Project 1: FYS-STK4155

Regression analysis and resampling methods

**Abstract**

**Introduction**

In this project our aim was to study the behavior of various regression methods for fitting and interpolation applied to both synthetic test data and real data. The regression methods which were used was the ordinary least squares (OLS) and least squares regression with two different regularization terms – ridge regression and lasso regression – which are also known as shrinking methods.

T**heoretical foundation**

**The ultimate goal of machine learning methods is to provide automatically detection of patterns in data, and use these “learned” patterns to predict future data. There exist a lot of different methods such as logistic regression, linear regression, OLS and regularization** – including **ridge- and lasso regression** –**, and many more. In this project we will focus on OLS, ridge- and lasso regression. The derivations of these methods follows from Wieringen (2018). The aim of regression analysis using linear regression is to explain a response variable in terms of an explanatory variable via the assumption of a linear relationship between and (Wieringen 2018). In machine learning we commonly denote the jointly distributed set of measurements as the training set. Then the linear regression model is defined as**

**Where is the noise term and is a -dimensional design matrix defined as**

**and -dimensional coefficient vector which are the parameters we want to estimate in order to fit our model to the data. The parameter is commonly referred to as the bias term (or intercept) implying that in our model the first column of the design matrix is defined as a constant variable of one’s: . Suppose the random variable is normally distributed, with expected value of the model and the variance, hence. From the regression model we then have two unknowns: and . We fit our model to the training data and estimate the parameters by maximizing the log-likelihood function (Wieringen 2018)**

We maximize the log-likelihood in  **by taking the partial derivative with respect to** and setting this to zero

If we assume that is positive definite we have the unique solution

**The OLS solution to the model described in then reads**

**The residual variance is estimated through taking the partial derivative of with respect to and setting this to zero**

which gives the estimate of the residual variance

The variance of

In case of singularity (of the matrix ), there exist many methods to attack this problem. Two methods which we will be looking into is the Ridge regression and Lasso. Ridge regression introduce an norm penalty on the parameters. We can write the cost function with the penalty parameter as follows (Hastie et al,)

where  **is commonly referred to as the shrinkage parameter.** Taking the derivative with respect to we get

Which gives

We see that the shrinkage parameter introduces a value to the diagonal of which makes it non-singular.

The mean squared error () and score are common metrics for measuring the similarities of the predicted- and true model. The *MSE* is defined as

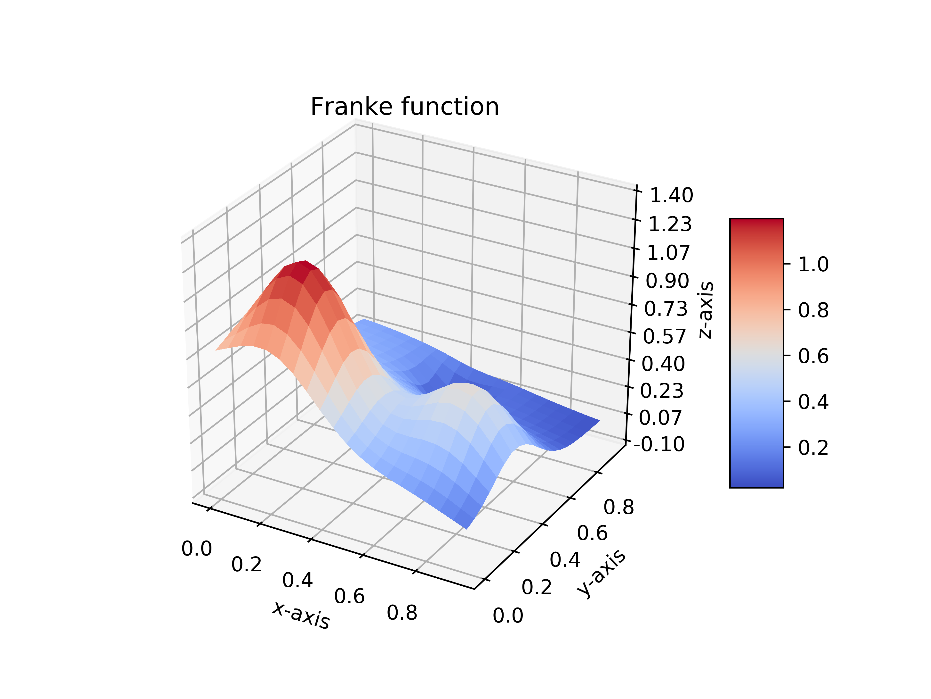
and the R2 score as

where  **is the true model, is the predicted model and is the mean of the true model. However, this relates to prediction error by only assessing training data directly, which is not a good measure for model assessment in machine learning problems (Hastie et al, ).**

**Instead, we can use resampling methods in order to have a better model assessment. A common method for model assessment in machine learning is the bootstrap method, which is a common tool for assessing machine learning models. Suppose now that we split the jointly distributed set of measurements into a training set and an independent test set. The basic idea in the bootstrap is to randomly draw samples from your training set with replacement (i.e. the same sample can be drawn more than once), let’s say times, and refit the model for each bootstrap sample, and then apply the same set of predictors to test set. The error function reads**

where , i.e. the squared error between the true model and the predicted model for the bth bootstrap. **We then assess the model performance by computing the average of the expected prediction error on both the training set and the test set. The underlying concept for training data splitting is that the prediction error consists of a tradeoff between the variance and the bias of the predicted model (and variance of the data which is irreducible), which shows an increase in prediction error for the test set towards higher model complexity. This is referred to as overfitting in machine learning, i.e. your model is not able to generalize in a good manner. The variance bias tradeoff can be derived from expected prediction error**

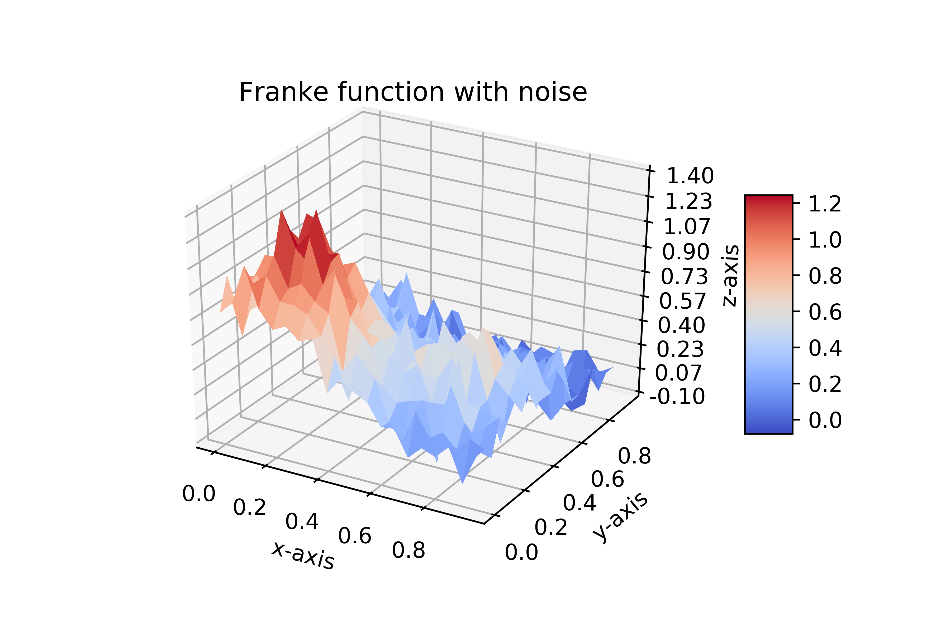
where we have the following relationships

**Results and discussion**

**In this section we apply the methods, as explained above in the theory section, on both synthetic- and real topographic terrain data from the southern part of Norway.**

**In order to test and evaluate parameterization of a real topographic surface using different methods of least squares, i.e. OLS, Ridge and Lasso, we apply the methods on a well-known function known as “Franke’s function”, which will be our synthetic topographic data in our testing. Franke’s function is displayed in Figure 1, where it is defined in the range** , with a spatial sampling of  **Franke’s function is known as a testing/evaluation function in interpolation problems. It is a sum of four exponentials and reads as follows**

**Figure 1:** Franke’s function where we defined the function at and spatial sampling of .



Before we apply the methods, we introduce stochastic noise to Franke’s function, with a normal distribution and a magnitude of 10% of . The result of Franke’s function with added noise is displayed in Figure 2.

**The algorithms of OLS and Ridge where written in python (see Appendix A). In the case of prediction using Lasso we chose to use the scikit-learn module since it don’t have an explicit formulation of the solution to , and therefore requires an iterative optimization algorithm in order to compute to best fit.**

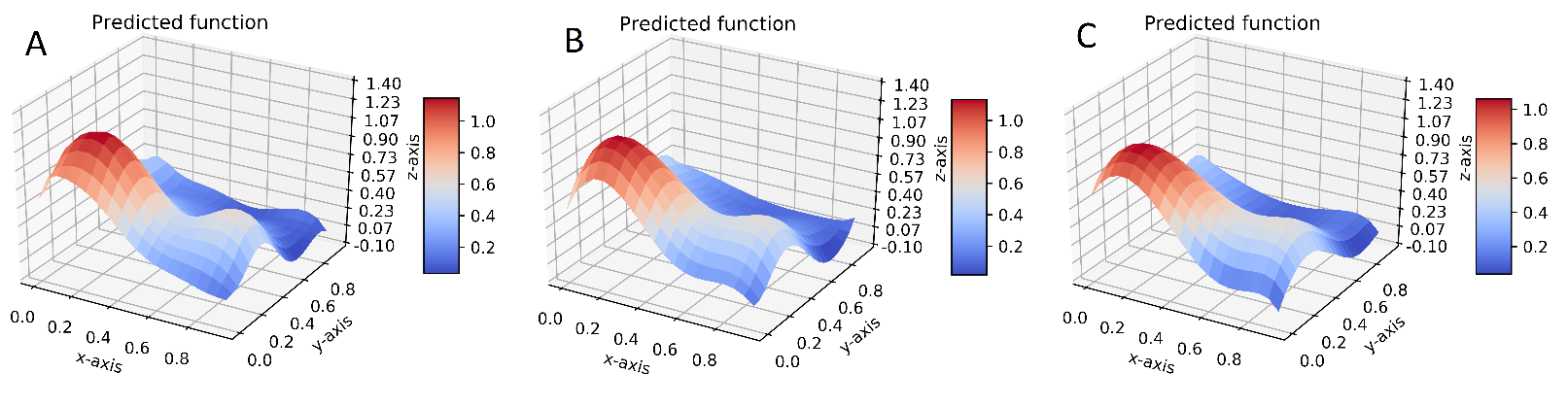
**Figure 2:** Franke’s function with stochastic noise. The noise is normal distributed and with magnitude of 10% of .

**The prediction scheme was setup in three steps:**

1. **Define our training set as where ;** ; **.**
2. **The design matrix was generated by flattening and into vectors of shape with polynomial degree up to eight order and with and dependency. The design matrix would read**

**In this case, a first order prediction, the design matrix would be the three first columns of , second order will be the first six columns of , and so on. With a design matrix up to eight order, the parameter will be of size including the bias term.**

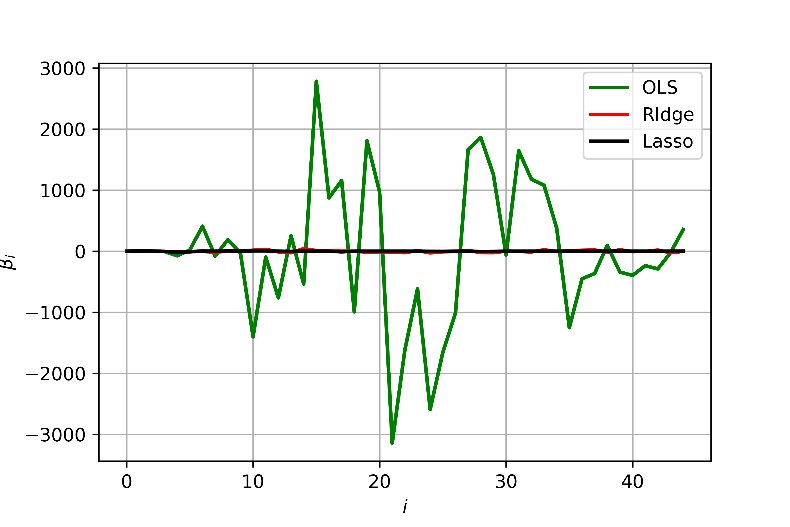
1. **Flatten the two dimensional function to a vector of shape . Predict the data** using the estimated parameters from either OLS, Ridge or Lasso.

**The predictions of Franke’s function using OLS, Ridge and Lasso are displayed in Figure 3. For the Ridge and Lasso predictions we used a penalty parameter of . Visually, the OLS prediction is slightly more satisfactory than the other two predictions (in this particular case), which tends towards smoother surfaces. The improved prediction result from OLS compared to Ridge and Lasso are supported by the MSE and R2 score for the different predictions, which are presented in table 1, and shows that the OLS prediction is closer to true result, than for Ridge and Lasso. However, the differences are not substantial.**

**Figure 3:** Prediction of the Franke’s function using OLS solution (A), Ridge (B) and Lasso (C) on the noisy data displayed in Figure 2. The design matrix was set up to eight order polynomial fit including and dependency.The penalty parameter for Ridge and Lasso was set to .

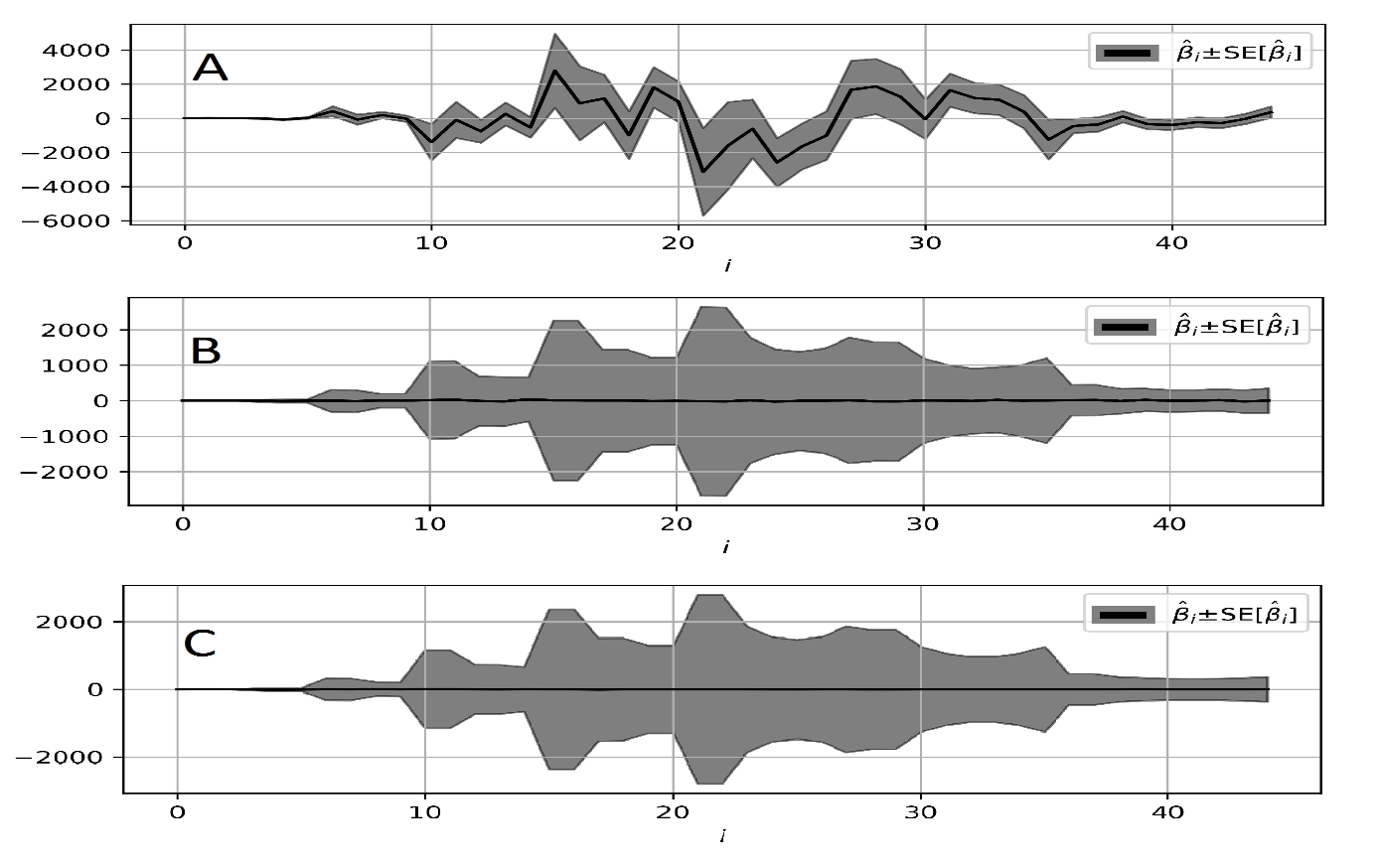
**Table 1:** MSE score and R2 score from the OLS, Ridge and Lasso predictions of the function displayed in Figure 2, using a design matrix up to eight order. Ridge and Lasso methods was computed using a penalty of . The MSE and R2 score shows that the predicted response from OLS lies closer to the true response than Ridge and Lasso does for this particular case.

|  |  |  |
| --- | --- | --- |
| **Method** | **MSE score** | **R2 score** |
| **OLS** |  |  |
| **Ridge** |  |  |
| **Lasso** |  |  |

**So, what lies behind the choice of the**  parameter value**? For a large parameter, say** , the regression fit using Ridge and Lasso closely resembles an two dimensional plane. This indicates that we are shrinking the parameters corresponding to second order polynomial and higher, towards zero, and as we lower we eventually obtain the solution from OLS. We observe in Figure 4, that even while using a penalty parameter of, most of the parameters are greatly dampened, but does not, however, greatly decrease the performance in the prediction as we observe in Figure 3 and Table 1.

The minimal decrease in performance Ridge and Lasso solution compared to OLS, even with a greatly dampening of the parameters, could be due to the parameters from Ridge and Lasso are still within the parameters confidence interval of the OLS solution, as observed in Figure 5.

**Figure 4:** Comparison of the parameters from OLS, Ridge and Lasso. We can see that the parameters are greatly dampened by using Ridge and Lasso regression, but does not greatly decrease the performance as we see from the prediction result in Figure 3 and the MSE and R2 score in Table 1.

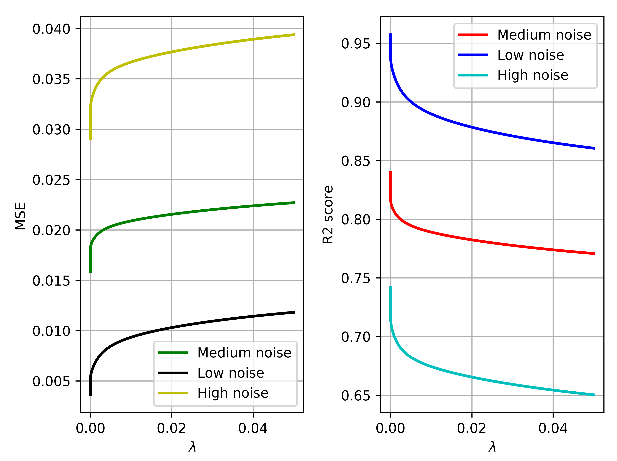
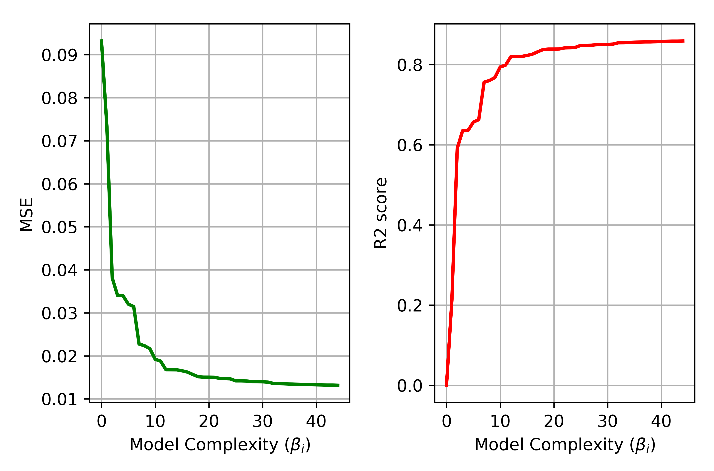
****

**Figure 5:** The parameters from OLS (A), Ridge (B) and Lasso (C) and their corresponding confidence interval. We observe from all regression results that the confidence intervals increases for parameters corresponding to higher order, and decreases around which corresponds to a polynomial fit of about 5th order. In addition, the parameters from Ridge and Lasso more or less lies within the confidence interval of the OLS.

**We can now ask ourselves:**

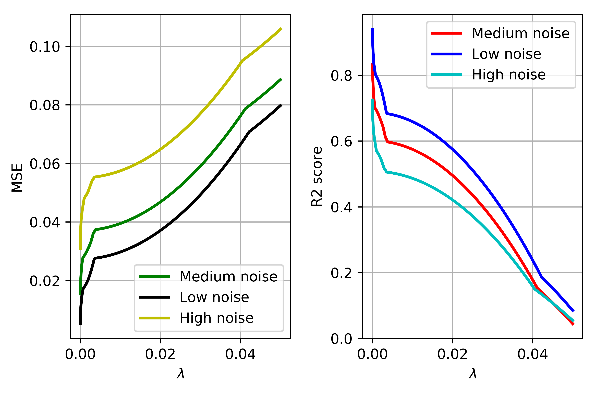
1. **Do we increase performance from using a higher order polynomial fit?**
2. **How do the predictions from OLS, Ridge and Lasso fit to other data?**

**By computing MSE- and R2 score wrt model complexity, we observe that increasing the number of parameters increases the prediction performance on the training data. This is displayed in Figure 6, where we have computed the MSE and R2 score of the OLS solution wrt to increasing order of polynomial fit.**

****

**Figure 6:** Computed MSE and R2 score wrt , using Ridge regression. High, medium and low noise corresponds to a magnitude of 15%, 10% and 5% of . The prediction error decreases, and model performance increases with lower values. However, since this is only wrt training data this is not a good assessment of model performance.

**Figure 7:** Computed MSE and R2 score wrt model complexity, i.e. order of polynomial fit within the prediction, using OLS. The prediction error decreases, and model performance increases as higher order polynomials are introduced to the model. However, prediction error using only training data is not a good assessment of model performance.

****In order to assess our model performance in a more correct manner, **we introduce a widely used method for model assessment in machine learning (and statistical learning), known as the k-fold cross validation algorithm. From the theory part we know that the prediction error is a tradeoff between the models irreducible error, bias and variance. As we increase the model complexity by introducing higher order polynomials we reduce the bias and increase the variance of the training set.**

**Figure 8:** Computed MSE and R2 score wrt , using Lasso regression. High, medium and low noise corresponds to a magnitude of 15%, 10% and 5% of . The prediction error decreases, and model performance increases with lower values. However, since this is only wrt training data this is not a good assessment of model performance.

**Figure 9:** Computed prediction error wrt to model complexity using k-fold cross validation. While only considering the training set, the prediction error decreases as model complexity increases. By considering the test set, the prediction error increases for . This is due to the bias and variance

**Parameterization of topographic data from southern Norway**

Appendix A