Project 1 Report

For the project I used Anaconda environment on Windows and created the project using Python. I also used Pycharm IDE in parallel for better execution and understanding. I used Python for the project as Python has many incredible packages like **Pandas, NumPy, SciPy, Scikit Learn, Matplotlib** that are very much aligned to the project and machine learning works. Initially I started with Glment library as suggested in Lab document but could not work with it as it had very limited support for Windows environment. I got the error as

```
 ValueError: loadGlmlib does not currently work for windows
```

So I switched to sklearn and used Ridge, Lasso, LassoCV etc from sklearn. Python also has functions for almost any statistical operation / model building that we may want to do. I had worked with Python earlier and hence found it easier to go with it.
What I found particularly interesting with python were its packages. Scikit-learn is the most popular machine learning library for Python. Built on NumPy and SciPy, scikit-learn offers tools for data mining and analysis that bolster Python's already-superlative machine learning usability. NumPy and SciPy impress on their own. They are the core of data analysis in Python and any serious data analyst is likely using them raw, without higher-level packages on top, but scikit-learn pulls them together in a machine learning library with a lower barrier to entry.[1]

[1] https://opensource.com/article/16/11/python-vs-r-machine-learning-data-analysis

The project execution and details of steps carried out are as follows.

Task 0:

i) Import the necessary libraries like numpy, pandas.

ii) Import the data csv

iii) Created the response and feature variables, i.e., X and y.

iv) Randomly selected and split data using sklearn.model_selection.train_test_split to have X_train, X_test, y_train and y_test

v) This X_train and y_train is sent for the 5-fold CV in Task 1.

Task 1:

i) In a loop executing 10 times, perform the following:
# for Linear Regression

i. Split the data into X_train, X_test, y_train and y_test

ii. In next loop executing 5 times, perform the 5-fold CV on X_train, y_train as follows:

i. Split X_train into 5 parts and treat 4 of them as train and remaining as validate.

ii. Create and fit a linear model

iii. Calculate MSE and R^2 for this Linear model

iv. Save the MSE and X_train, y_train generated in the current CV iteration

iii. For the X_train, y_train that has minimum MSE, create and fit the linear model.

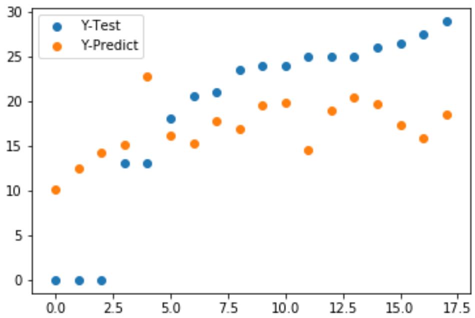
iv. After fitting the model, predict the values of X_test

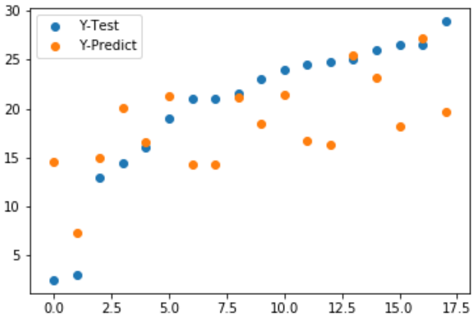
v. Calculate the MSE and R^2 using the predicted y values and y_test.

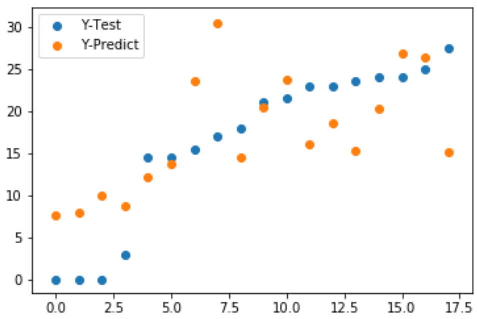
vi. Plot the graph between y_predict and y_test

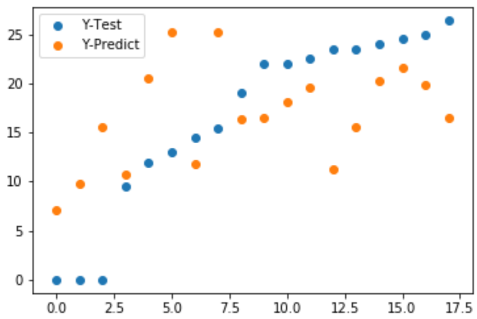
Calculate the mean of MSE and R^2 obtained in 10 iterations.

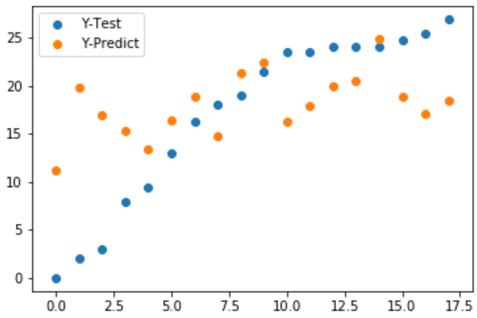
The graphs between y_predict and y_test for each iterations are as follows:

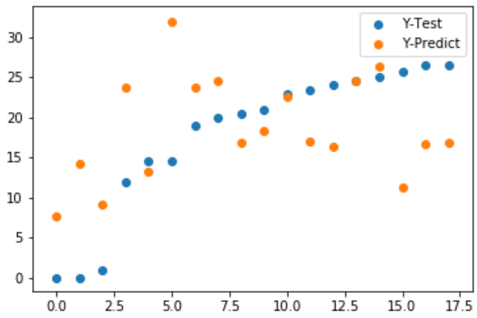

Y-Test
Y-Predict
0
5
10
15
20
25
30
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

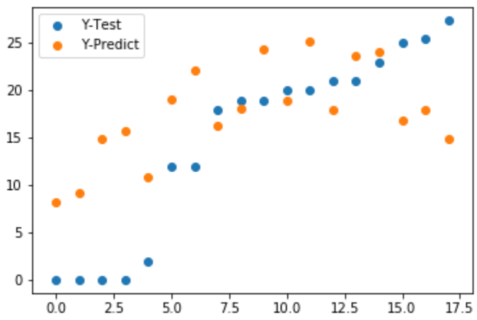

Y-Test
Y-Predict
5
10
15
20
25
30
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

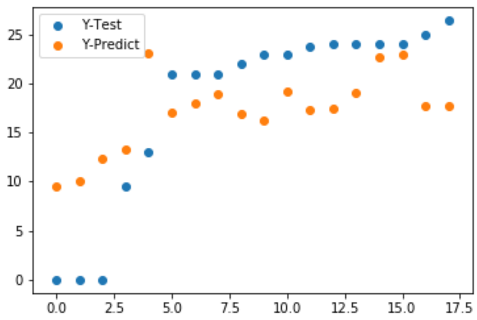

Y-Test
Y-Predict
0
5
10
15
20
25
30
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

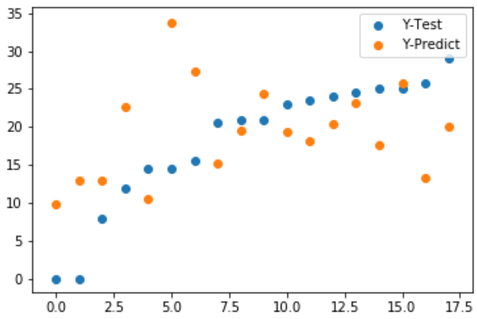

Y-Test
Y-Predict
0
5
10
15
20
25
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

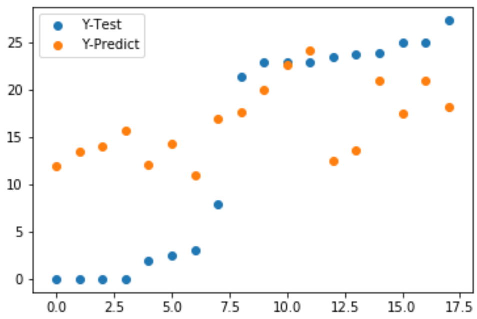

Y-Test
Y-Predict
0
5
10
15
20
25
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


Y-Test
Y-Predict
0
5
10
15
20
25
30
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


Y-Test
Y-Predict
0
5
10
15
20
25
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


Y-Test
Y-Predict
0
5
10
15
20
25
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


Y-Test
Y-Predict
0
5
10
15
20
25
30
35
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


Y-Test
Y-Predict
0
5
10
15
20
25
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

ii) Repeat the same for Lasso with varying alpha values but calculating only MSE.
iii) Repeat the same for Ridge with varying alpha values but calculating only MSE.

Result:

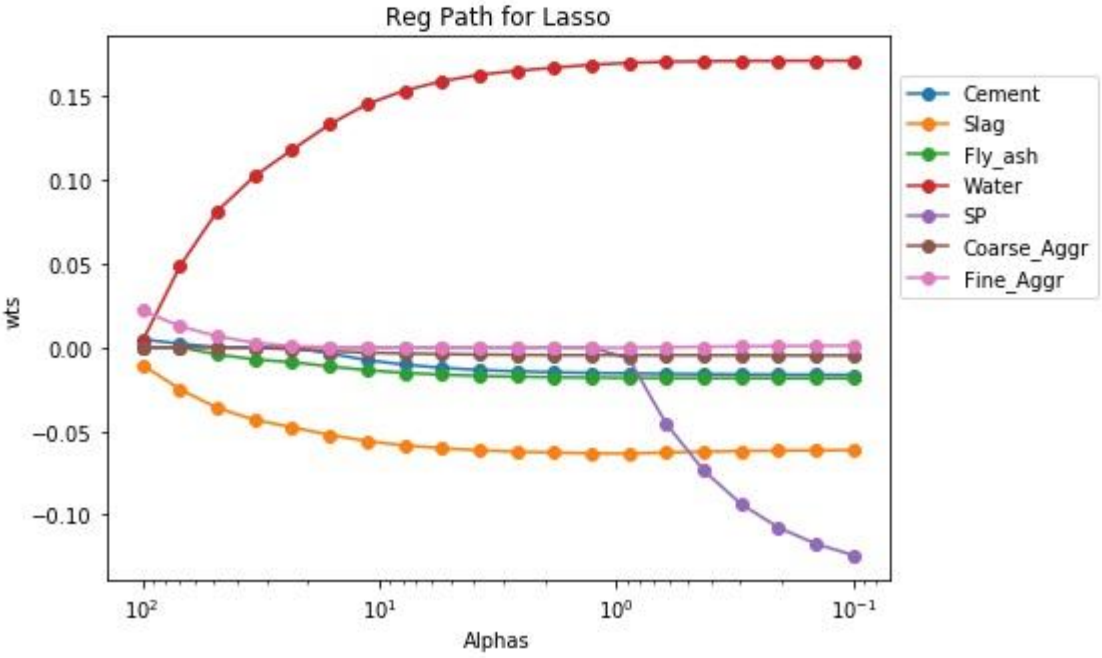
I obtained the following values averaged over 10 iterations.

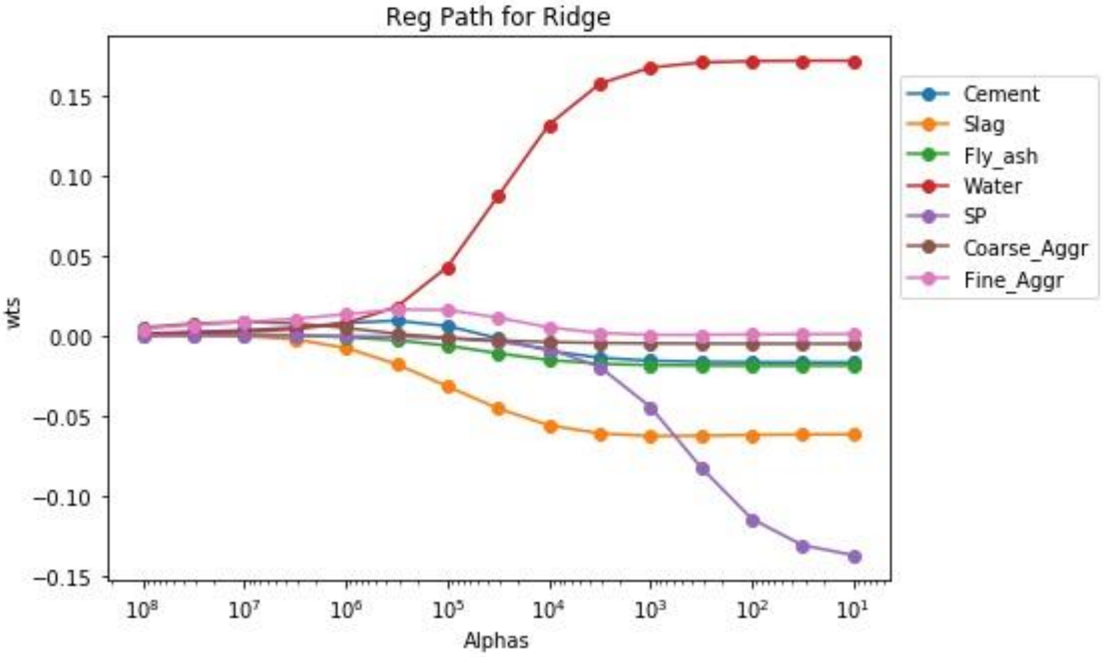
R^2 mean value: 0.256946915347
MSE mean Value Unregularized Regression: 53.8363977613
MSE mean Value Lasso: 55.8374792321
Coefs.that produces the minimum error: [0.01473612,-0.,0.01685403,0.29139523,-0.31708469,0.03158856,0.03774138]
Corresponding MSE: 32.3097516355

MSE mean Value Ridge: 53.7500712703
Coef.that produces the minimum error: [0.042488, 0.04060353, 0.04539437,0.37747263, -0.33322708,0.06493326,0.07407684]
Corresponding MSE: 32.4892531383

Task 2: Regularization path graphs for ridge and lasso

For the coefficients as obtained from above loops and the varying alpha, we plotted the regularization path as shown below:





Resources:
I spent around 4-5 days on the project. Initially, I went through the concepts to understand the requirements. The requirements were a bit unclear but thanks to Prof. Kevin and TA Wang, for clarifying the same over piazza and during office hours.

The list of resources I used are as below:
Environment: Windows
IDE: Pycharm
Notebook: Jupyter IPython
Python Libraries: Numpy, Pandas, Sklearn, Matplotlib
Textbook: Machine Learning: A Probabilistic Approach, Murphy
Online resources:
https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6
https://matplotlib.org/users/pyplot_tutorial.html
https://github.com/probml/pyprobml
http://www.ritchieng.com/machine-learning-evaluate-linear-regression-model/
https://stackoverflow.com/questions/4700614/how-to-put-the-legend-out-of-the-plot/
http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html
http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LassoCV.html
http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lars.html
http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html
http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.RidgeCV.html
http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.lars_path.html
http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.lasso_path.html
http://scikit-learn.org/stable/auto_examples/linear_model/plot_lasso_model_selection.html
https://plot.ly/matplotlib/scatter/
https://matplotlib.org/api/legend_api.html
https://matplotlib.org/api/markers_api.html
https://docs.scipy.org/doc/numpy-1.8.1/reference/generated/numpy.sum.html
https://docs.scipy.org/doc/numpy-1.8.1/reference/generated/numpy.mean.html
https://www4.stat.ncsu.edu/~post/josh/LASSO_Ridge_Elastic_Net_-_Examples.html
https://stackoverflow.com/questions/49247396/python-matplotlib-how-to-use-non-gui-and-gui-backend-in-one-program
https://stackoverflow.com/questions/11656767/how-to-take-the-log-of-all-elements-of-a-list
https://plot.ly/scikit-learn/plot-ridge-path/

And of course: https://piazza.com/class/jcnwfv8i2rp1a