-variance Trade-off on Terrain Data

Next, I performed bias-variance trade-off on the terrain data.

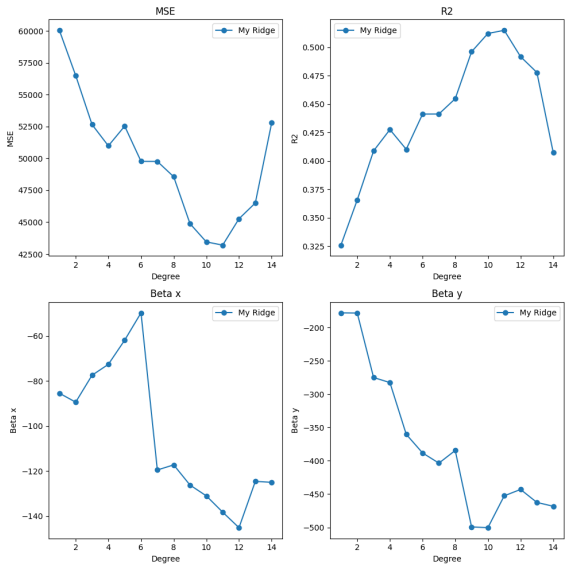

Figure 14: MSE, R2 and Betas on terrain data using Ridge regression

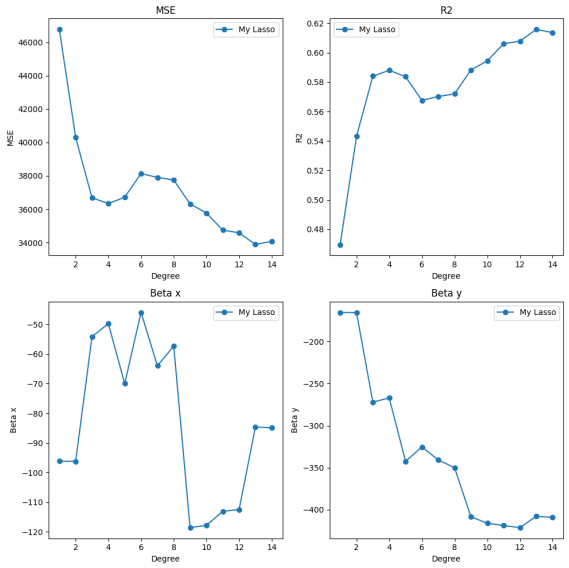

Figure 15: MSE, R2 and Betas on terrain data using Lasso regression

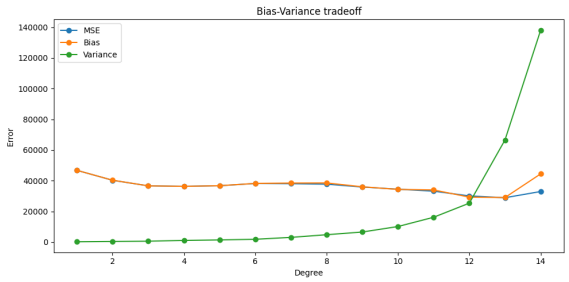

Figure 16: Bias-variance trade-off on terrain data

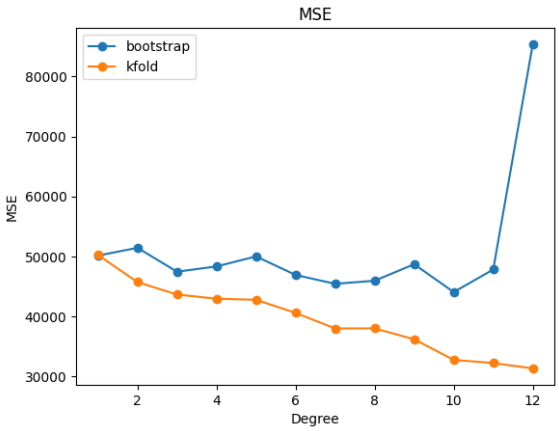

Figure 17: Comparing MSE for Bootstrap versus K-fold cross-validation using

3.12 K-fold Cross Validation on Terrain Data

From Figure 17, we can see that the model performs much better using k-fold cross-validation as compared to using bootstrap.

4 Results

From the results, we can see that OLS regression performs the best on Franke's function as polynomial degree equal to 7, and the OLS regression performs the best on Terrain data at polynomial degree equal to 12. In both case, using the K-fold cross-validation resampling method indeed improved the performance of the model.

5 Conclusion

In conclusion, I learnt 3 method of regression method for machine learning, and resampling methods such as bootstrap and k-fold cross-validation do have a significant effect on the performance of the title.

References

developers, scikit-learn (2024). *sklearn.utils.resample — scikit-learn 1.3.0 documentation*, Accessed: 2024-10-04. **available at**: `https://scikit-learn.org/stable/modules/generated/sklearn.utils.resample.html`.

Fellows, Richard F. and Liu, Anita M. M. (2021). *Research Methods for Construction*, 5th ed. Wiley-Blackwell, p. 384. ISBN: 978-1-119-81473-3.

Mehta, Pankaj *et al.*, (2019). "A high-bias, low-variance introduction to Machine Learning for physicists", *Physics Reports*, Vol. 810. A high-bias, low-variance introduction to Machine Learning for physicists, pp. 1–124. ISSN: 0370-1573. DOI: `https://doi.org/10.1016/j.physrep.2019.03.001`. **available at**: `https://www.sciencedirect.com/science/article/pii/S0370157319300766`.

OpenAI (2024). *ChatGPT*, `https://chat.openai.com/`. Accessed: 2024-10-04.

Pedregosa, Fabian *et al.*, (2011). "Scikit-learn: Machine Learning in Python", *Journal of Machine Learning Research*, Vol. 12, pp. 2825–2830. **available at**: `https://jmlr.org/papers/v12/pedregosa11a.html`.

Ranstam, J and Cook, J A (2018). "LASSO regression", *British Journal of Surgery*, Vol. 105 No. 10, pp. 1348–1348. ISSN: 0007-1323. DOI: `10.1002/bjs.10895`. eprint: `https://academic.oup.com/bjs/article-pdf/105/10/1348/36206240/bjs10895.pdf`. **available at**: `https://doi.org/10.1002/bjs.10895`.