ORIGINAL ARTICLE

Journal Section

FYS-STK Project 1 on Machine Learning Fall 2019

Alexander D. Aliferis | Marius Stette Jessen

Department of Physics, University of Oslo, Norway

[1]Department of Physics, University of Oslo, Norway

Correspondence
Email: alexadal@fys.uio.no

This report considers performance analysis of three different regression methods, Ordinary Least Squares, Ridge and Lasso on two different data sets. The performance of the different methods is compared and analysed. One of the data sets is generated from the well-known Franke function with added noise, while the other set is from real terrain data from an area in Norway. The python code supporting the analysis in this report is uploaded to the GitHub repository. The file runMe.py in the master branch provides a program that can be used to analyse the data-sets used in the report.

KEYWORDS

Ordinary Least Squares, Ridge, Lasso, Machine Learning

1 | INTRODUCTION

In terms of modern science and general data analysis, there are huge efforts in finding patterns and making predictions based on historical observations. These efforts go across a large variety of disciplines and their results have already made significant impact on politics, industry, healthcare and trading to mention some. An ever more trending expression these days, describing the process of predicting certain outcomes based on obtained data, without necessarily giving specific sets of instructions, is Machine Learning. The many sets of algorithms that build up under the subject of Machine Learning, can be divided into two main categories - supervised and unsupervised learning. Within the category of supervised learning, regression models are widely used, and one of the most common subgroup of these is linear regression. This paper will focus on the implementation of three different types of linear regression models, namely ordinary least squares, ridge and lasso regression. Their corresponding algorithms will be used to fit the true Franke function generated with added random noise, as well as real terrain data from the coastline of Norway. The study will further assess how well the different models perform in terms of statistical properties such as mean squared error, R^2-score and how the interpretability of the models varies with flexibility.

2 | METHODOLOGY

The general theory in this section is obtained from the course lecture notes Hjorth-Jensen [3] and the textbook Hastie et al. [2].

2.1 | Linear regression models

.

Regression models make predictions on the response value Z, based on new inputs $\boldsymbol{X}$, and a function $f(\boldsymbol{X})$. This relationship is given by equation 1

$$Z = f(\boldsymbol{X}) + \epsilon \tag{1}$$

where ϵ is the irreducible error and represent measurement errors and other discrepancies. The function $f(\boldsymbol{X})$ itself, requires by statistical decision theory, a punishment for prediction-errors in the form a loss function, $L(Z, f(\boldsymbol{X}))$. The most common loss function is the mean squared error loss, $L(Z, f(\boldsymbol{X})) = (Z - f(\boldsymbol{X}))^2$, and the function that minimizes this loss is called the regression function $f(\boldsymbol{X}) = E(Z|X = x, Y = y...)$. With limited amounts of data points however, it is seldom possible to compute the exact expectation of Z at any given point. Instead, the function is relaxed and approximated. A good and easily interpretable approximation is the linear regression function, where the ideal function is parameterized by $p + 1$ parameters $\beta = [\beta_0, \beta_1, ...\beta_p]$

$$f(\boldsymbol{X}) = \beta_0 + \beta_1 \boldsymbol{X_1} + \beta_2 \boldsymbol{X_2} + ... + \beta_p \boldsymbol{X_p} \tag{2}$$

or in matrix representation

$$f(\boldsymbol{X}) = \boldsymbol{X}\beta \tag{3}$$

Here, $\boldsymbol{X_1}, \boldsymbol{X_2}, .., \boldsymbol{X_p}$ are vectors and column elements of the design matrix, $\boldsymbol{X}$,which is a $N \times p$ dimensional matrix. N is the number of observations or measured values and p is the characteristic of the N^{th} observation. The number of parameters in the design matrix depends on the complexity, or in other words, the polynomial degree of the regression model. In the case of this paper, where the predictions are based on two independent variables, the number of parameters $p = ((degrees + 1) \times (degrees + 2))/2$.

By using the dependencies above, one can obtain estimated values of the parameters or coefficients, $\hat{\beta}$, and compute predictions on new sets of observations $\tilde{Z}$. There are several ways of optimizing these coefficients, and perhaps some of the most popular ones are given in the subsequent sections.

2.1.1 | Ordinary Least Squares, OLS

The squared sum of residuals, RSS, is defined by taking the spread of each observation-prediction pairs, $z_i - \tilde{z}_i$, and squaring the sum of these pairs

$$RSS_i = \sum_i^p (z_i - \tilde{z}_i)^2 \tag{4}$$

Similarly, one can obtain equation 5 by using the equality $\tilde{Z} = \hat{f}(\boldsymbol{X}) = \boldsymbol{X}\hat{\beta}$, equation 2 and matrix notation

$$RSS(\beta) = (Z - \boldsymbol{X}\hat{\beta})^T (Z - \hat{\beta}\boldsymbol{X}) \tag{5}$$

Minimizing equation 5 by taking its derivative with respect to $\hat{\beta}$, gives the unique solution

$$\hat{\beta} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T Z \tag{6}$$

The expression in equation 6 can, if $(\boldsymbol{X}^T \boldsymbol{X})$ is non-singular and invertible, be used to evaluate $\tilde{Z}$.

OLS is a powerful tool that often gives good and easily interpretable predictions with a relatively "simple" estimation procedure. However, the model has some drawbacks and limitations. In particular, if the number of parameters become very large and $p > N$, the design matrix cannot have linearly independent columns, and the expression $\boldsymbol{X}^T \boldsymbol{X}$ in equation 6 is not invertible. OLS is also sensitive to extreme observations or outliers. This is a problem that rise from the cost function itself. For extreme observations, the coefficients in the model will become large, minimizing the distance between the regression line and the outliers. Hence, outliers will affect the slope of the regression line to a greater extent than observations near the mean value of the data set.

To mitigate these negative effects, one can use shrinkage methods that modify the OLS model. Two examples of these are the Ridge and Lasso models that penalize the coefficients for becoming too large, and tries to shrink them towards zero.

2.1.2 | Ridge Regression

Similarly to OLS, Ridge regression tries to minimize the the distance between the regression line and the data set. However, it also applies a shrinkage method that results in the following cost function

$$RSS + \lambda \sum_j^p \beta_j^2 \tag{7}$$

where $\lambda \in \{1, \ldots, \infty\}$, is a tuning parameter that is to be determined by cross-validation. Cross-validation will be discussed later in the methodology. Minimizing the expression in 7, gives the solution

$$\hat{\beta}^{Ridge} = (\boldsymbol{X}^T \boldsymbol{X} + \lambda \boldsymbol{I})^{-1} \boldsymbol{X}^T \tilde{Z}. \tag{8}$$

$\boldsymbol{I}$ is the identity-matrix and multiplied with λ, it creates a diagonal matrix that resolves any singularity of $\boldsymbol{X}^T\boldsymbol{X}$. Yet, Ridge regression still holds the disadvantage that it contains coefficients for all variables p, as it can only shrink their values towards, but not equal to zero.

2.1.3 | Lasso Regression

Lasso regression is a model that excludes some of the predictors by having the cost function in equation 9.

$$RSS + \lambda \sum_{j}^{p} |\beta_j| \tag{9}$$

This cost function, on the contrary to the two previously mentioned ones, does not have a simple analytical solution for its coefficients. Therefore, for the purpose of this study, a built-in function from Skicit-Learn will be used to evaluate Lasso-coefficientsSci [1]. Scikit-Learn uses gradient descent, which is an iterative method to minimize the cost function.

A final note to the penalty models (Lasso and Ridge), are whether or not to include the intercept in the computation of parameters. As the intercept is closely related to the expected value of the function "zero-ground", penalizing the intercept can lead to bad predictions if the "zero-ground" of the function is large in value. To do this for the Ridge-model, one has to center the design matrix by subtracting the mean of each column. The same has to be done with the corresponding observation set. It is also important to exclude the first column in the design matrix to avoid having a singular matrix in the computation of parameters, and lastly, to add the observation-mean to the predictions before evaluating model performance. However, with limited time-resources this has not been implemented in the current analysis. The authors still anticipate that the analysis will give fair predictions as the Franke-function gives a "hilly" surface that extends from low to high values. Similarly in the case of real data, choosing a terrain that does not explicit contain a mountain but a slope that extends from water level. The Skicit-Lasso model on the other hand, has a built in option to omit the intercept that is used throughout the results.

2.2 | Resampling

When using the models in the previous section to compute predictions, one wishes to obtain a fresh set of samples and evaluate how well the predictions perform. This is however not always possible, as data might be limited. Also, re-using the sample pairs of Z and $(X = x, Y = y \dots)$ that have determined the model parameters, will give too promising results.

2.2.1 | K-fold Cross Validation

A solution to the problem of limited data is cross-validation. Cross-validation divides the data into two groups and label them as training-data and test-data. The training-data are used to fit the model parameters and the test-data are used to validate the corresponding fit these parameters generate. To compensate for any dependency of which parts of the data set that are labeled as test and train data, the K-fold resampling technique repeats the process as follows:

1. Shuffle the data randomly
2. Divide the data into k number of equal-sized groups or "folds"

3. Thereof, for every fold do:
   a. Decide the group used for validation
   b. Use the remaining folds as training data
   c. Fit the model with the training data and evaluate its performance with the test-data
   d. Retain the statistical (to be defined later) scores and discard the model
4. The cross-validation score is sets of statistical properties for the k-fold. Complete the k-fold method by averaging these.

The final, and perhaps most critical part of the algorithm is to choose the number of folds. Choosing too few folds give training sets that are quite similar and correlated, creating high variance in the expected test error. On the contrary, too many folds will give larger bias in the predictions as the training data are only representing small parts of the original training set. A good compromise is taking the number of folds equal to 5 or 10. For the purpose of this study, the number of folds is set to 5.

2.2.2 | Bootstrap

Bootstrap resampling is different from KFold as it resamples with replacement. The procedure is described by the following process

1. Split the data-set into test and train data
2. With the train data, for the number of straps selected, randomly draw a new set of independent variables to construct a surrogate train-data-set.
3. Use the surrogate train-data-set to construct model parameters and hence predictions for new observations.
4. Evaluate how well the predictions are relative to the test-data from point 1.

To compare, K-fold cross-validation is primarily used to measure model performance by estimating prediction errors, whereas bootstrap merely give standard errors of the predictors. Thus, for the purpose of this study, Kfold is the main re-sampling technique used for model performance. Bootstrap is however included to illustrate the bias-variance relationship with the Mean-Squared-Error(MSE), which will be discussed further in detail under section 2.3.

2.3 | Statistical properties

To better understand how well the different regression models are performing in the sense of predictions, the following statistical properties are used.

2.3.1 | Variance

The variance-covariance matrix of $\hat{\beta}$, can be found by the following expressions for OLS and Ridge regression respectively

$$Var\left(\hat{\beta}_{OLS}\right) = (\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X})^{-1}\sigma^2 \tag{10}$$

$$Var\left(\hat{\beta}_{Ridge}\right) = (\boldsymbol{X}^T\boldsymbol{X} + \lambda_{diag.M})^{-1}\sigma^2 \tag{11}$$

An unbiased unbiased estimate for σ, can be found by

$$\hat{\sigma}^2 = \frac{1}{N-(p-1)}\sum_i^N(\tilde{z}_i - z_i) \tag{12}$$

Where $\tilde{z}_i$ is a single prediction for observation z_i, that comes from a pool of N corresponding prediction-observations pairs $\tilde{Z}$ and Z. Similarly to prior sections, p stands for the number of parameters.

As the variance of the coefficients are the diagonal elements of $Var(\hat{\beta})$, equation 10 and 12 can be used to investigate confidence intervals of the coefficients $\hat{\beta}$. This is done by defining $STD_{\hat{\beta}}$ as the squared root of the variance and the expression

In the current study, as the actual variance of the noise is available for the Franke-function, the estimation in 12 is replaced for the actual variance of the noise.

$$\hat{\beta} \pm 1.96 \times STD_{\hat{\beta}} \tag{13}$$

2.3.2 | MSE ans R2-score

To evaluate the model performances, the statistical measures mean squared error, MSE, and R2-score are used. These are respectively expressed in the equations below.

$$MSE = \frac{1}{N}\sum_i^N(z_i - \tilde{z}_i) \tag{14}$$

$$R2 = 1 - \frac{\sum_i^N(z_i - \tilde{z}_i)}{\sum_i^N(z_i - \bar{z}_i)} \tag{15}$$

MSE is simply the average value of the prediction errors, giving the most optimal score of zero. R2 on the other hand, which is a normalized version of MSE that measures how well the variance of observations are explained by the regression line, has the most optimal score of 1.

2.3.3 | Bias-Variance Trade-off

The MSE can be decomposed into three components that describes different prediction behaviour as seen below

$$MSE = E[(Z - \tilde{Z})^2] = E[\tilde{Z}^2] - 2 \times E[\tilde{Z}]E[Z] + E[Z^2] = E[(\tilde{Z} - \bar{\tilde{Z}})^2] + \bar{\tilde{Z}}^2 - 2 \times (\bar{\tilde{Z}} f(\boldsymbol{X})) + E[(y - f(\boldsymbol{X}))^2] + f(\boldsymbol{X})^2$$
$$= E[(\tilde{Z} - \bar{\tilde{Z}})^2] + (\tilde{Z} - f(\boldsymbol{X}))^2 + E[(Z - f(\boldsymbol{X}))^2] = Var(\tilde{Z}) + Bias(\tilde{Z})^2 + E[\sigma^2])$$

Where the lemma

$$E[Z] = \frac{1}{N} \sum_{i}^{N} (Z) = \bar{z}_i \tag{16}$$

$$E[Z^2] = E[(Z - \bar{Z})^2] + \bar{Z} \tag{17}$$

is used. The decomposition gives a description of how the model complexity changes the different sources of the error. With a high model complexity one expect that the prediction bias will decrease, but unfortunately, the variance of the on the other hand will increase. That is, the certain points of prediction will give a better fit relative to the observations, but the predictions will be much more fluctuating as the sampled points are changed.

3 | RESULTS

3.1 | Franke Function

The Franke function is described by the following equation

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

Figure 1 illustrates the MSEs for the various models with increasing degree of model complexity are illustrated. The Franke-function has in all these scenarios a normally distributed added noise with mean and variance equal to 0 and 1 respectively. In the middle two plots, the MSE is also adjusted for the tuning parameter λ. Results of the lowest MSE-scores from the models are also listed in table 1. It should be noted that the Lasso model in some of the algorithm-runs had issues with converging leading to some difficulties in obtaining the absolute optimal λ. Nevertheless, with this much level of noise there is not easy to distinct the models from each other. Ridge gives the better overall score closely followed by Lasso and OLS. This result is likely determined by the variance of OLS becoming successively large as discussed more in detail below.

TABLE 1 Results Franke Function

Method	MSE	R2
OLS(Deg = 4)	0.0107	0.8698
Ridge(Deg = 5, $\lambda = 10^{-3}$)	0.0096	0.8836
Lasso (Deg = 3, $\lambda = 10^{-4}$)	0.0103	0.8739

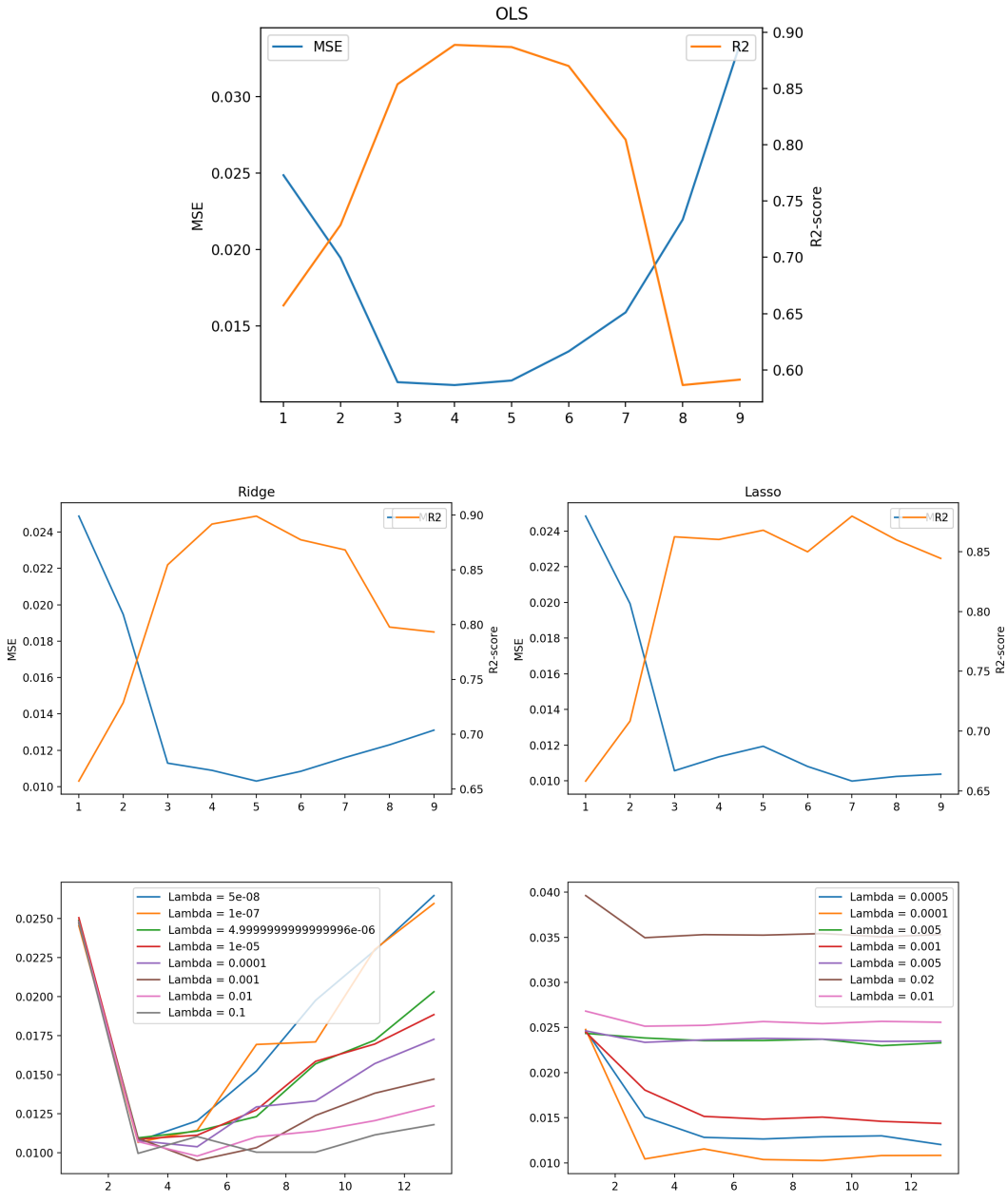

FIGURE 1 The results are after cross-validation using 5 folded KFold *Top center:* OLS *Left:* Ridge optimal $\lambda = 10^{-3}$ *Right:* Ridge optimal $\lambda = 10^{-4}$

Further, by evaluating the 95% confidence intervals of the models, it is clear that the parameters are highly affected by the noise level. These results are listed in the tables 4-9. Since the polynomials are of different order, the coefficient sizes are not directly comparable. Nevertheless, the behaviours of beta as a function of the tuning parameter λ are also shown in figure 2. There is an obvious distinction between Ridge and Lasso as the coefficients of Ridge never reaches zero. The damping of the parameter size is also much more rapid.

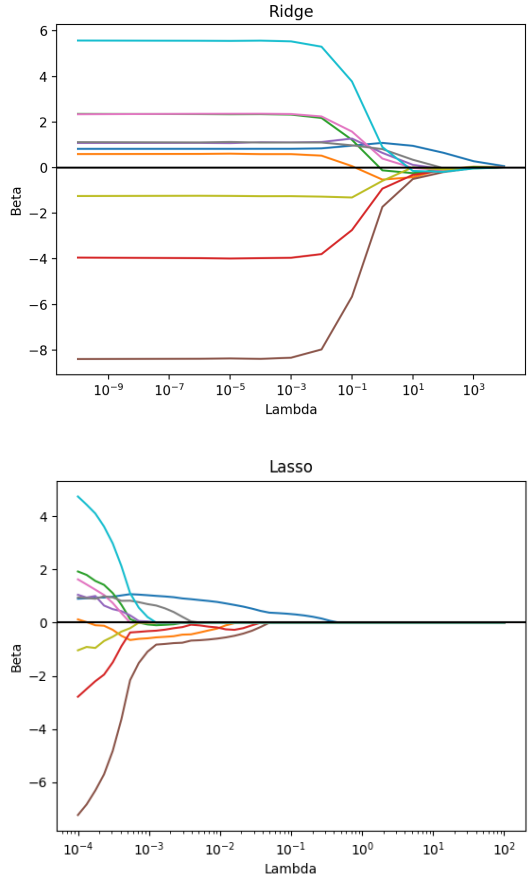

FIGURE 2 The plots show the development of beta relative to *lambda Top:* Ridge degree 3 *Bottom:* Lasso degree 3

Dividing the data into train and test samples, performing Kfold cross-validation and studying the development of model MSE, derives the bias-variance trade-off as seen by figure 5. Although the scaling is slightly different, there are, not surprisingly, lower spreads for the coefficient-penalizing models. Note that the values of λ are set high to illustrate the effect of damping the parameters and thus reducing the train-test-split. Further, note that the calculation of statistics here is not based on validation set from the true Franke-function but a noisy one resulting in poor performance. A comparison of MSE and R2-score for OLS with and without cross-validation is showcased in table 2. There is a clear difference between the values as the No-resample case is highly dependent on the train-test set used.

TABLE 2 Statistics comparision with and without resampling

Method	MSE	R2
OLS(Deg = 4, Cross-validation)	1.04867	0.0733
OLS(Deg =4, No-resample)	1.0319	0.0864

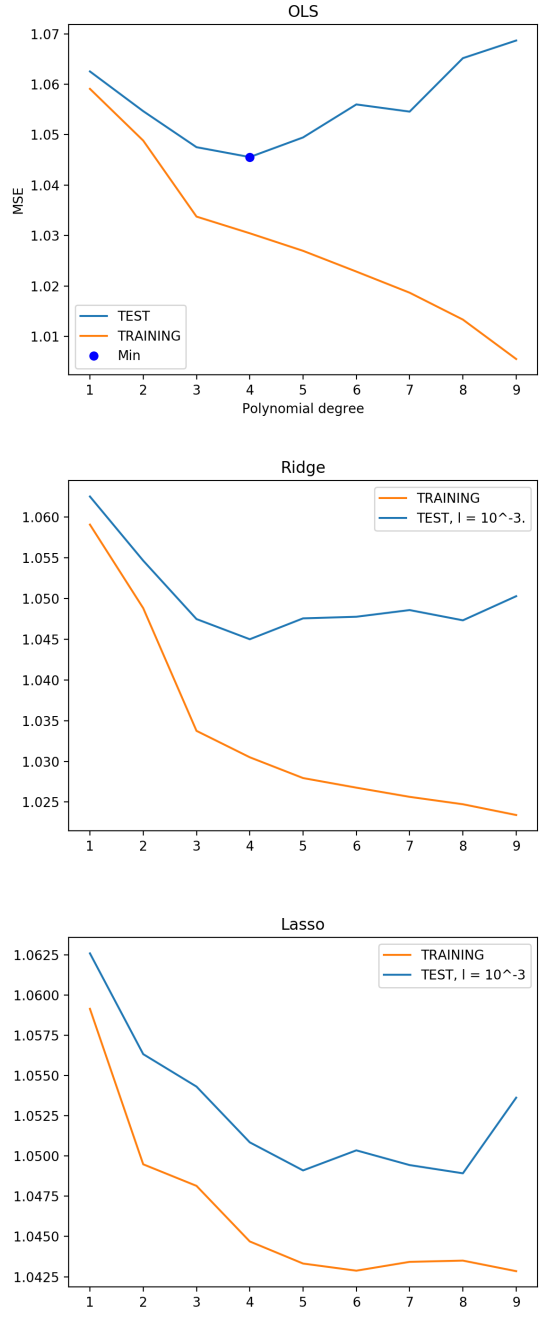

FIGURE 3 The plots show MSEs for train and test data used in fitting the Noisy Franke-function. The *Top:* OLS *Middle:* Ridge $\lambda = 10^{-5}$ *Bottom:* Lasso $\lambda = 10^{-3}$

Lower train-test spreads are also illustrated by bias-variance decomposition of ME in figure 4. For OLS, there is a very strong dependency of the variance to the polynomial degree of the fit, and the added noise factor σ. This phenomenon arises from the models over-fitting the noise instead of the general behaviour of the true Franke-function function. Evaluating the other model plots, there is a significant lower dependency of the variance as the coefficients are damped, giving a better expected prediction-performance for higher degrees of polynomial fit.

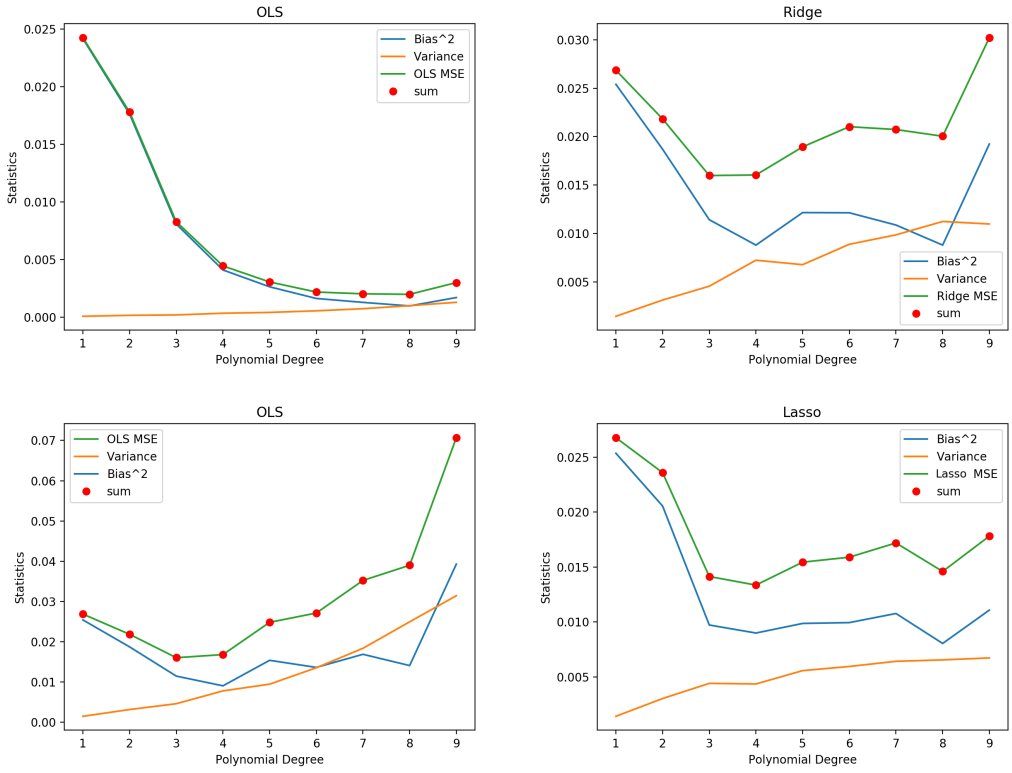

FIGURE 4 The plots show the decomposition of bias and variance for OLS fitting the Noisy Franke-function. The number of data points N = 2500. To generate the plot, bootstrap cross-validation with 50 straps is used. *Top Left:* OLS, σ = 0.2 *Bottom Left:* OLS, σ = 1*Top Right:* Ridge, σ = 1 *Bottom Right:* Lasso, σ = 1

3.2 | Real Terrain Data

To widen the perspective of the model performances, they have also been applied to real terrain data from the west coast of Norway. The data used are imported from a geoTiff-file that originates from a "SRTM 1 arc-second global" radar scan. The resolution of the terrain data are about 1 point for every 30 meters. To avoid extensive time-consumption on running the models on the complete terrain file, a subset-sample of size 25×25 and is chosen. Following the argumentation of not omitting the intercept in the ridge model, the sample is also somewhat chosen to contain a varying landscape and not a single mountain.

In table 3, one can see that the Ridge model again gives best statistical scores, and hence the most adequate fit. This is seemingly due to the Ridge function reaching higher levels of complexity without having to suffer from large variance. In other words, the damping of lambda is sufficient to maintain the variance at a low level, without giving up on too much of resolution in the fit. The R2-score also show that all the models are performing well by describing the majority of variance in the observations. Figure 6 visualizes the predicted values from each regression method together with the real terrain data. Observable in the figure, is how data is lost in the highly λ-dependable Lasso model, compared to the Ridge model.

TABLE 3 Results Terrain

Method	MSE	R2
OLS(Deg = 9)	20.5942	0.9901
Ridge(Deg = 14, $\lambda = 10^{-9}$)	15.7464	0.9925
Lasso (Deg = 20, $\lambda = 10^{-12}$)	57.6468	0.9712

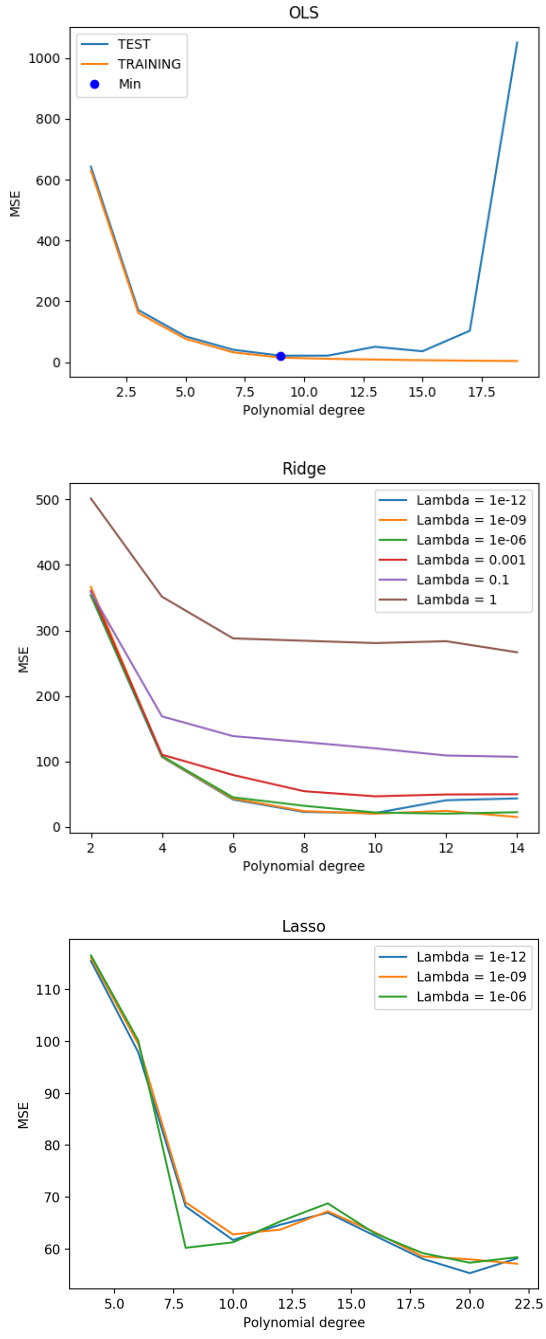

FIGURE 5 The plots show MSEs for train and test data used in fitting the Noisy Franke-function. The *Top:* OLS *Middle:* Ridge *Bottom:* Lasso

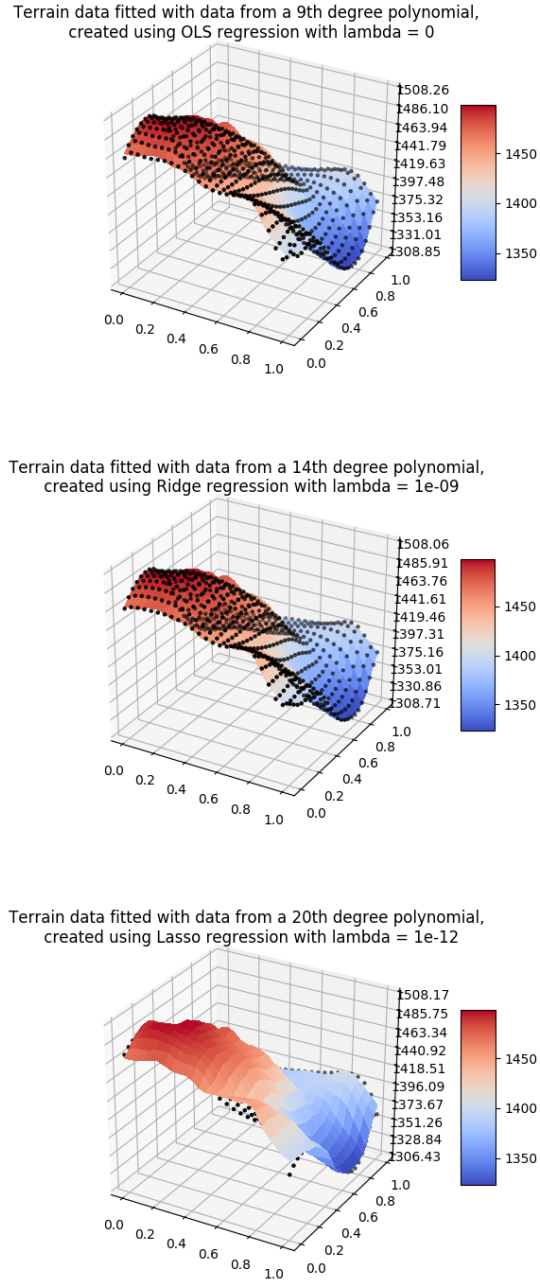

FIGURE 6 The dots shows the predicted values for the three different regression methods on the terrain data.

4 | CONCLUSION

In this project, both terrain data and data generated from the Franke Function have been analysed by using three different types of regression: Ordinary Least Squares (OLS), Ridge and Lasso. The results obtained show that Ridge regression give the overall best performance, regardless of the data-set used. From a more detailed discussion of the bias-varaince theory, this result is mainly explained by how the OLS-model suffers from high variance with high levels of complexity. The variance again, is found to be highly correlated to the noise. By "removing" some noise while still retaining much of the parameter resolution, Ridge is able to outperform both the other models. Lasso regression, simply looses too much information to give the same fit. With less noise, lower resolution of the parameters would seemingly reduce the model performances leading to more optimal predictions by OLS.

Although not completely accounted for by this study, the authors expect that increasing the sample-size of the terrain will give better scores for OLS as the level of over-fitting is inverse-proportionally reduced. Reducing the sample size on the other hand, is expected to give the opposite results as singularity might appear for low complexities.

Thus, as a follow-up to this paper, evaluating model statistics for both lesser and greater amounts of data is of strong interest. Further, applying even more complex and less interpretable models within regression such as Log-regression could perhaps give better results.

REFERENCES

1. () Scikit-learn, 1.5. stochastic gradient descent howpublished = https://scikit-learn.org/stable/modules/sgd.html, note = Accessed: 2019-10-09.

2. Hastie, T., Tibshirani, R. and Friedman, J. H. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition*. Springer Series in Statistics. Springer. URL: https://doi.org/10.1007/978-0-387-84858-7.

3. Hjorth-Jensen, M. (2019) Lecture notes fys-stk4155.

TABLE 4 **OLS, degree =4 - Franke Function:** Beta 95 % Confidence Intervals, $\sigma = 0$

β_1	0.623518	±	0.0
β_2	4.164318	±	0.0
β_3	2.994711	±	0.0
β_4	-19.168284	±	0.0
β_5	-2.505195	±	0.0
β_6	-11.684622	±	0.0
β_7	25.029755	±	0.0
β_8	8.62138	±	0.0
β_9	1.443741	±	0.0
β_{10}	11.386474	±	0.0
β_{11}	-10.760773	±	0.0
β_{12}	-5.478523	±	0.0
β_{13}	-0.058134	±	0.0
β_{14}	-1.91369	±	0.0
β_{15}	-2.916142	±	0.0

TABLE 5 **OLS, degree = 4 - Franke Function:** Beta 95% Confidence Intervals, $\sigma = 1$

β_1	0.827539	±	0.434242
β_2	1.837049	±	3.532311
β_3	2.511331	±	3.532311
β_4	-12.438406	±	11.847608
β_5	7.015032	±	9.254296
β_6	-13.367646	±	11.847608
β_7	16.059963	±	16.656077
β_8	-1.909781	±	12.724447
β_9	-10.262892	±	12.724447
β_{10}	16.369411	±	16.656077
β_{11}	-6.45715	±	8.239405
β_{12}	-2.148939	±	7.170325
β_{13}	6.299229	±	7.034064
β_{14}	1.943276	±	7.170325
β_{15}	-5.989764	±	8.239405

TABLE 6 **Ridge, Degree = 5 - Franke Function:** Confidence Intervals, $\sigma = 0, \lambda = 10^{-3}$

β_1	0.681115	±	0.0
β_2	3.300657	±	0.0
β_3	2.552148	±	0.0
β_4	-14.113471	±	0.0
β_5	-1.934789	±	0.0
β_6	-8.777169	±	0.0
β_7	11.340326	±	0.0
β_8	10.168883	±	0.0
β_9	2.119743	±	0.0
β_{10}	1.439972	±	0.0
β_{11}	5.743697	±	0.0
β_{12}	-12.602466	±	0.0
β_{13}	6.983559	±	0.0
β_{14}	-10.976937	±	0.0
β_{15}	12.129305	±	0.0
β_{16}	-6.988068	±	0.0
β_{17}	2.840049	±	0.0
β_{18}	2.377706	±	0.0
β_{19}	-7.037836	±	0.0
β_{20}	8.343375	±	0.0
β_{21}	-7.789916	±	0.0

TABLE 7 **Ridge, Degree = 5 - Franke Function:** Confidence Intervals, $\sigma = 1, \lambda = 10^{-3}$

β_1	0.843484	±	0.431636
β_2	1.05597	±	3.868083
β_3	2.329927	±	3.868083
β_4	-6.257825	±	16.197381
β_5	5.34329	±	13.964064
β_6	-9.556135	±	16.197381
β_7	-0.058692	±	34.91144
β_8	-2.432883	±	30.355751
β_9	-1.786426	±	30.355751
β_{10}	0.110799	±	34.91144
β_{11}	10.490852	±	38.349378
β_{12}	0.942307	±	36.267947
β_{13}	6.894706	±	34.521093
β_{14}	-14.748561	±	36.267947
β_{15}	18.921265	±	38.349378
β_{16}	-6.046581	±	16.62622
β_{17}	-4.4728	±	19.320228
β_{18}	5.736752	±	20.469916
β_{19}	-6.279287	±	20.469916
β_{20}	11.803795	±	19.320228
β_{21}	-12.590092	±	16.62622

TABLE 8 **Lasso, degree 3 - Franke Function:** Confidence Intervals, $\sigma = 0, \lambda = 10^{-4}$. Brackets used as standard deviation estimated with variance of betas in kfold

β_1	0.972102	±	[1.3961]
β_2	-0.18472	±	[1.388]
β_3	0.798125	±	[1.2991]
β_4	-2.353656	±	[1.3889]
β_5	0.895691	±	[1.3928]
β_6	-3.610247	±	[1.4154]
β_7	1.931199	±	[1.3808]
β_8	0.829016	±	[1.3587]
β_9	0.0	±	[1.3547]
β_{10}	0.0	±	[1.4039]

TABLE 9 **Lasso, deegree 3 - Franke Function:** Confidence Intervals, $\sigma = 1, \lambda = 10^{-4}$. Brackets used as standard deviation estimated of betas in kfold

β_1	0.951278	±	[1.4761]
β_2	-0.133652	±	[1.6544]
β_3	0.985796	±	[1.4255]
β_4	-1.840187	±	[1.6228]
β_5	2.418584	±	[1.5519]
β_6	-3.820452	±	[1.6764]
β_7	0.018622	±	[1.6602]
β_8	0.056588	±	[1.6577]
β_9	-2.605992	±	[1.4849]
β_{10}	0.0	±	[1.5046]