Draft version October 9, 2019
Preprint typeset using LaTeX style emulateapj v. 12/16/11

REGRESSION ANALYSIS AND RESAMPLING TECHNIQUES: THE FRANKE FUNCTION AND REAL TERRAIN DATA FROM MARS

Anna Eliassen [1,2]
In collaboration with Kristina Othelia Lunde Olsen

Draft version October 9, 2019

ABSTRACT

In this project we have studied different regression methods and a resampling technique, with the goal of creating a model that can predict a given dataset. We have used Ordinary Least Squares (OLS), Ridge regression and Lasso regression, together with the resampling technique k-fold cross-validation. We have studied both the well-known Franke function, as well as a real terrain dataset of an unnamed crater on Mars. We found that OLS and Ridge provided the best prediction for both the Franke function and the real terrain data, and it was hard to distinguish between these two. For the Franke function the MSE for both methods was of the order 10^{-3}, while MSE for lasso was in the order of 10^{-2}. On the terrain data we got an MSE of 753 for OLS and 730 for Ridge, while MSE with Lasso regression resulted in a MSE of 1200.

Subject headings: Terrain data — physics: data analysis — methods: numerical, statistical

1. INTRODUCTION

In this project we will use different regression methods and a resampling technique on a function named The Franke function, and later we will look at real terrain data of Mars. With the linear regression analysis, we want to predict the behavior and properties of the dataset we are looking at. In other words, we want to create a model that fits our set of datapoints.

The regression methods discussed in this project is the Ordinary Least Squares (OLS), Ridge regression and Lasso regression. We will also include K-fold cross validation as a resampling technique on these three regression methods. Finding out which model fits the data best, we will look at different characteristics of the models, and evaluate things such as the mean squared error (MSE), R2 score function and the confidence interval of the β-parameters.

The Franke function is a two-dimensional function that is widely used in these kinds of analysis. Using this function, we can first test how our methods and algorithms behave, before moving on to the analysis of unknown data. A three-dimensional plot of The Franke function kan be ssen in figure 1, and gives an idea of how we want our model to look like.

Since astronomy lies close to our heart, we decided to use a terrain data of Mars, taken by NASA [2]. The section of the image we chose, shows an unamed crater on the Mars surface, which can be seen in APPENDIX D. A three-dimensional surface plot of the crater, which we will try to recreate with our model, can be seen in figure 2.

We will first give a description of the theory and methods used, before moving on to the results. Following is a discussion of the results and the main conclusion. At the end we have Appendix A-E, and at last references.

[1] annaeli@student.matnat.uio.no
[2] Department of Physics, University of Oslo, P.O. Box 1048 Blindern, N-0316 Oslo, Norway

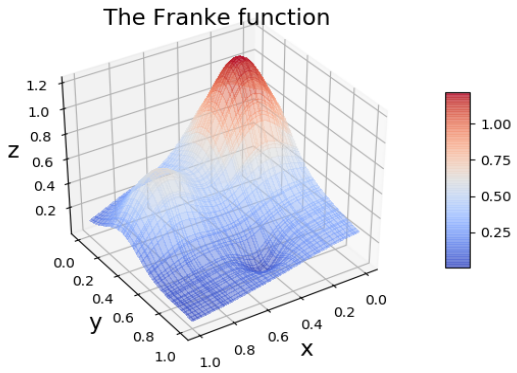

Fig. 1.— A 3D surface plot of The Franke function.

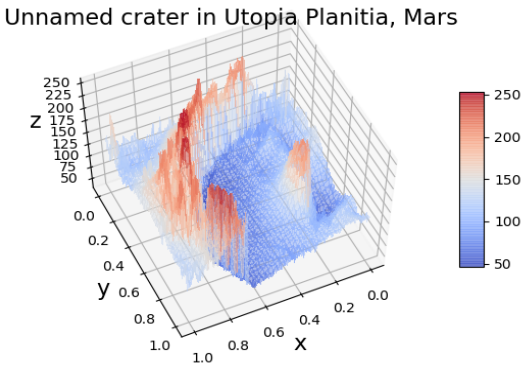

Fig. 2.— A 3D surface plot of an unnamed crater on Mars.

2. THEORY

The two-dimensional Franke function is given by

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

A regression model is linear when all the terms in the equaiton is constant or variable multiplied with an independent variable. So a linear regression model can be described as

$$\tilde{y} = X\beta,$$

where X is the design matrix and β is the regression parameters. Linear regression provides a set of analytical equations for the β parameters.

The βs for OLS regression can be written as

$$\beta = (X^T \cdot X)^{-1} \cdot X^T \cdot y,$$

where X is the design matrix and y is the data values.
The mean squared error (MSE), gives us the mean squared difference between the actual value and the estimated or predicted value. It should be as close to zero as possible for a model to be a well fitted to the data. The MSE can be written as a sum

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

where $\hat{y}$ represents the actual data and $\hat{\tilde{y}}$ is the estimated or predicted values. If $\hat{\tilde{y}}$ is the predicted value, then the MSE indicates the quality of the prediction.

The R^2-score function can be described as

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

where $\hat{\tilde{y}}$ is the predicted value, and the mean of the actual values defined as

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

The MSE can also be written as a sum of the bias and variance

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

3. METHOD

To model the curvature of the Franke function and the real terrain data, we use linear regression methods that includes non-linear variables, in this case polynomials.
As mentioned above, we started our analysis on the Franke function. The first we did was to define x and y as $x, y \in [0,1]$ with (100x100) data points, which were used to calculate the Franke function. We then added a normally distributed noise, N(0,1), to the function. Using OLS and polynomials in x and y up to fifth order, we calculated the confidence intervals of the parameters β, the MSE and the R2 score.

Then we added more complexity to our model, adding a resampling technique. We chose to use k-fold cross-validation, with $k = 5$. Splitting the data into a training set and a test set, we now calculated the MSE and R2 resulting from the test data. We also included a plot of the MSE as a function of the model complexity for the training set and the test set.

To look at the bias-variance tradeoff, we now calculated the bias and variance for each value of k. To calculate the bias we used the test data and the prediction for each k, and returned a mean value of the bias from our k-fold cross-validation function. In the same way we calculated the variance of the prediction for each k, and returned the mean value.

After this analysis using the OLS regression method, we moved on to Ridge regression. We chose a range of lambda values, and plotted the MSE as a function of lambda, to evaluate which lambda that would result in the best fitted model. After finding the best lambda, we created a final model for the Ridge Regression, and computed the MSE and R2 and plotted the confidence interval of the β parameters. We also wrote a pytest checking that $\lambda = 0$ gave the same result for MSE as the OLS regression.

For the Lasso regression, we followed the same procedure as with Ridge regression. However, here we used the Scikit built-in functions for fitting the model and making the predictions.

After our regression analysis of the Franke function, we moved on to look at real terrain data from Mars. We first cropped the image to get a smaller size to work with, only including the region we wanted to make a model of. Then we resized this region into a (200x200) matrix, using a python built-in function with nearest neighbor interpolation. We then repeated the regression analysis done on the Franke function, using the same regression methods and k-fold cross-validation.

The python code used to do the regression analysis can be found in our github repository [4], and are named
project1.py
project1_func.py
project1_plot.py
project1_test.py

4. RESULTS

4.1. *Franke function*

A plot showing the mean squared error of the test and training data as a function of model complexity is shown below in figure 3. In Appendix A a simular plot can be seen for Ridge and Lasso regression, in figure 17 and 16 respectively. Appendix A also includes a plot where MSE for train and test data are plotted together for all the three model, see figure 18.

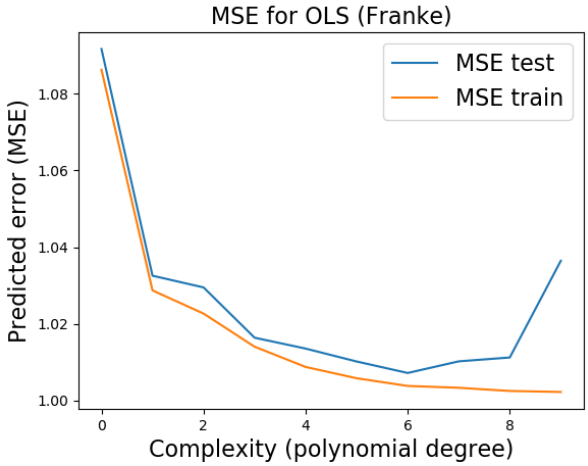

FIG. 3.— MSE of the test data as a function of model complexity, OLS regression on the Franke function

The 95 % confidence interval of the parameters β is illustrated by figure 4. The same plot are also made for Ridge (figure 23) and Lasso regression (figure 24) , which can be found in Appendix B.

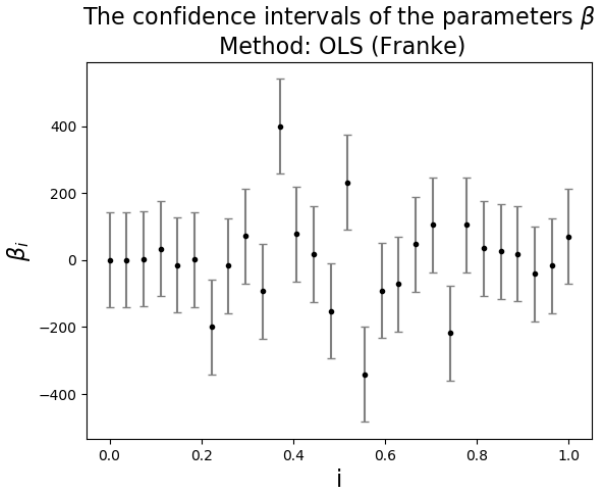

FIG. 4.— The confidence intervals of the β parameters

For Ridge regression, the MSE resulting from the test data as a function of different λ-values, can be seen in figure 5.

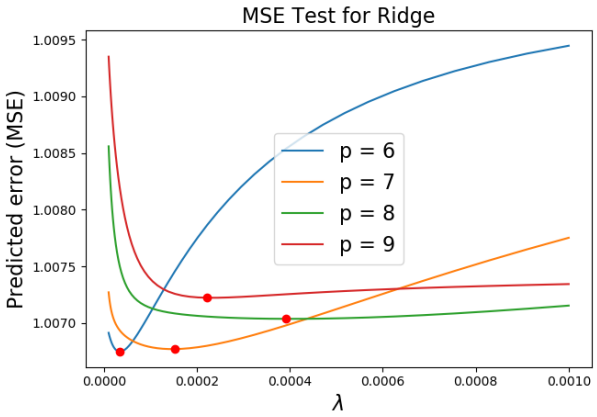

FIG. 5.— The MSE of the test data as a function of lambda values

For Lasso regression, the MSE resulting from the test data as a function of different λ-values, can be seen in figure 6. A figure zooming in on the lowest MSE and corresponding lambda values may be seen in figure 20.

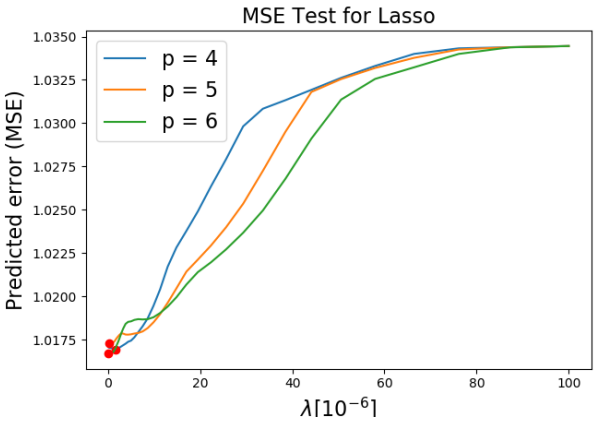

FIG. 6.— The MSE of the test data as a function of lambda values

The main results from the regression analysis on the Franke function are summarized in table 1. Here we have calculated the MSE and R2 between the true Franke function (without noise) and the predicted value. The 'p' is the polynomial degree, so for OLS we found that polynomials up to 6th order gave the best model, and similar we found that 7 and 4 where best for Ridge and Lasso respectively.

	OLS	Ridge	Lasso
p	6	7	4
MSE	0.00212	0.00147	0.01044
R2	0.97	0.98	0.87
λ	–	0.00148497	$1.67683 \cdot 10^{-6}$

TABLE 1
Main results from the regression analysis on the Franke function

The final model of the Franke function with OLS regression and k-fold cross-validation is illustrated in figure 7. The actual Franke function (without noise) are plottet as a black scatter plot, to illustrate the difference between the true function and the model.

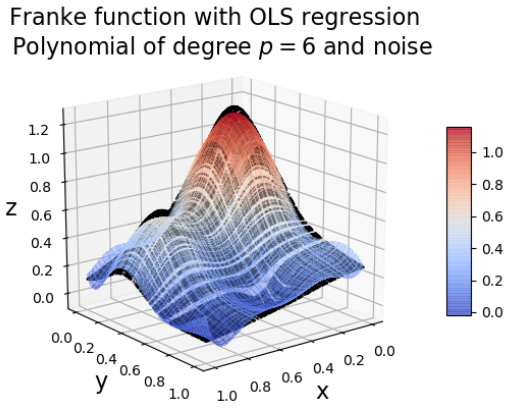

Fig. 7.— The Ordinary Least Square model of the Franke function

The final model of the Franke function with Ridge regression and k-fold cross-validation is illustrated in figure 8.

Fig. 8.— The Ridge regression model of the Franke function

The final model of the Franke function with Lasso regression and k-fold cross-validation is illustrated in figure 9.

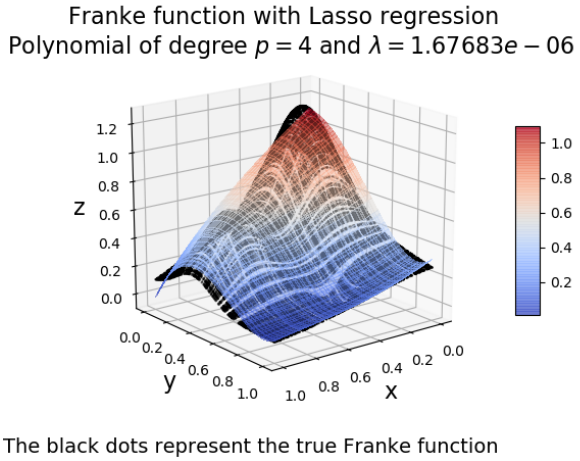

Fig. 9.— The Lasso regression model of the Franke function

4.2. *Terrain data - Mars*

Figure 10 shows the MSE as a function of model complexity for the OLS regression on the terrain data. Simular figures for Ridge and Lasso can be seen in figure 28 and 29 in Appendix A.

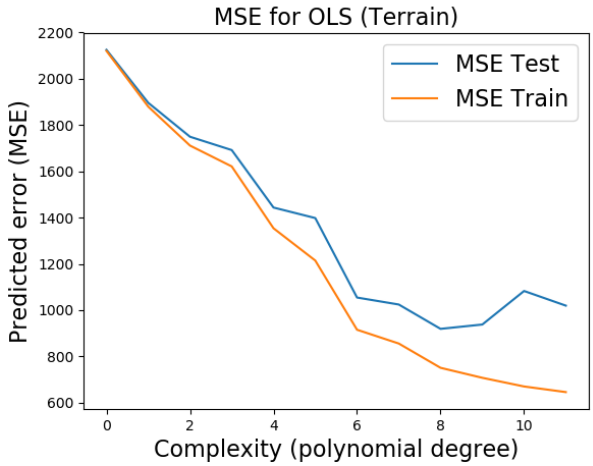

FIG. 10.— MSE as a function of model complexity for the terrain data

The confidence interval of the parameters β for all three models are given in Appendix B, figure 25, 26 and 27, for OLS, Ridge and Lasso respectively.

For Ridge regression, a plot showing the MSE resulting from the test data as a function of different λ-values can be seen in figure 11.

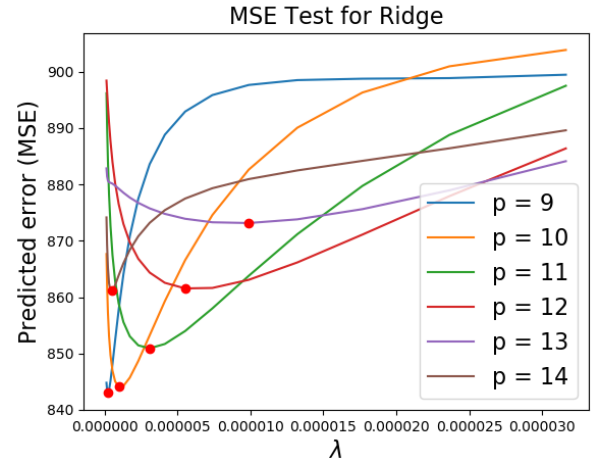

FIG. 11.— MSE on the test data as a function of λ for Ridge regression on the real terrain data

For Lasso regression, a plot illustrating the MSE resulting from the test data as a function of different λ-values can be seen in figure 12.

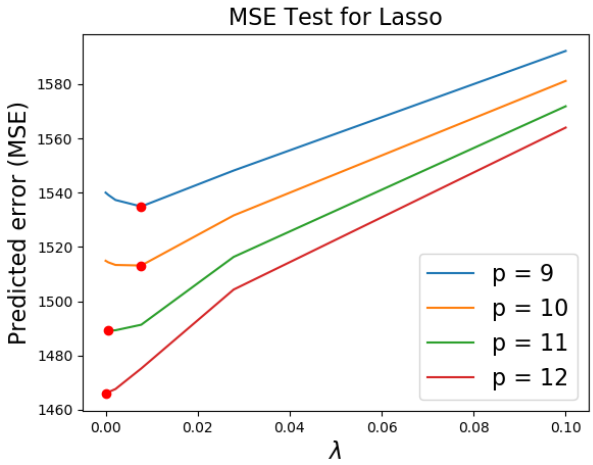

FIG. 12.— MSE on the test data as a function of λ for Lasso regression on the real terrain data

The final model of the OLS method with k-fold cross-validation for the unnamed Mars crater is represented by figure 13.

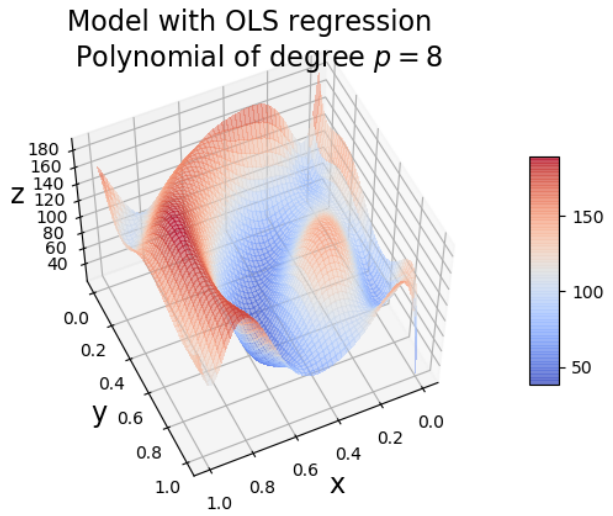

FIG. 13.— The OLS model with resampling of the unnamed Mars crater

The final model for the unnamed Mars crater using Ridge regression and k-fold cross-validation is represented by figure 14.

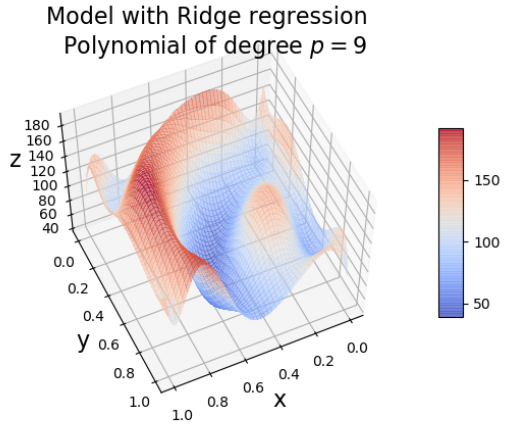

FIG. 14.— The Ridge model with resampling of the unnamed Mars crater

The final model for the unnamed Mars crater using Ridge regression and k-fold cross-validation is represented by figure 15.

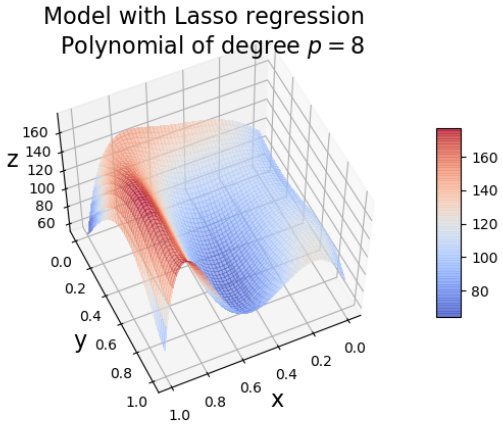

FIG. 15.— The Lasso model with resampling of the unnamed Mars crater

The main results from the regression analysis on the real terrain data are summarized in table 2.

	OLS	**Ridge**	**Lasso**
p	8	9	8
MSE	752.996	729.121	1166.15
R2	0.644904	0.656163	0.450073
λ	–	$5.389 \cdot 10^{-7}$	0.00129155

TABLE 2
MAIN RESULTS FROM THE REGRESSION ANALYSIS ON THE REAL TERRAIN DATA

5. DISCUSSION

5.1. *The Franke function*

From figure 3, we see that around a polynomial degree of 6, the MSE for the test data starts to differ more and more from the training data. Therefore, we used polynomials up to 6th order when we calculated the design matrix and our final model, resulting the MSE and R2 seen in table 1. The MSE is pretty low, and R2 is close to 1. This is what we would expect with the OLS regression, and our model seems to give good predictions for the true Franke function.

From figure 5, we see that polynomial degree 6 gives the lowest MSE for Ridge regression. However, a slightly increase of lambda results in the MSE for the test data to increase drastically. Since a polynomial degree of 7 results in nearly the same minimum MSE and does not increase that drastically, we figured this would be a better choice for our final model. As we can see from table 1, the final model for the Ridge regression results in a slightly lower MSE than for the OLS. As illustrated in figure 7 and 8, both the OLS method and Ridge method seems to predict the Franke function pretty good. However, on the edges of the function, the Ridge regression seems to yield a better prediction.

The method we had the most troubles with was the Lasso regression. This method was very sensitive to our choice of λs, and computations took a long time. After hours of trying to balance the correct choice of values and the computational time, we had to ''give up" in order to finished within the scheduled time. We therefore ended up choosing a few set of polynomials and a pretty small range of lambdas to proceed with the analysis. We ended up choosing polynomials up to 4th order as a best prediction. As seen in table 1, the MSE ended up ten times larger than for OLS and Ridge. We can also see from figure 9 that the model do not fit the Franke function as good as the two other methods.

For all three methods we hade problems with the bias-variance tradeoff, see figure 19 in Appendix A for an example. More examples can be found in our github-repository as well. The bias should be slightly lower than MSE for the test set, however this was in general not the case. We was not able to figure out exactly what went wrong with our calculation of the bias, or if it was something in our k-fold causing the MSE for the test set to be slightly higher than it should.

5.2. *Terrain data - Mars*

From figure 10 we can see the tendencies of the MSE for train and test data of increasing complexity. Based on this figure we decided to use a polynomial degree of 8 for our best model, since the test data for higher polynomials differ increasingly. In table 2, we can see that the resulting MSE is approximately 753 and the corresponding R2 is 0.64. We did actually expect a higher MSE, but we probably managed to get it this ''low" because of our way of resizing the cropped image with the nearest neighbor interpolation.

Based on figure 11, we chose polynomials up to 9th order for our best model. From the figure it may seem like higher order polynomials would be better, however plotting the 3D model with higher order polynomials resulted in a poorer prediction. From table 2 we can see that MSE is slightly lower for Ridge than for OLS, and R2 is a bit higher. Comparing figure XX and XX with the terrain surface in figure XX, it may seem like Ridge is slightly better at predicting the surface, but it is very hard to distinguish these two methods based on our regression analysis.

As for the Franke function, we had trouble finding good values to use in the scikit built-in functions to get a good prediction that did not take too long to compute. At the end, we ended up just choosing polynomials up to 8th orders, and as listed in table 2, we get a much larger MSE than for OLS and Ridge regression. In figure XX, it is also easy to see that out Lasso model did not predict the crater surface very well.

6. CONCLUSIONS

As expected, the Ordinary Least Square results in good predictions for this kind of analysis, especially on the Franke function. The method is pretty easy to implement, and if the OLS assumptions for linear regression is satisfied, this method should give the best estimates. We also found that our implementation of the Ridge regression, gave pretty similar results as OLS and was equal if $\lambda = 0$.

As for the bias-variance tradeoff analysis, using bootstrap as a resampling technique would probably be easier and should provide a better result.

Working with Lasso regression was really hard, because of sensitivity to the choice of lambda values and the computation time. Unfortunately, this stole a lot of time from other parts of the project. However, a better understanding of the scikit built-in functions for Lasso, how they work and what parameters to use in our case, would probably result in better predictions than we managed to do in this project. So here we have some room improvements for future projects.

Overall, we are satisfied with the results we managed to get at the end. We were not very familiar with linear regression methods or resampling techniques when starting this project, so this have definitely increased our knowledge regarding data analysis.

APPENDIX A

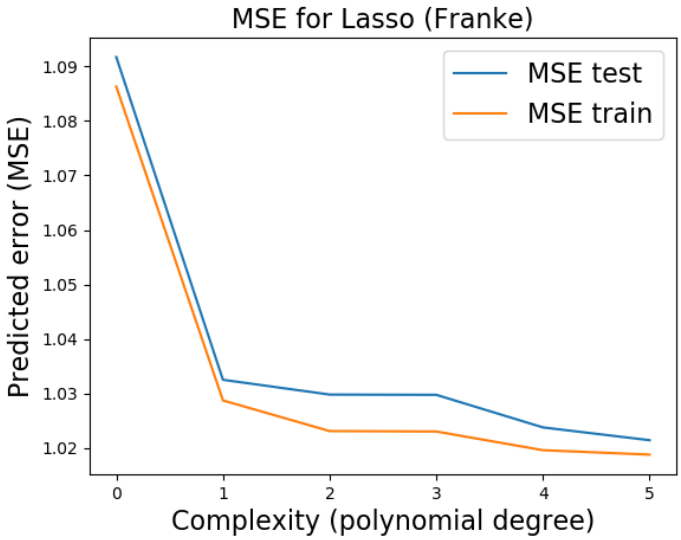

FIG. 16.— The MSE for the test set as a function of complexity, Lasso regression on the Franke function

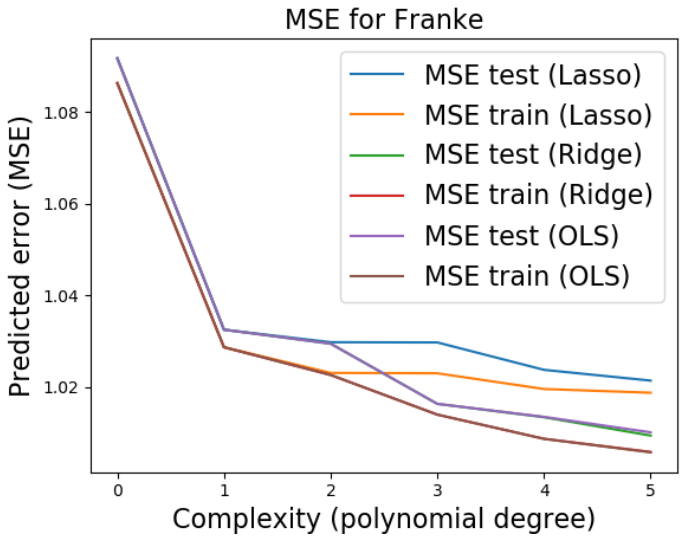

FIG. 18.— The MSE for train and test for the different models on the Franke function

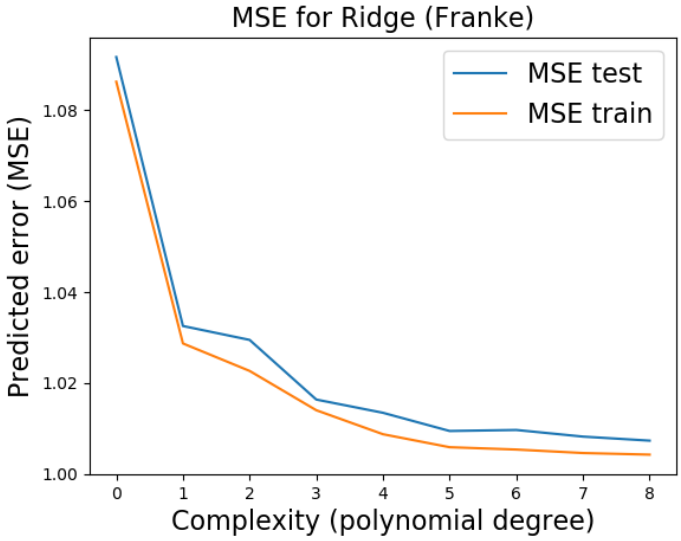

FIG. 17.— The MSE for the test set as a function of complexity, Ridge regression on the Franke function

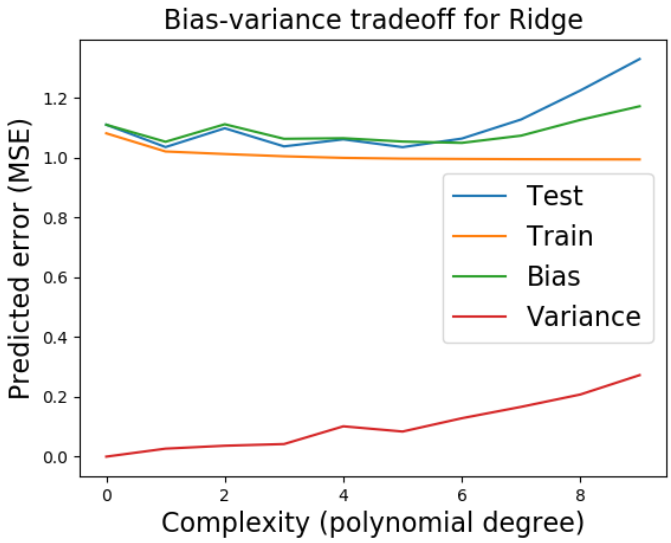

FIG. 19.— Bias-variance tradeoff for Ridge regression

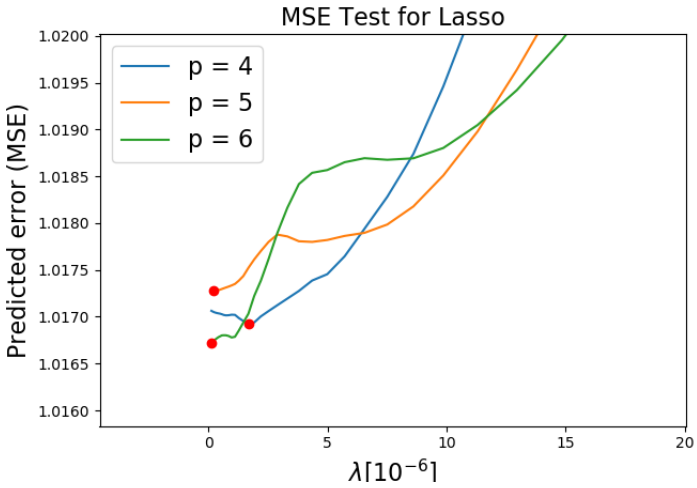

FIG. 20.— Zoomed in on the lowest MSE of the test data.

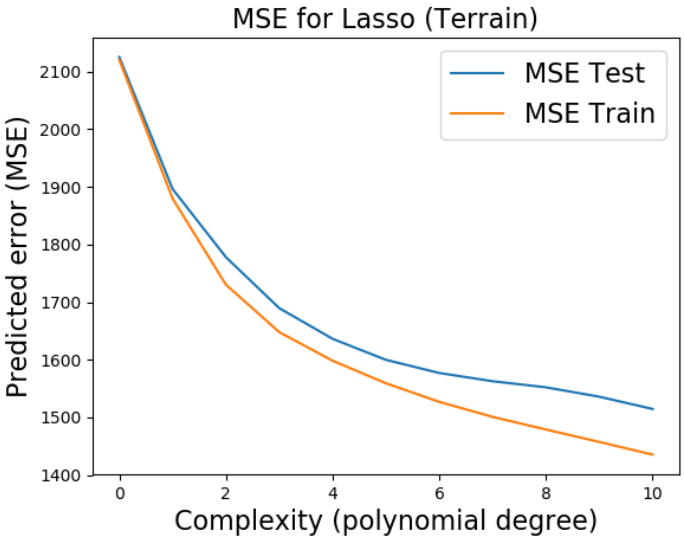

FIG. 22.— MSE as a function of model complexity for the terrain data

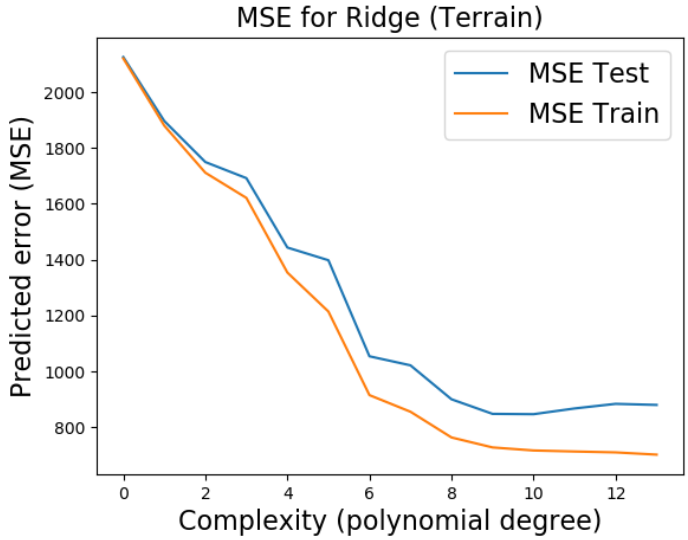

FIG. 21.— MSE as a function of model complexity for the terrain data

APPENDIX B

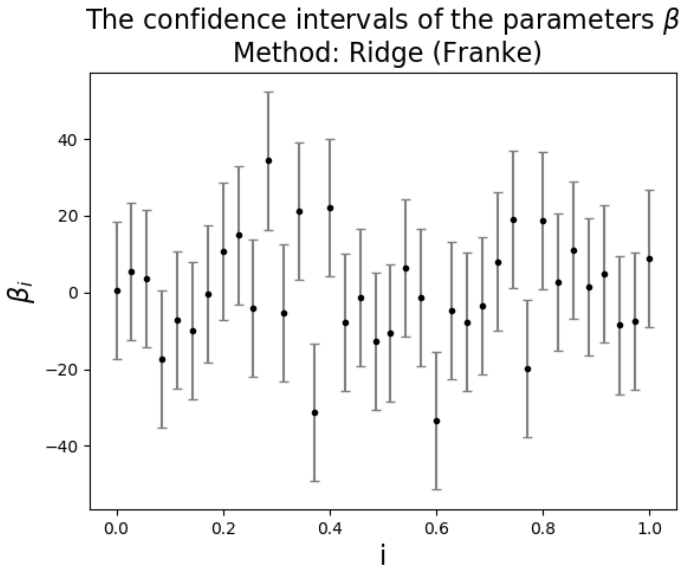

FIG. 23.— The confidence intervals of the parameters β for Ridge regression

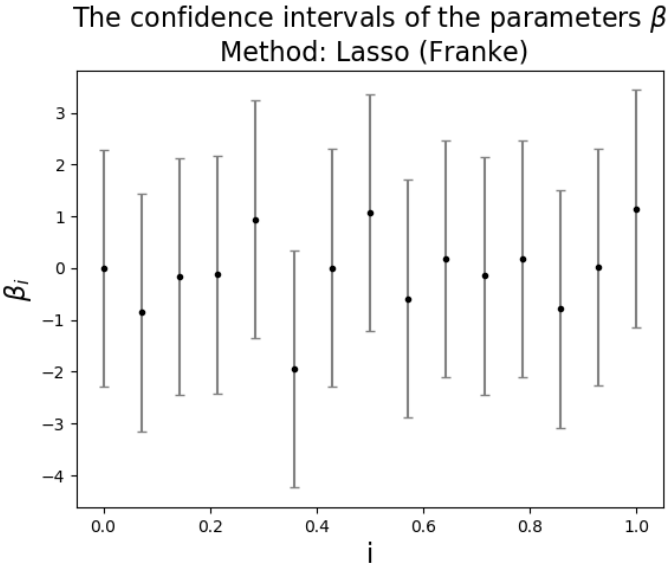

FIG. 24.— The confidence intervals of the parameters β for Lasso regression

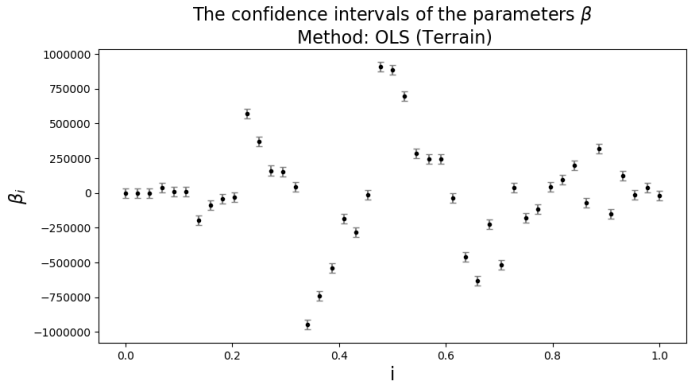

FIG. 25.— The confidence intervals of the parameters β for OLS regression, terrain data.

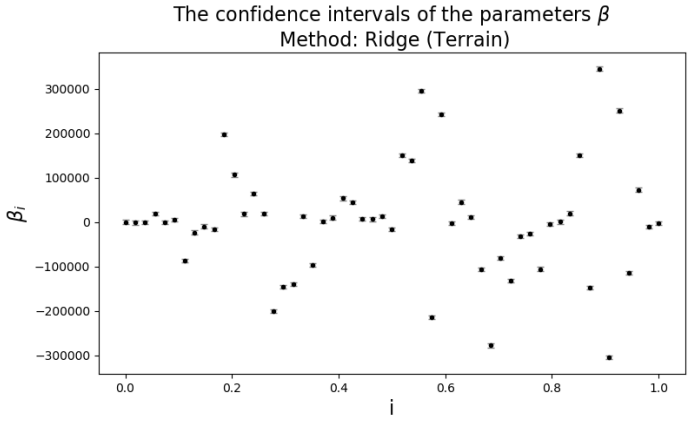

FIG. 26.— The confidence intervals of the parameters β for Ridge regression, terrain data

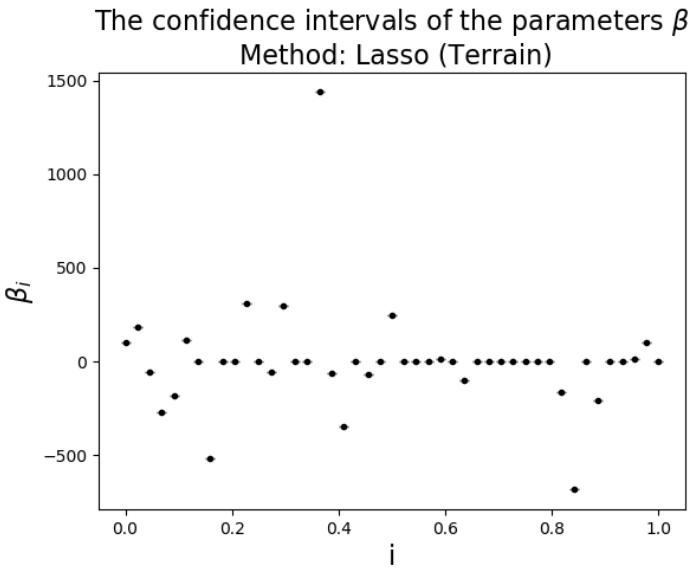

FIG. 27.— The confidence intervals of the parameters β for Lasso regression, terrain data

7. APPENDIX C

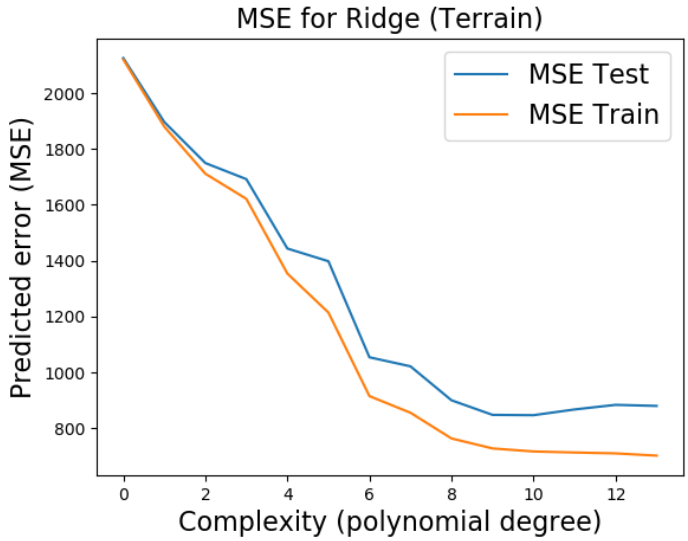

FIG. 28.— MSE as a function of model complexity for the terrain data

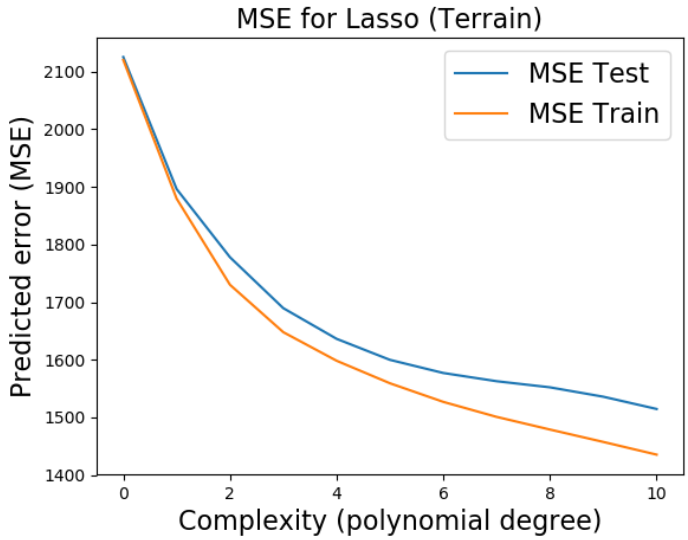

FIG. 29.— MSE as a function of model complexity for the terrain data

8. APPENDIX D

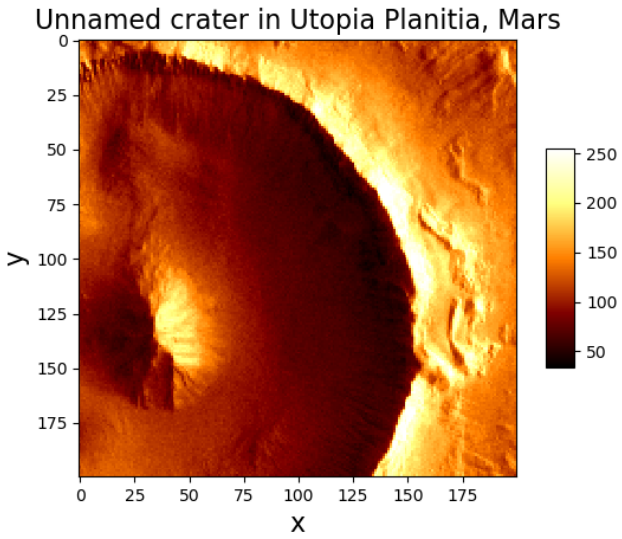

FIG. 30.— The cropped and resized image of the unnamed crater on Mars

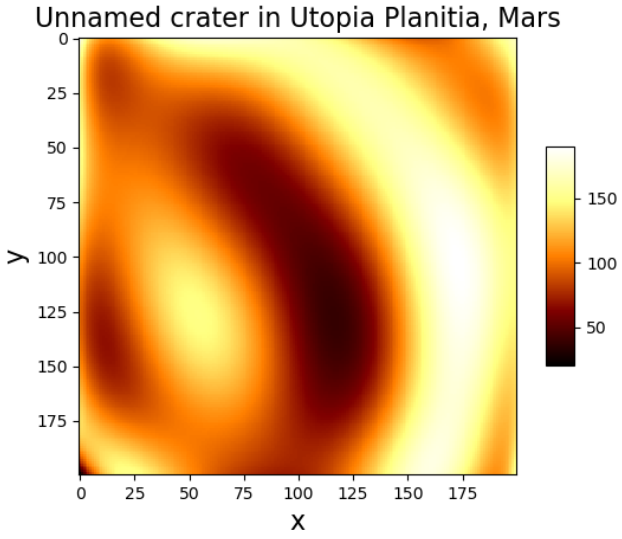

FIG. 31.— The best terrain image for OLS

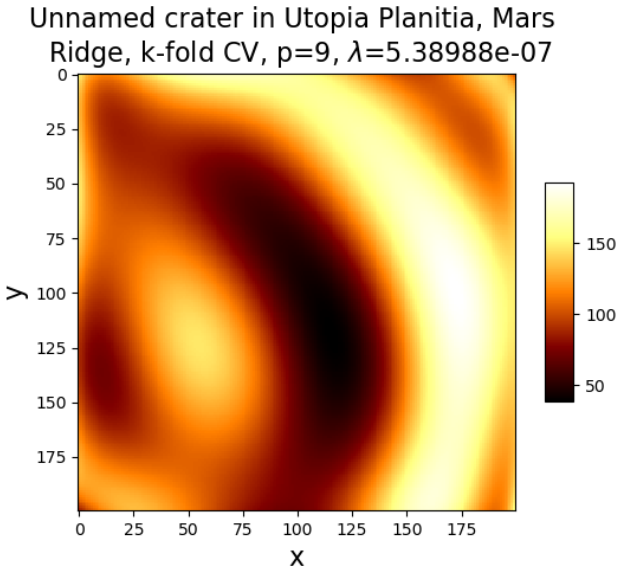

FIG. 32.— The best terrain image for Ridge

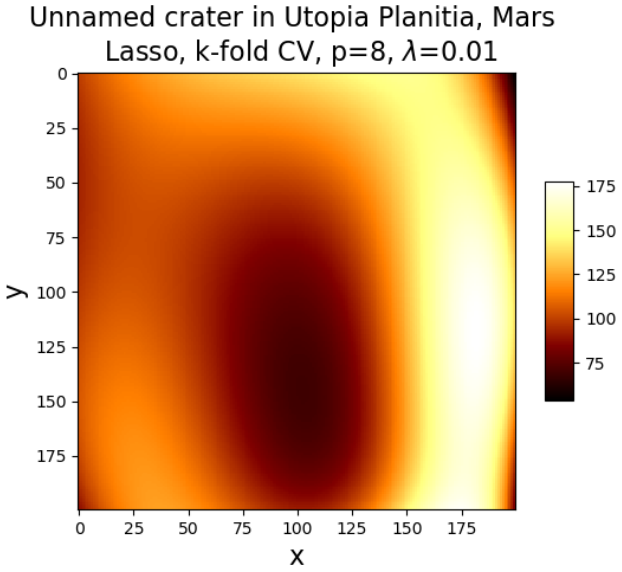

FIG. 33.— The best terrain image for Lasso

APPENDIX E

p	MSE_{OLS}	MSE_{Ridge}	λ_{Ridge}	MSE_{Lasso}	λ_{Lasso}
0	1.092	1.092	0.1		
1	1.033	1.032	0.1		
2	1.030	1.030	0.1		
3	1.016	1.016	0.1		
4	1.014	1.013	$3.2 \cdot 10^{-2}$		
5	1.010	1.009	$3.8 \cdot 10^{-3}$		
6	1.007	1.007	$3.4 \cdot 10^{-5}$		
7	1.010	1.007	$1.5 \cdot 10^{-4}$		
8	1.011	1.007	$4.1 \cdot 10^{-4}$		
9	1.040	1.007	$3.4 \cdot 10^{-4}$		

TABLE 3
TABLE SHOWING THE TEST MSE FOR OLS, RIDGE AND LASSO

REFERENCES

[1] https://github.com/CompPhysics/MachineLearning/tree/master/doc/Projects/2019/Project1/pdf
[2] https://photojournal.jpl.nasa.gov/catalog/PIA23328?fbclid=IwAR0s89UNmpcVSpXViHhDoakOR1x51o4tSzUW6iHePyEY__rUsqgiDuqeDGY
[3] https://statisticsbyjim.com/regression/ols-linear-regression-assumptions/
[4] https://github.com/kristinaothelia/FYS-STK4155/tree/master/Project1