Project 1 on Machine Learning

Maziar Kosarifar

October 2019

Abstract

In this project we aim to evaluate and study different features, and properties of three different regression methods including Ordinary Least Squares(OLS), Ridge regression and Lasso regression. And the application of resampling techniques. And we compare their result for Franke function and then introduce real terrain data, as inputs.

1 Introduction

Regression methods are widely used to study the relation between two variables, and estimate if and how they are related, and understand whether there is a close numerical correlation between variables. Ordinary Least Squares, Ridge regression and Lasso regression are the one implemented in this project.

In this paper I will briefly explain each one of them. Then I'll introduce the use of statistical methods to evaluate each of these regression techniques.

In the last part we will introduce real world data of geographical terrains as the input, to evaluate the differences between these methods.

2 Regression models

2.1 Ordinary Least Squares

Ordinary Least Squares are one of the methods used in approximation theory, which seeks to find an approximate curve that would minimize the error norm, when given a set of real values such as f_0, f_1, ..., f_n at real data points X_0, X_1, ..., X_n which we call nodes.

So we want to find a polynomial P(X):

$$P(x, y) = \Sigma_{i=0}^{n} \beta_i x^i y^{n-i}$$

to approximate the value of F(x, y).

This problem can be turned into:

$$y = X\beta + \epsilon \tag{1}$$

This can be turned into a problem of minimizing the following value of S

$$S = ||X\beta - y||_2^2 \tag{2}$$

Which is a set of normal equations, and the minimum value will be found when the gradient of S is zero, so we have:

$$X^T X\beta = X^T y$$

Which can be solved with:

$$\beta = (X^T X)^{-1} X^T y \tag{3}$$

```
def OLS(X, expected_value):
    beta = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(expected_value)
    return beta
```

2.2 Ridge regression

OLS treats all variables as equals, and considers that the result equally depends on all of them. Ridge regression on the other hand introduces λ to add some bias to the estimation.

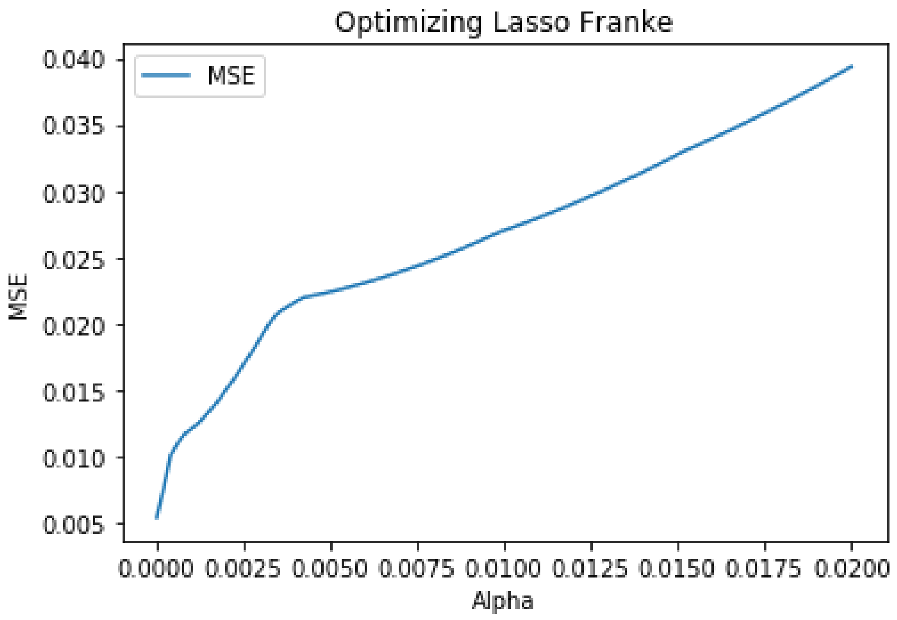
$$\beta^{Ridge} = (X^T X + \lambda I)^{-1} X^T y \tag{4}$$

```python
def OLS(X, expected_value):
    beta = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(expected_value)
    return beta
```

2.3 Lasso regression

Lasso (Least Absolute Shrinkage and Selection Operator) is a method used in regression analysis, to find the features relevant to the approximation, and eliminate the rest, by a parameter called α.

The parameter α here is the rate in which the elements get examined. Therefore it is important to find an α that suits the study case(For Franke function).



Here is a plot of studying the effect of different α values over different polynomial degrees, and the resulting MSE.

Since implementing Lasso regression is not as straight forward as the earlier methods, I have used the implementation available in Sklearn library.

3 Cross validation

Cross validation methods are a set of methods used to split the given data into primary two sets, training set, and testing set. The training set is used for the regression analysis and to train our parameters, and the testing set is used to study the effectiveness of our approximation result.

I have used two different methods to split the data, first the Sklearn method of "train-test-split", and k-fold cross validation.

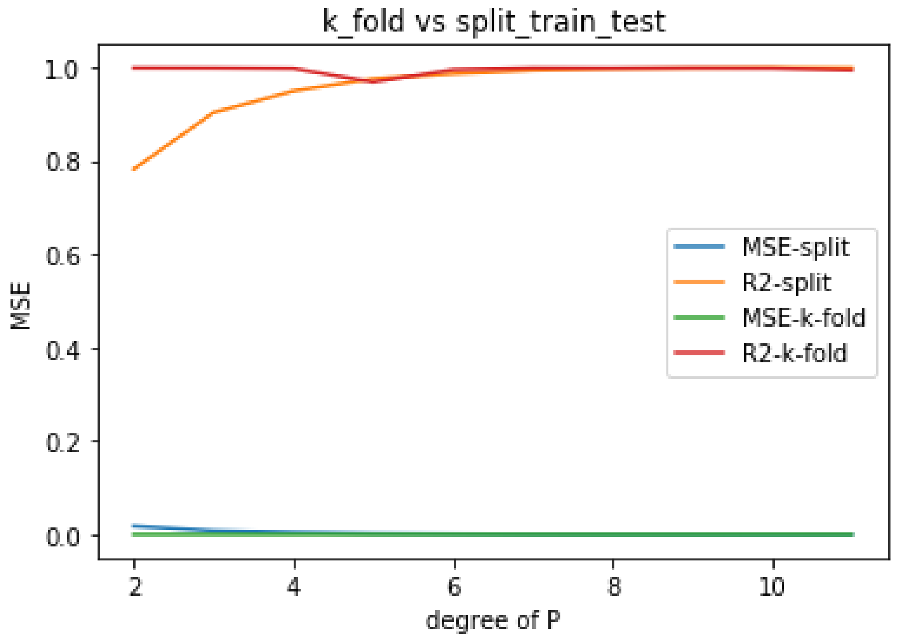
The train-test-split simply just divides the data set into two data sets. I have used a training portion of $\frac{3}{4}$ which leaves fourth of data for testing set.

K-fold cross validation, will divide the data set into k different sets. It will run the learning process k times, and each time, the i-th section is used for testing and the rest are the learning material. This can result in a slower learning, but it can produce a better set of results.

```python
def k_fold_split(self, k):
    n = len(self.x)
    print ("length of the whole data ", n)
    # making random numbers
    arr = np.arange(n)
    # making the set even
    np.random.shuffle(arr)
    print(len(arr))
    # make the lenght dividable by k:
    print(len(arr))
    k_fold = []
    for i in range(0, int(n/k)*k, int(n/k)):
        print (i, i + int(n/k))
        k_fold.append([])
        k_fold[-1] = arr[i:i + int(n/k)]
        print (len(k_fold[-1]))

    return k_fold# -*- coding: utf-8 -*-
```

Here we can see that k-fold cross validation although would take longer than ”train-test-split” method, since it has to run the program k times, but it would produce a better, more stable result.



4 Test materials

Here we test our regression models using two set of data, Franke function and digital terrain data obtained through earth explorer.

In each dataset first I have tried to optimize the λ, and α values for Ridge, and Lasso regression, and then compare their results to figure out which model works best.

4.1 Franke function

Franke function is a widely used as a test function for methods in regression analysis.

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

The Franke function is defined for $x, y \in [0, 1]$.

4.2 Terrain data

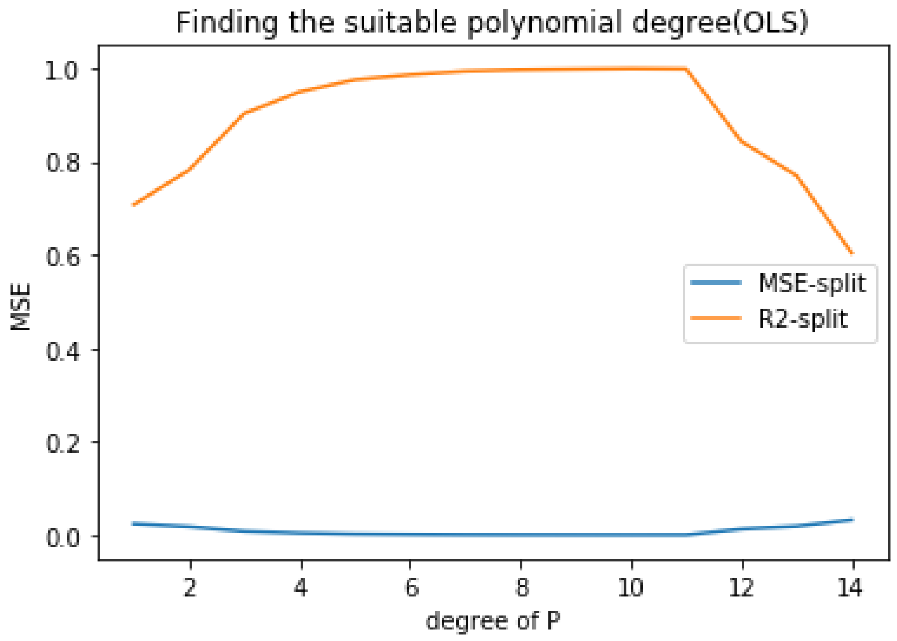
There is a big set of terrain data available through https://earthexplorer.usgs.gov, but I have only used the terrain data given from Norway.

5 Regression Analysis

First I attempt to find the best polynomial degree that would avoid over-fitting, and then optimize the value for α, and λ, for Lasso, and Ridge regression.

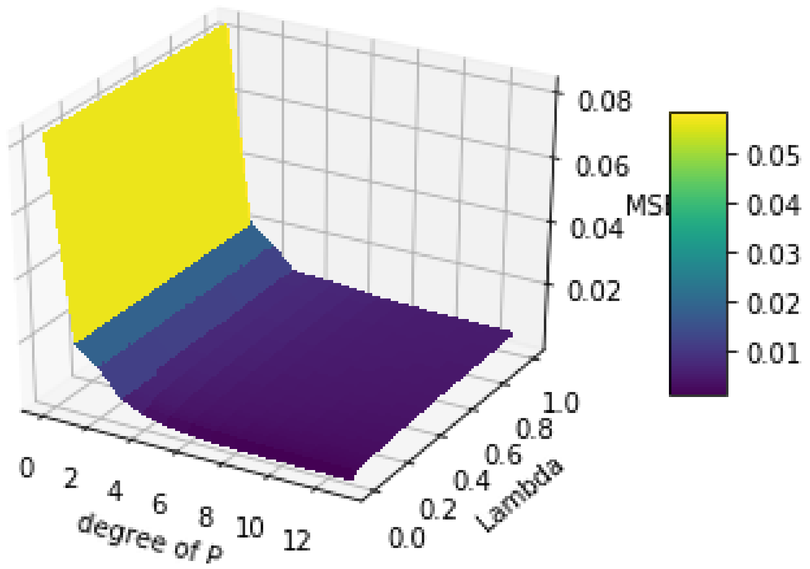
5.1 Franke function

To find the best polynomial degree for Franke Function, I have used OLS, as the method amongs the chosen three that is most susceptible to over-fitting, and plotted the MSE, and R2 score, for different degrees of the polynomial.



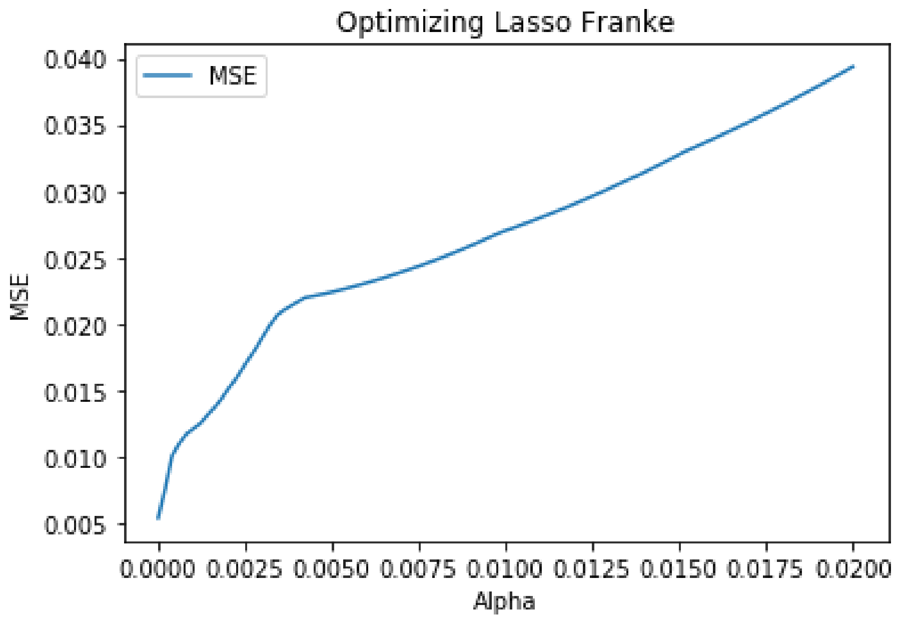
And realized that $n = 11$ produces the best result.

Running the same analysis for Ridge regression, to find the best value for λ over different set of values for n produced the plot:



Showing that Ridge regression also results improve when n is approaching 11, and the $\lambda = 0.2$

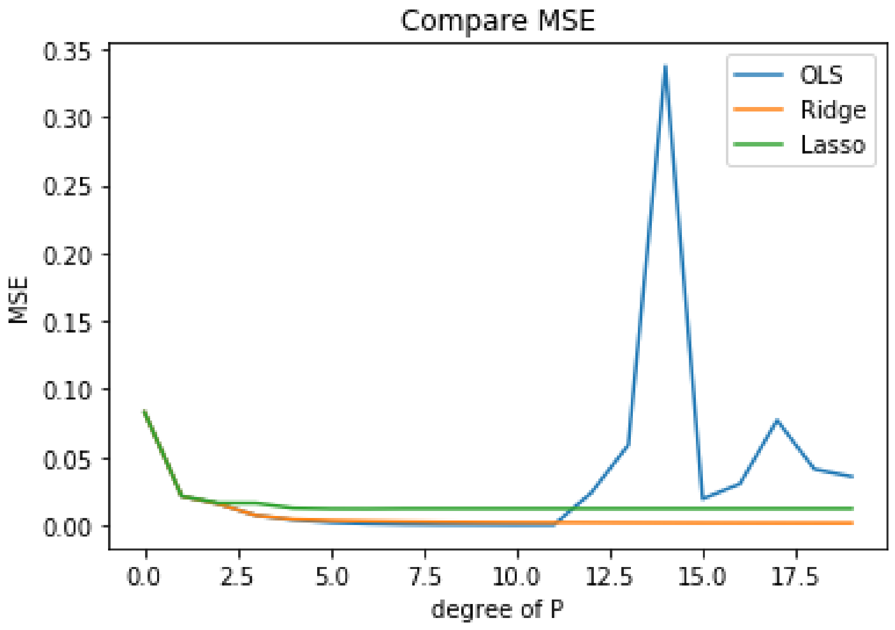
I attempt to run the same analysis for Lasso, but since Lasso regression is generally a slower process, I have only tried to optimize the α variable for n = 11.



Which shows that Lasso regression results in the best MSE values when α

approaches zero. Which can be interpreted as Lasso regression not being able to use it is full potential for Franke function.

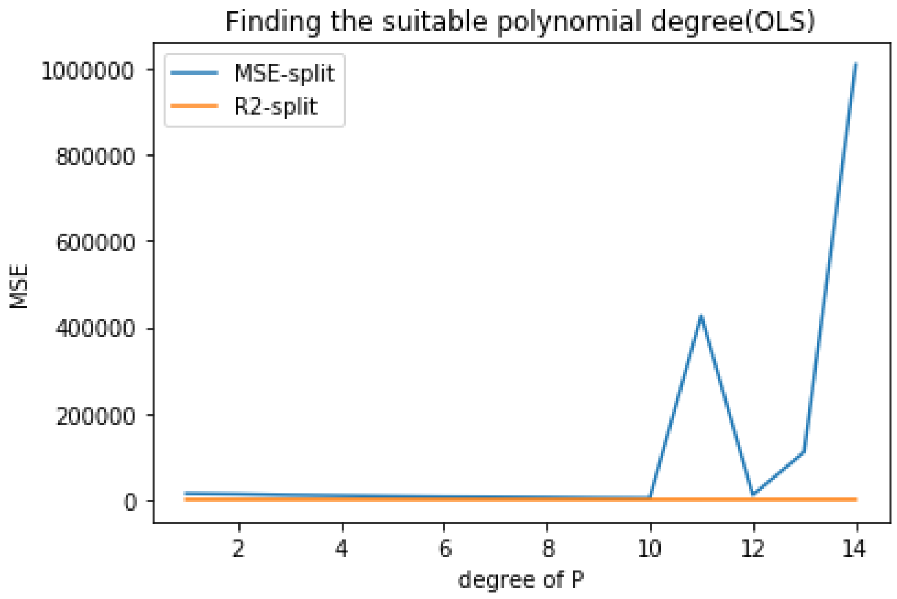
Finally comparing the MSE for different values of n, for all three methods, when all their variables are optimized. $\lambda = 0.02$ for Ridge Regression $\alpha = 0.0001$ for Lasso Regression



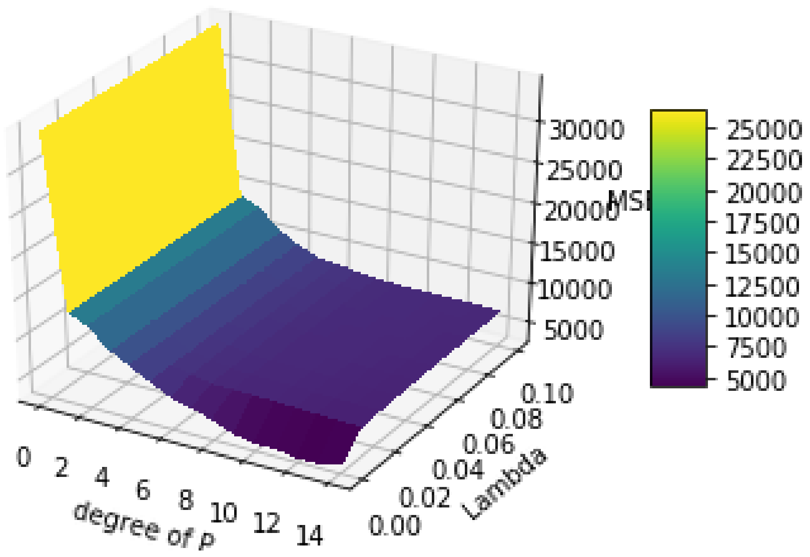
This plot shows that OLS produces the best results for Franke function test case, but is more prone to over-fitting. While Ridge and Lasso regression keep producing more stable results.

5.2 Terrain data

For terrain data I have chosen a bigger set of inputs, 800, by 400. Running the OLS, for different polynomial degrees resulted in the following plot:

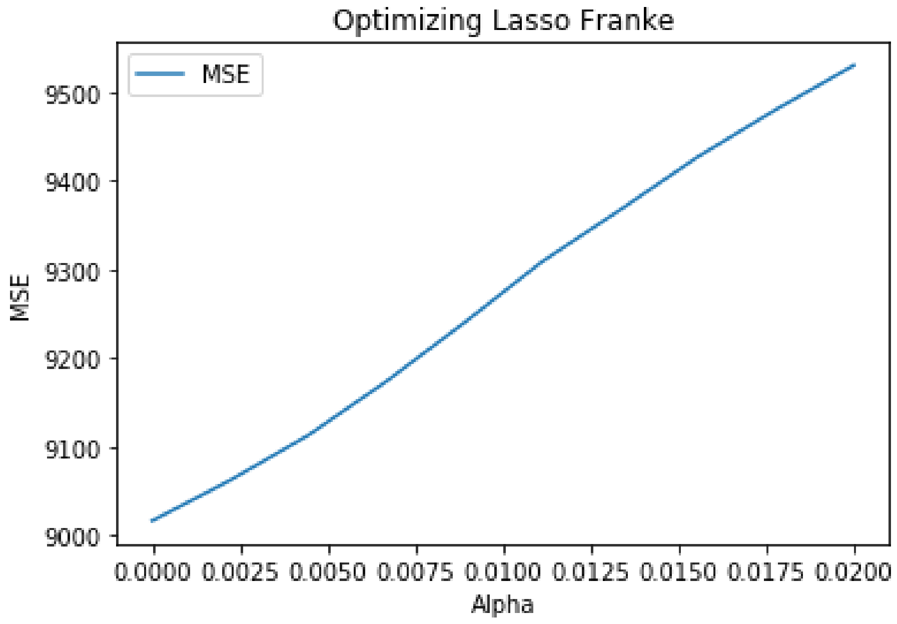


This plot shows that the best suitable polynomial degree for OLS, is 10. Now trying to optimize the λ value for Ridge regression for lambda:

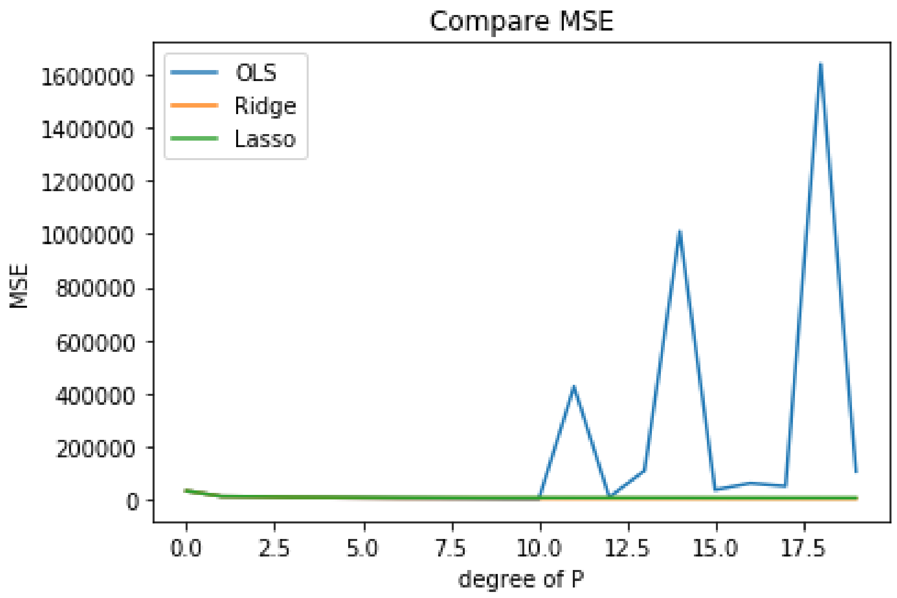


This plot shows that Ridge regression also produces it is best around polynomial degrees 10, with lambda values approaching 0.002.

Running the same analysis this time for Lasso to find the optimized α value, here again, I've set the polynomial degree to be 10, and only test Lasso regression for various values of α, since for this test case I had to increase the max number of iterations for Lasso regression.



Now using the optimized values for both Lasso and Ridge regression, we can compare their results for test group in terrain data.


Compare MSE
OLS
Ridge
Lasso
MSE
degree of P
1600000
1400000
1200000
1000000
800000
600000
400000
200000
0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

6 Conclusion

In this report we have implemented, and studied the use and results of Ordinary Least Squares, Ridge and Lasso regression over two different test cases.

It shows that for test cases that follow a well, mathematically expressed data, OLS, is both fast, and reliable (Franke function), and doens't need to be optimized for its parameters. But for more complicated test cases, like study of terrains, Lasso, and Ridge can produce better results, which are more stable. Optimizing Lasso, though time consuming can result a better outcome. Therefore it is important to choose the method of regression based on the learning material.

7 Bibliography

[1] Walpole, R. E., Myers, R. H., Myers, S. L. Ye, K. (2007) *Probability & Statistics for Engineers & Scientists* (8th ed.). New Jersey: Pearson Education, Inc.
[2] Smith, R. C. (2014) *Uncertainty Quantification: Theory, Implementation, and Applications.* Philadelphia: Society for Industrial and Applied Mathematics.
[3] Faul, A. C. (2016) *A Concise Introduction to Numerical Analysis.* New York: Chapman and Hall/CRC.