Programming Assignment II

Shaun Pritchard & Isammel Lopez

Florida Atlantic University

CAP 5625

November 08, 2021

M.DeGiorgio

Programming Assignment II

The scope of this project was to build Elastic Net penalized regression with coordinate gradient descent algorithm from scratch and with libraries(for comparison) while implementing a grid based search 5K-fold Cross Validation to test the effect of the tuning parameters on the inferred regression coefficients.

For this assignment II we both started out implementing the projects independently, then we collaborated to finalize the assignment deliverables. This resulted in the completion of 2 different methods for creating the assignment 2 algorithms from scratch (Primary & Secondary) namely implementing the same assignment parameters of Elastic Net with Coordinate descent on 5 K-folds penalized regression. Each instance also has a full with labaries implementation in the code file. The aim was to display different methods of achieving the same abstraction. To complete both instances of the project, we implemented the following algorithm.

Assignment 2 - Algorithm 1 (vectorized):

Step 1. Fix tuning parameters λ and α

Step 2. Generate N-dimensional centered response vector $\mathbf{y}$ and N×p standardized

(centered and scaled to have unit standard deviation) design matrix $\mathbf{X}$

Step 3. Precompute bk,k =1,2,...,p, as

$$b_k = \sum_{i=1}^{N} x_{ik}^2$$

Step 4. Randomly initialize the parameter vector β =[β1,β2,...,βp]

Step 5. For each k, k =1,2,...,p:

compute

ak = xkT($\mathbf{y}$–$\mathbf{X}\beta$ + xk βk)

and set

βk = sign(ak)($|ak|-\lambda(1-\alpha)$ 2)+ bk +$\lambda\alpha$

Step 6. Repeat Step 5 for 1000 iterations or until convergence (vector β does not change)

Step 7. Set the last updated parameter vector as β =[β1,β2,...,βp]

Deliverables:

This project required six deliverables produced from the algorithms I created using the python programming language. The deliverables are as follows.

- Deliverable 1: Illustrate the effect of the tuning parameter on the inferred elastic net
- regression coefficients by generating six plots
- Deliverable 2: Illustrate the effect of the tuning parameters on the cross validation error by generating a plot of six lines (one for each α value) with the y-axis as CV(5) error, and the x-axis the corresponding log-scaled tuning parameter value log10(λ).
- Deliverable 3: Indicate the pair of values λ and α that generated the smallest CV(5) error.
- Deliverable 4: Given the optimal λ and α pair, retrain your model on the entire dataset of N=400 observations and provide the estimates of the p=9 best-fit model parameters.

For simplicity we have delivered two implementations of assignment listed as *Primary* and *Secondary* the following is an overview of the instructions and deliverables for the assignment 2 project.

Primary - shows a combination of algorithms implemented from scratch combined with the libraries for comparison.

Secondary - Shows the implementation of assignment 2 from scratch and then assignment 2 with libraries for comparison .

Instructions:

The algorithms in this assignment were created using python and Google Colaboratory Jupyter notebooks. I created two separate notebooks, one for Primary assignment 2 and one for Secondary assignment 2. This report will show the comparisons of the algorithms I developed from scratch with the python scikit learn library implementations. The contents of this assignment 2 including this report and as per deliverables required and produced are as follows: separate Jupyter notebook files (which can be ran with interpreter online or offline), a PDF print out for each containing the working code, results, and testing output, and separate raw python (.py) script files.

All files are included and are also both implementations are accessible online for viewing and commenting at the following URLs:

Primary Google Colab implementation:

https://colab.research.google.com/drive/1ur5u_FGm5g4PkkwdpOazER2h3G67Rg5t?usp=sharing

Secondary Google Colab Implementation:

https://colab.research.google.com/drive/1o6PXiGmn3MCE20Ir3h2hhIZGwo0u2KeJ?usp=sharing

Note: For using standard python scripts on a local machine make sure the required imports are installed using pip, pip3, or anaconda. Also, rename the file-path if importing the dataset from another location other than the original implementation.

Files and directories:

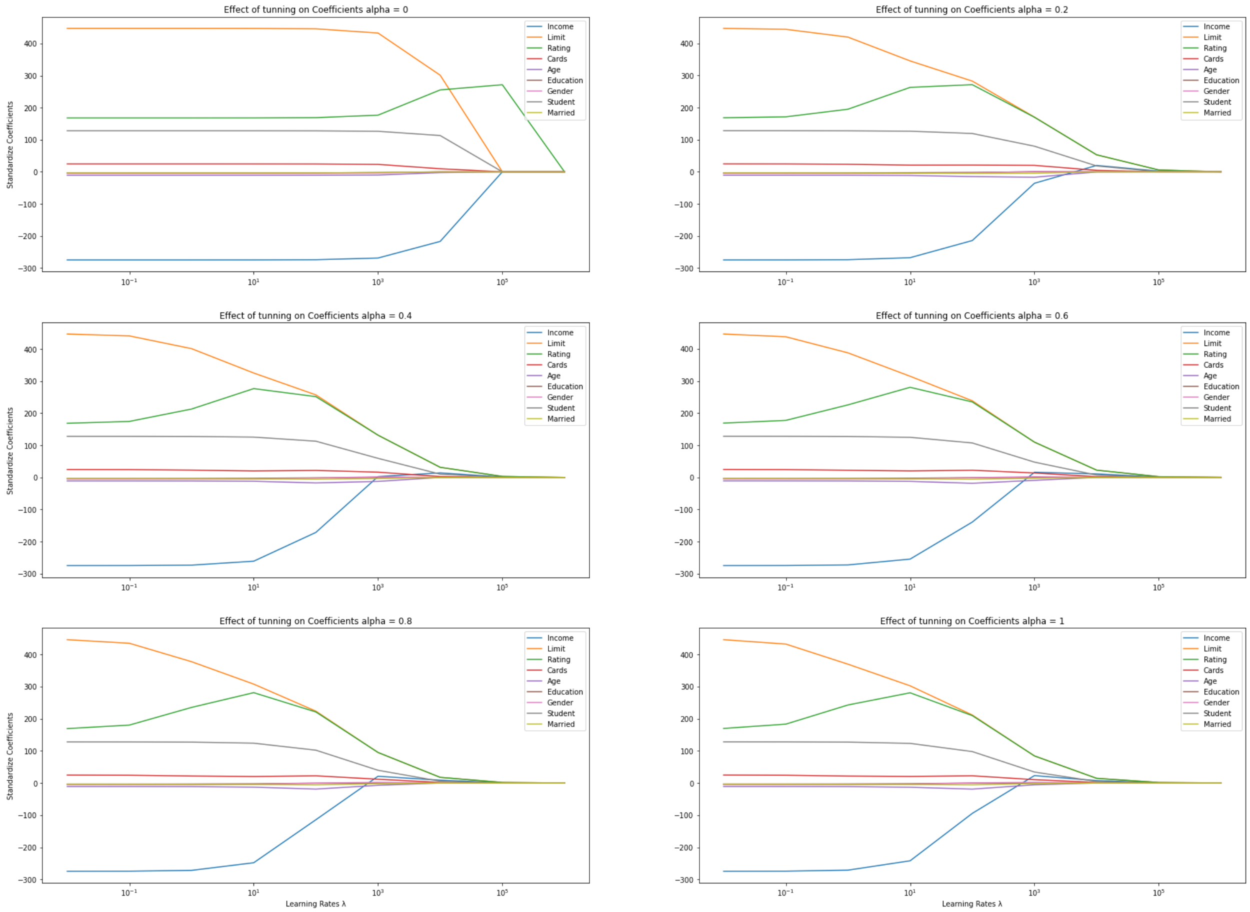
- Assignment 2 Instructions := instructions as per assignment for CAP 5625 Assignment 2
- Dataset := contains the working dataset for this project
- Primary Implementation directory := PDF of code and results, ipynb file of code, and .py script file of code, for deliverables specific to the code used with libraries only.
- Secondary Implementation directory := PDF of code and results, ipynb file of code, and .py script file of code, for deliverables specific to the code used with libraries only.
- Instructions and comparison file : = this file in word and pdf format

Note: Devliberable numbering below does not directly correspond to the actual primary and secondary implementation code cells for the assignment we produced.

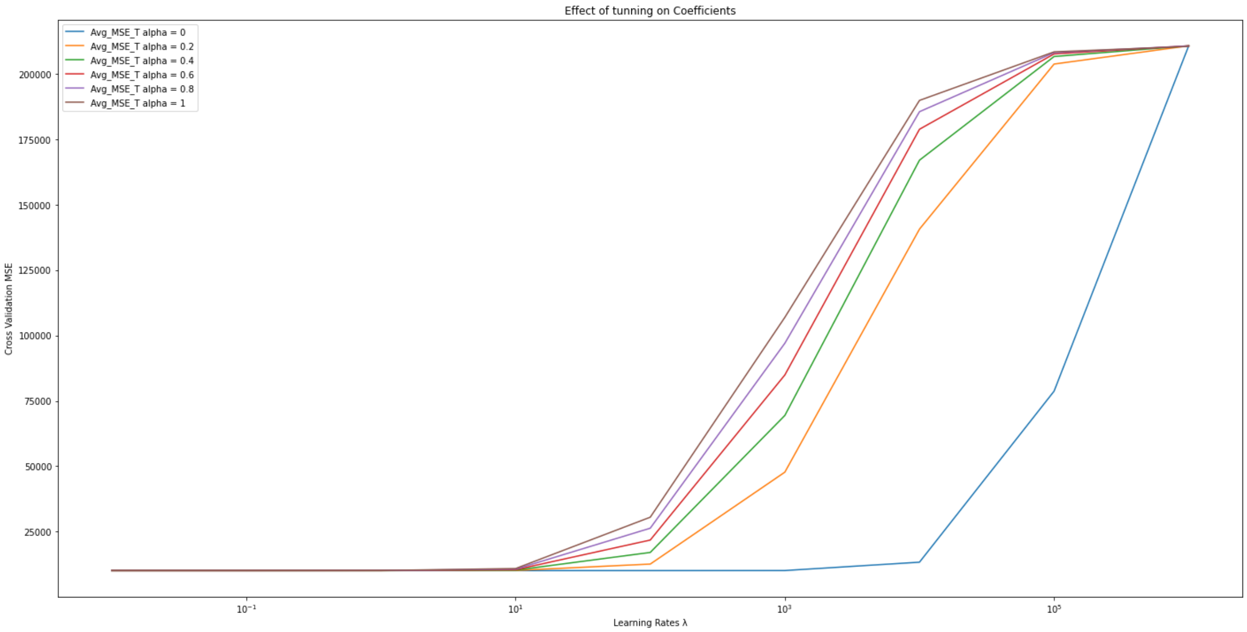
Primary results From scratch vs. libraries for ridge regression:

Using basic python numpy and pandas libraries as (P1), we created an algorithm from scratch. We computed the initial Elastic Net with a coordinate descent algorithm. The results of these two algorithms were comparable as seen in the output for the λ tuning parameters As follows:

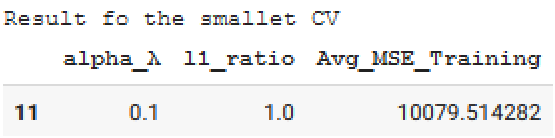
Deliverable 1:



Deliverable 2:



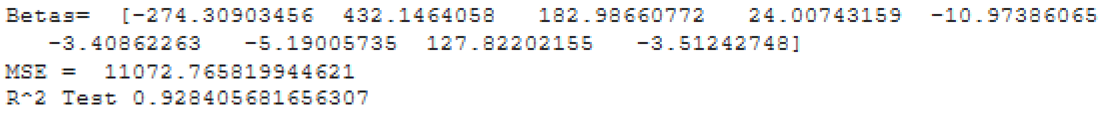
Deliverable 3:



Result fo the smallet CV

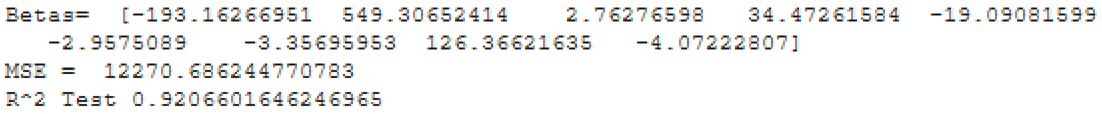
	alpha_λ	l1_ratio	Avg_MSE_Training
11	0.1	1.0	10079.514282

Deliverable 4.1:



```
Betas=  [-274.30903456  432.1464058   182.98660772   24.00743159  -10.97386065
   -3.40862263   -5.19005735  127.82202155   -3.51242748]
MSE =  11072.765819944621
R^2 Test 0.928405681656307
```

Deliverable 4.2:

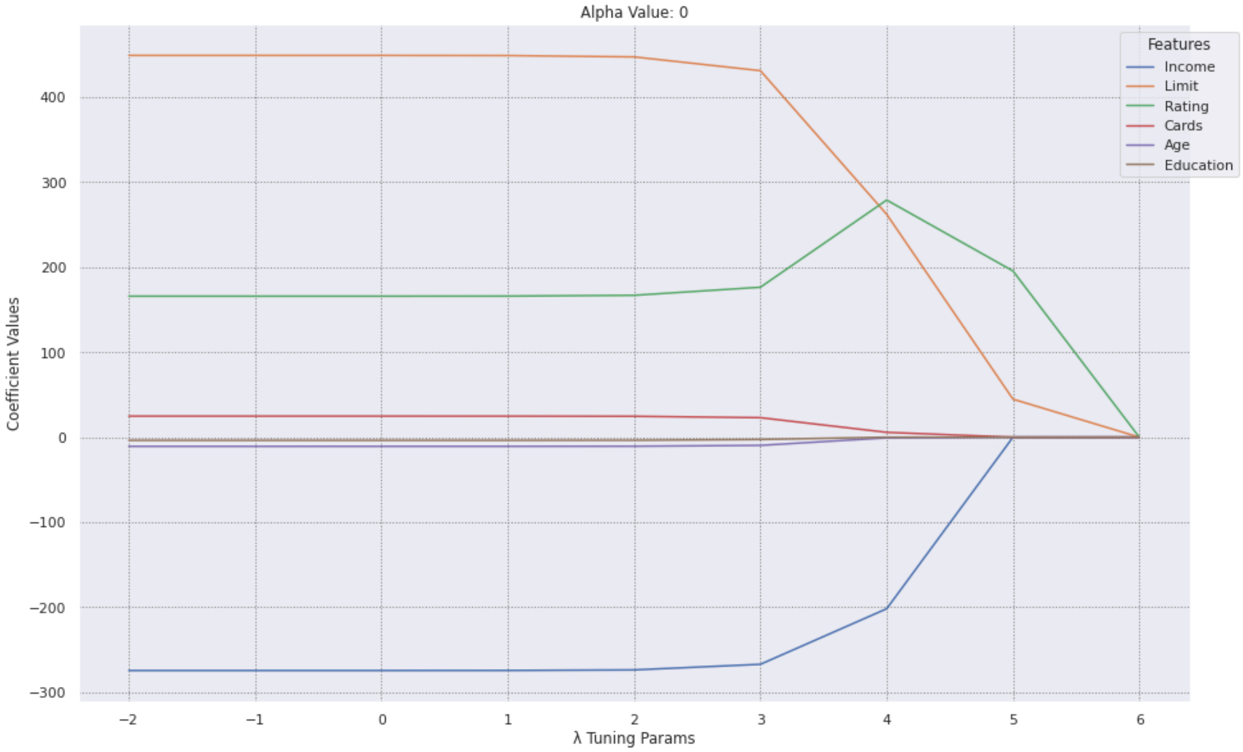


```
Betas=  [-193.16266951  549.30652414    2.76276598   34.47261584  -19.09081599
   -2.9575089    -3.35695953  126.36621635   -4.07222807]
MSE =  12270.686244770783
R^2 Test 0.9206601646246965
```

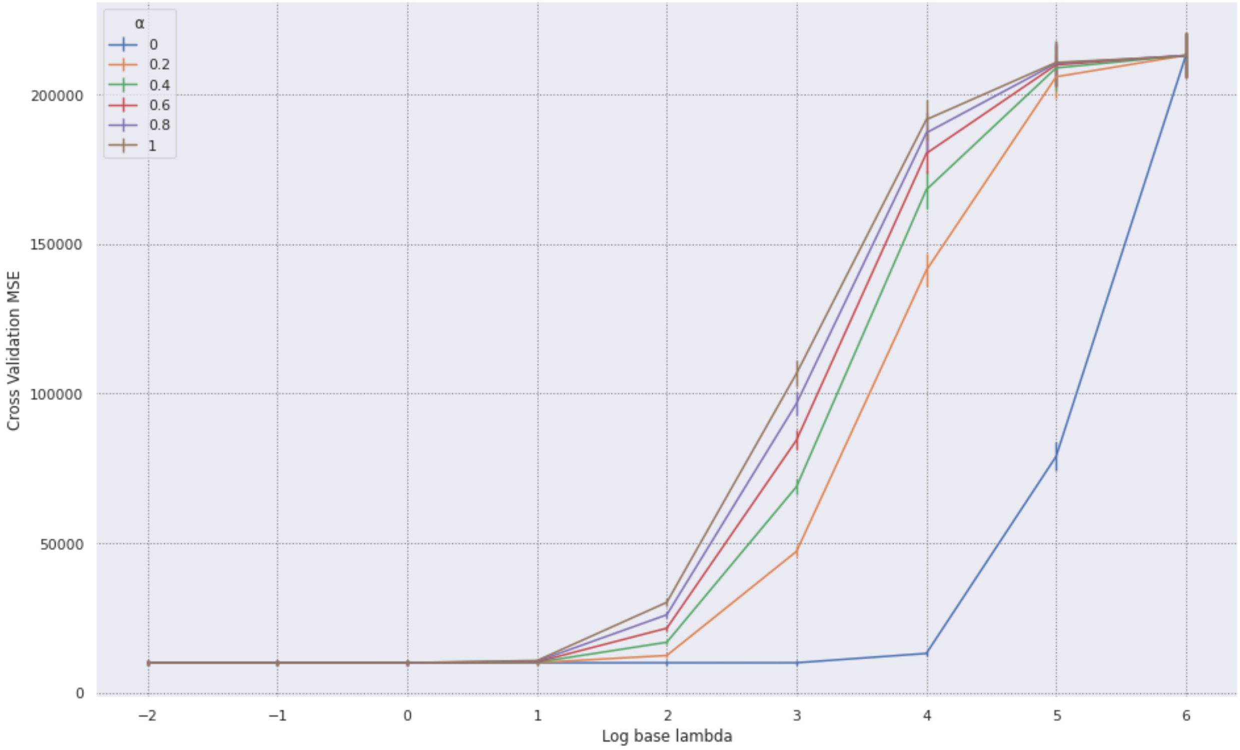
Secondary results From scratch vs. libraries for ridge regression:

Using basic python numpy and pandas libraries as (S1), we created an algorithm from scratch. As a first step, we computed the initial Elastic Net with a coordinate descent algorithm. The results of these two algorithms were comparable as seen in the output for the λ tuning parameters As follows:

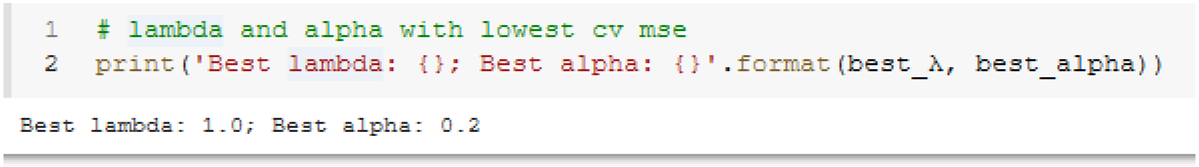
Deliverable 1:



Deliverable 2:



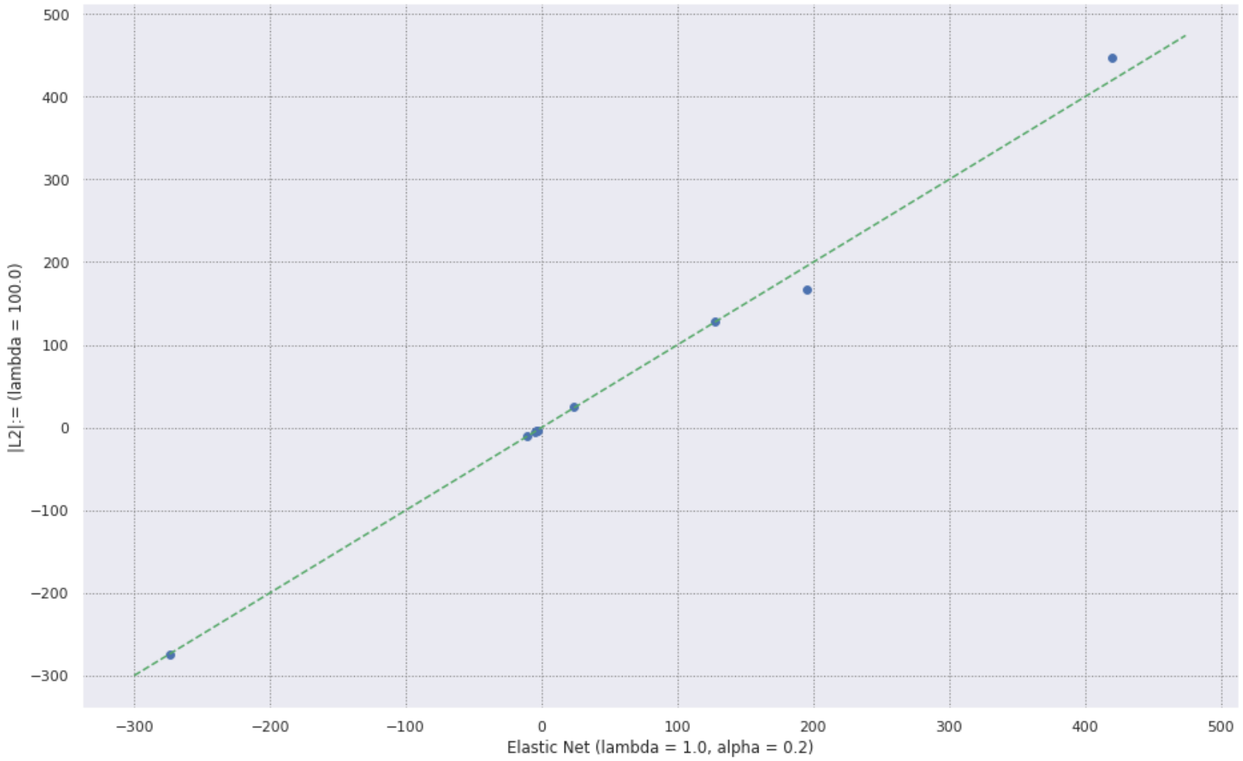
Deliverable 3:



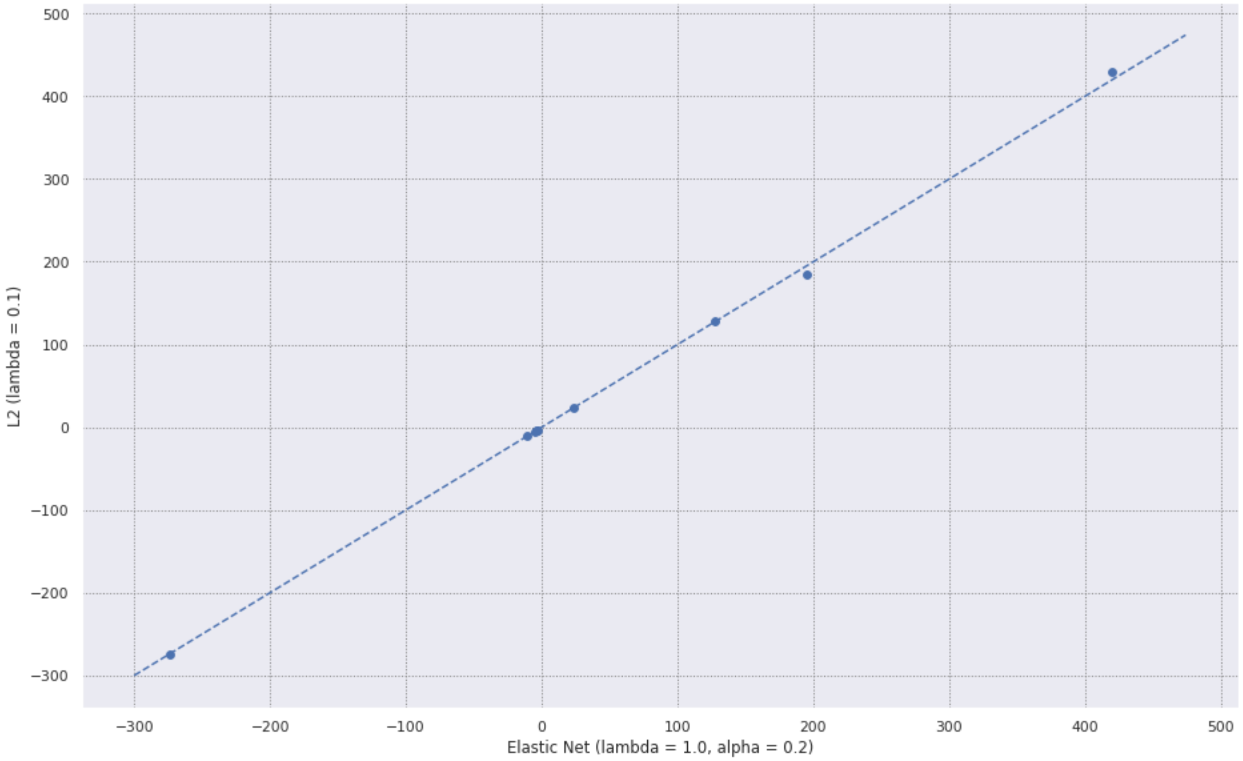
```
# lambda and alpha with lowest cv mse
print('Best lambda: {}; Best alpha: {}'.format(best_λ, best_alpha))
```

```
Best lambda: 1.0; Best alpha: 0.2
```

Deliverable 4.1:



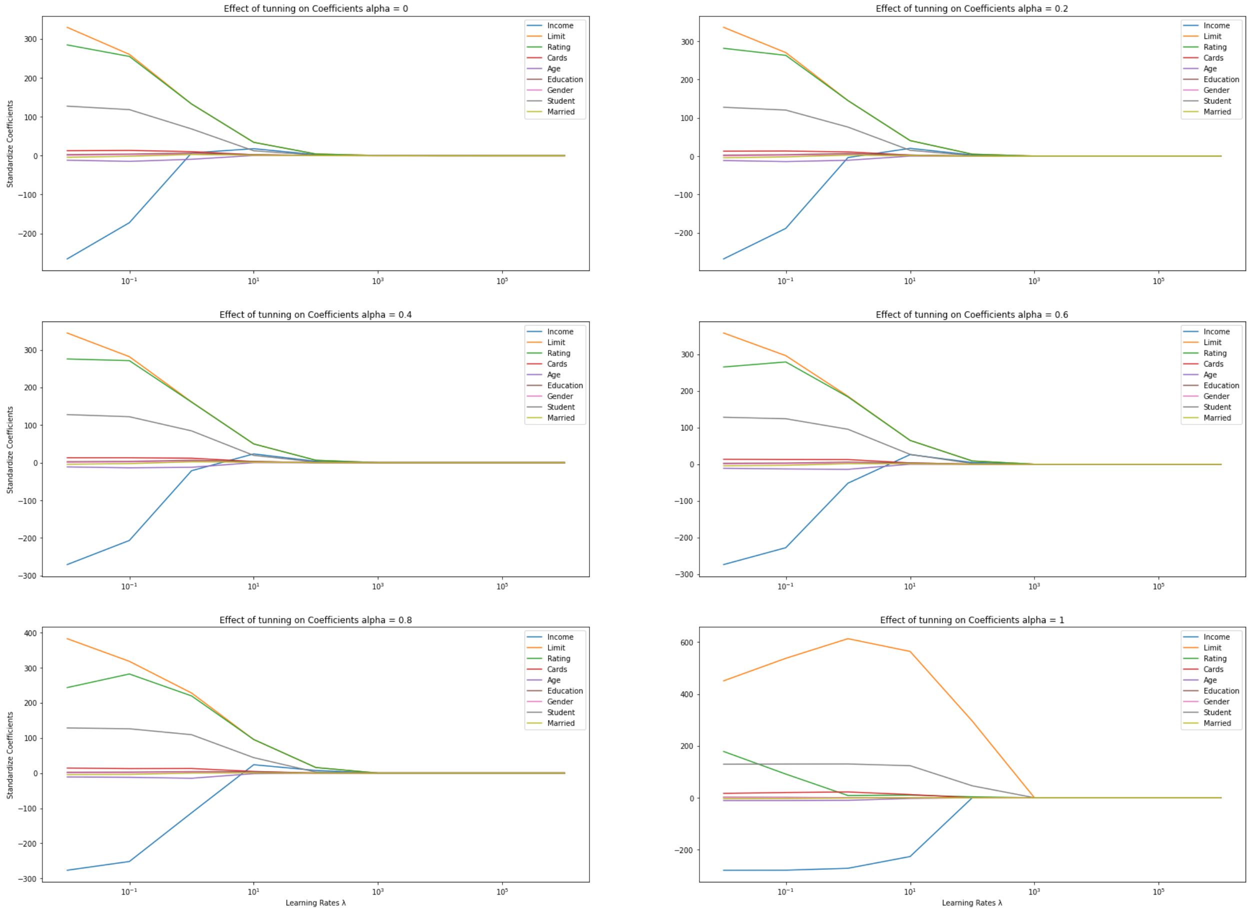
Deliverable 4.2:



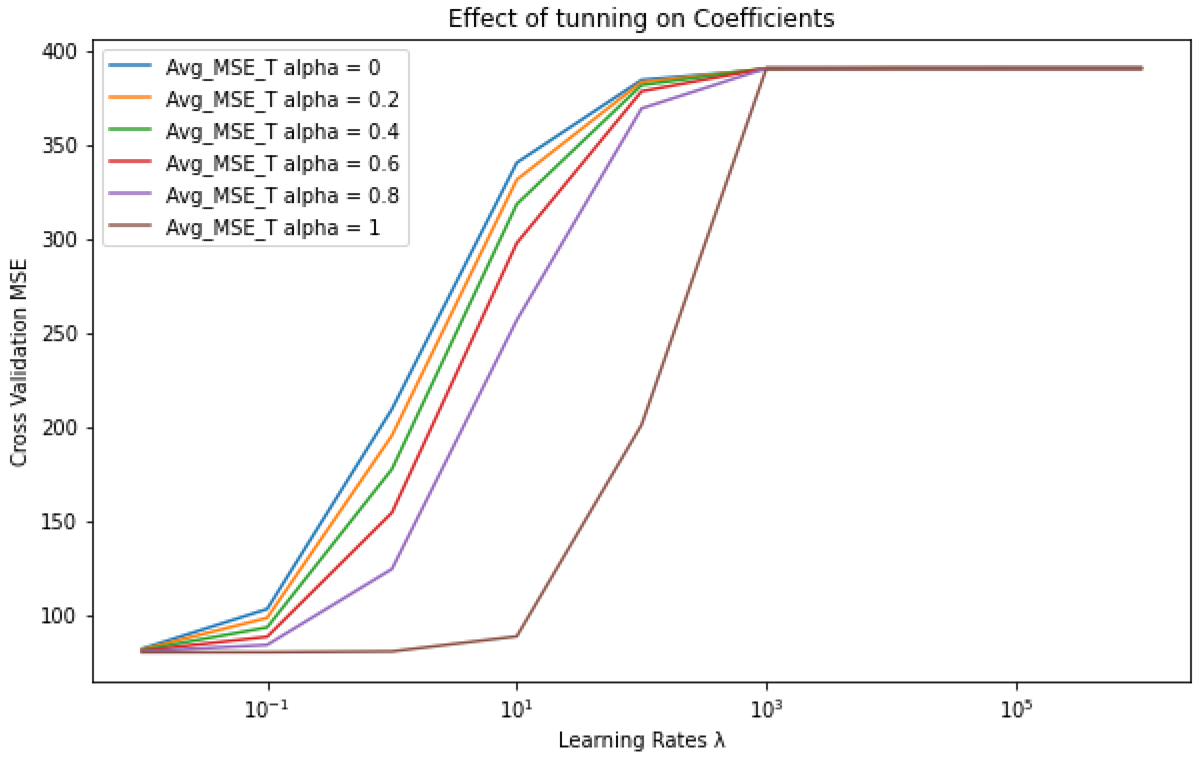
With libraries results deliverable 1 From scratch vs. libraries for ridge regression:

The comparison for deliverables 5-6 was implemented using Scikit-Learn (L1). We computed the initial Elastic Net using a coordinate descent algorithm. The results of these two algorithms were comparable as seen in the output for the λ tuning parameters As follows below.

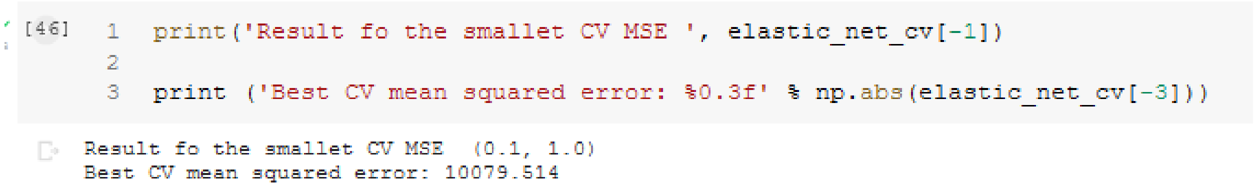
Deliverable 5.1:



Deliverable 5.2:



Deliverable 5.3:

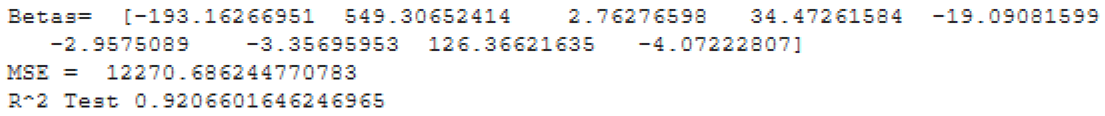


```
[46]
print('Result fo the smallet CV MSE ', elastic_net_cv[-1])

print ('Best CV mean squared error: %0.3f' % np.abs(elastic_net_cv[-3]))
```

```
Result fo the smallet CV MSE  (0.1, 1.0)
Best CV mean squared error: 10079.514
```

Deliverable 5.4:



```
Betas= [-193.16266951  549.30652414    2.76276598   34.47261584  -19.09081599
   -2.9575089    -3.35695953  126.36621635   -4.07222807]
MSE =  12270.686244770783
R^2 Test 0.9206601646246965
```

Conclusion:

Although we implemented the two algorithms from scratch, we found that the convergence happens much faster using the dependent libraries for elastic net and current over 5 K-folds cross validation.

The reason for this is that the sci-kit learn libraries implement more precise tuning parameters. By retraining the model, we can also determine the optimal λ and α pairs using two variations of the algorithm. We see that the estimates both compare to the estimates obtained from ridge regression (α =1 under optimal λ for α=1) and lasso (α =0 under optimal λ for α =0) leaning more toward being more optimized with the Lasso algorithm.