Assignment 4

Regularised Regression

Introduction to the data:

Superconductivity Dataset

The dataset consists 81 features extracted from 21263 superconductors, based on these independent variable, the critical temperature is measured.

Data cleaning:

Among the 81 features extracted, many of the features has their weighted means as another feature, so due to multicollinearity we manually dropped those columns and kept the weighted means columns only and this resulted in having 41 columns.

Now to check the multicollinearity, we first calculate the correlation matrix and then remove the columns which has correlation higher tham 0.75 or lower than -0.75 with any other column, to do this we first take the upper triangular matrix of the initial correlation matrix to avoid dropping columns respected to eachother twice.

After performing this we have 19 attributes left.

Now, we will fit the data to regression models.

Ordinary Least Square (OLS)

The OLS regression gives us

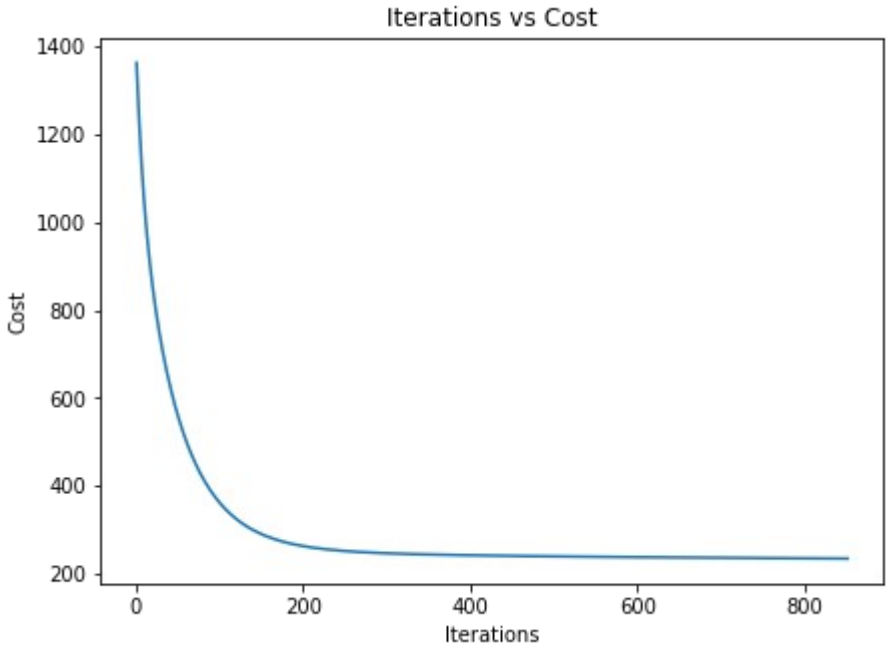
```
R_squared value:  0.8366660172230774
Testing Accuracy in training data: 0.4458957469539231
Testing Accuracy in test data: 0.3715599052116054

The optimum coefficients are:

        1.28844845e+00, -1.54839620e-02, -2.99293823e-02,  3.59957698e-01,
        1.49100806e-02, -1.00111302e+01,  1.34203846e-02, -1.23266099e-01,
       -2.54870388e-03, -1.91720117e-01,  2.29964349e-01,  1.09044254e-01,
       -5.04914602e-01,  6.41687680e-01, -6.23093198e-01,  2.32431991e+01,
       -9.13235901e-02,  3.09950084e+00, -1.89617404e+01
```

Sequential Gradient Descent (Widrow Hoff)

The widrow hoff algorithm takes 851 iterations and the scores are:

```
Train Score: 0.3806799042913319
Test Score:  0.2944854254744175
```

The iteration vs cost graph is :



We can see as iteration increases, the cost decreases rapidly near 200, then gradually converges to 200.

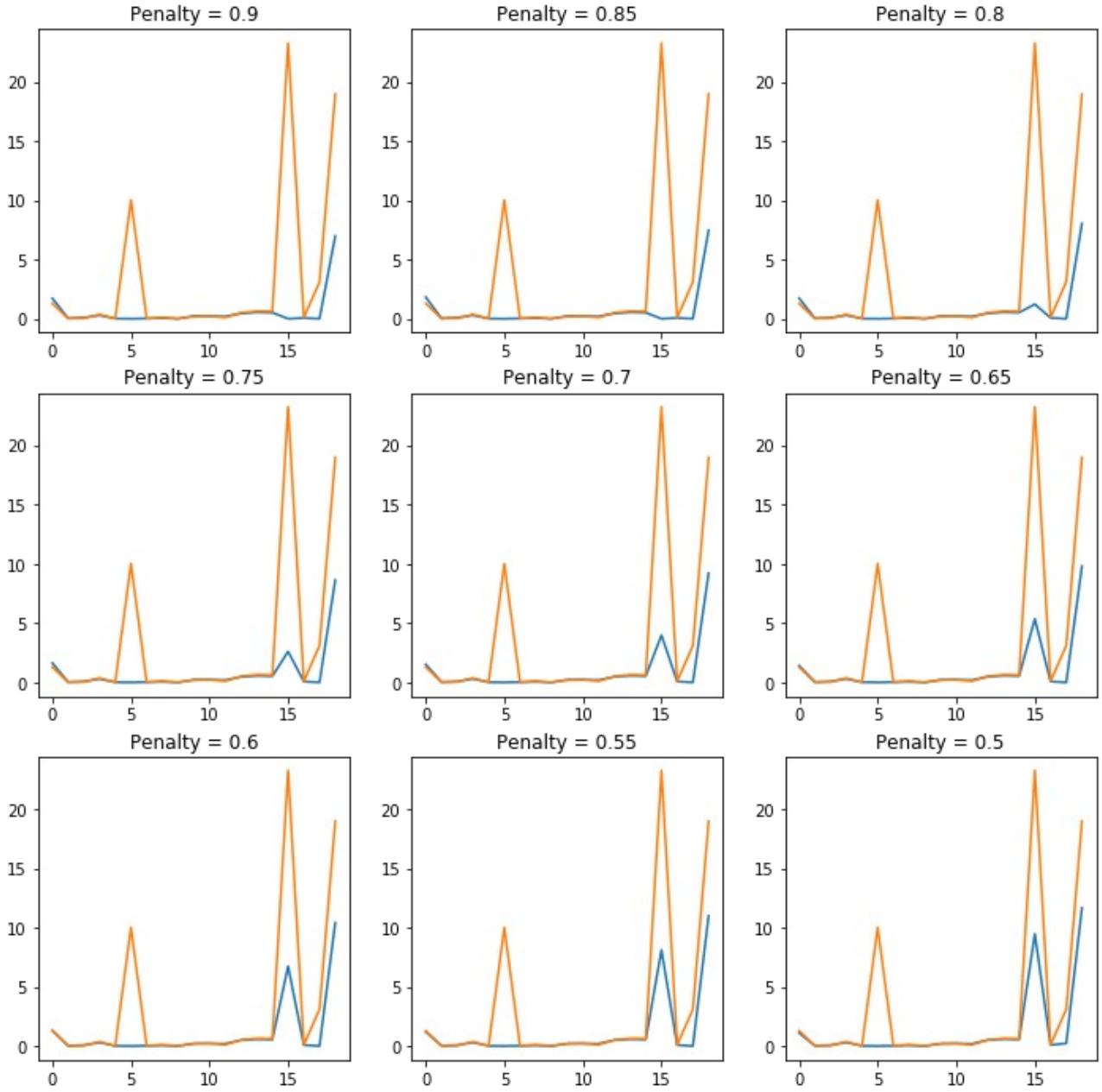
Lasso Regularization

For the following values of penalty : [0.9, 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55, 0.5]
We get the following result using lasso regularization:

```
alpha =  0.9
Number of Coefficients used:  16
R squared training set 62.16
R squared test set 52.12
Accuracy  0.2592864572519009

alpha =  0.85
Number of Coefficients used:  16
R squared training set 62.24
R squared test set 52.13
Accuracy  0.2623512088745874

alpha =  0.8
Number of Coefficients used:  17
R squared training set 62.47
R squared test set 52.45
Accuracy  0.2717418121485117

alpha =  0.75
Number of Coefficients used:  17
R squared training set 62.7
R squared test set 52.76
Accuracy  0.2815525365990489

alpha =  0.7
Number of Coefficients used:  17
R squared training set 62.92
R squared test set 53.01
Accuracy  0.29097179572223597

alpha =  0.65
Number of Coefficients used:  17
R squared training set 63.12
R squared test set 53.2
Accuracy  0.29998512149513534

alpha =  0.6
Number of Coefficients used:  17
R squared training set 63.31
R squared test set 53.34
Accuracy  0.3085799983712664

alpha =  0.55
Number of Coefficients used:  17
R squared training set 63.48
R squared test set 53.42
Accuracy  0.3167463287474188

alpha =  0.5
Number of Coefficients used:  18
R squared training set 63.66
R squared test set 53.51
Accuracy  0.3247877508063035
```

As the value of the penalty decreases we can see the lasso regualrization selects more features, like for penalty 0.9 and 0.85 it was dropping 3 features (i.e. making the corresponsing coefficients values 0), but for 0.8 to 0.55, the model took 17 features and dropped 2, and for penalty 0.5 it dropped only one attribute. The accuracy increases as we decrease the penalty value, this is due to selection of more attributes.

Comparing the Lasso regularizations coefficients with OLS coefficients:

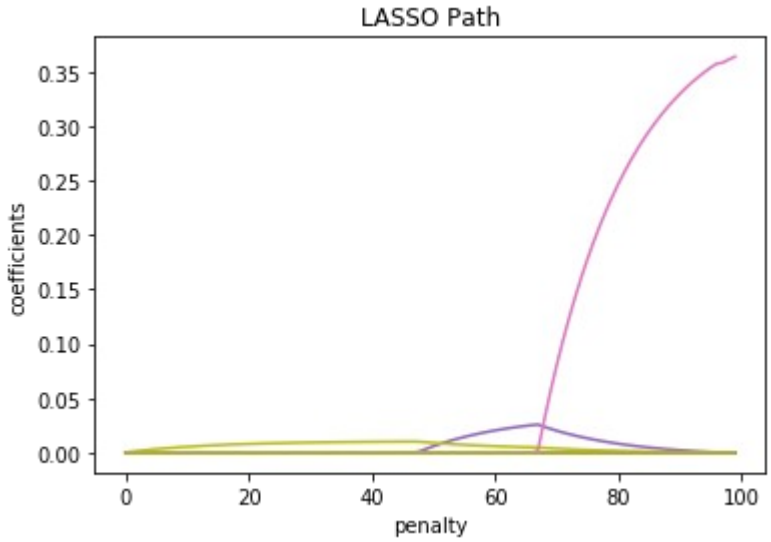


We can see that some of the coefficient values are dropped to 0, and later increased gradually as the penalty value decreases. (example: the coefficients whose value is higher than 20 is dropped to zero when penalty is 0.9 and then raised gradually near 10, first when penalty is 0.8 and then increases)

X axis: ith coefficients
Y axis: Valie of the coeeficients

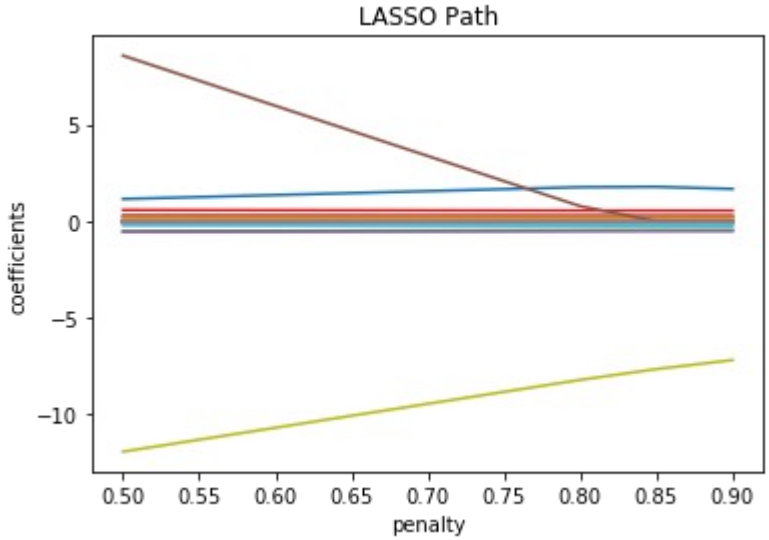
Orange: OLS coefficients
Blue: Lasso Regularized coefficients

Lasso Path:



Here the penalties taken are computer genarated.

Lasso path for penalty : [0.9, 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55, 0.5]



This shows us how the optimum value of each 19 coefficients changes as we change the value of the penalty.

Lasso Coefficients:

	alpha = 0.90	alpha = 0.85	alpha = 0.80	alpha = 0.75	alpha = 0.70	alpha = 0.65	alpha = 0.60	alpha = 0.55	alpha = 0.50
0	1.713281	1.817528	1.733729	1.627072	1.520445	1.413822	1.307202	1.200592	1.105388
1	0.036112	0.034030	0.033523	0.033208	0.032893	0.032578	0.032263	0.031947	0.031708
2	-0.091492	-0.088557	-0.086494	-0.084538	-0.082581	-0.080624	-0.078667	-0.076710	-0.075980
3	0.291055	0.291254	0.292769	0.294444	0.296118	0.297791	0.299465	0.301138	0.304983
4	0.013861	0.013510	0.014135	0.014877	0.015620	0.016363	0.017106	0.017848	0.019325
5	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
6	0.024680	0.024450	0.024409	0.024390	0.024371	0.024352	0.024333	0.024314	0.023699
7	-0.051895	-0.053406	-0.053387	-0.053183	-0.052978	-0.052774	-0.052569	-0.052365	-0.054341
8	-0.002373	-0.002348	-0.002362	-0.002381	-0.002399	-0.002417	-0.002436	-0.002454	-0.002488
9	-0.231233	-0.230037	-0.227070	-0.223887	-0.220705	-0.217523	-0.214340	-0.211158	-0.209582
10	0.220618	0.222005	0.220878	0.219445	0.218013	0.216581	0.215149	0.213718	0.211957
11	0.189576	0.191122	0.191778	0.192326	0.192875	0.193424	0.193974	0.194523	0.189878
12	-0.443799	-0.448235	-0.457304	-0.466934	-0.476565	-0.486196	-0.495828	-0.505459	-0.510292
13	0.545287	0.544944	0.550227	0.556195	0.562159	0.568121	0.574083	0.580044	0.585384
14	-0.516548	-0.516237	-0.521898	-0.528287	-0.534670	-0.541053	-0.547436	-0.553817	-0.558835
15	0.000000	0.000000	1.227828	2.604583	3.981244	5.357890	6.734526	8.111130	9.465002
16	-0.065162	-0.067554	-0.070013	-0.072481	-0.074946	-0.077410	-0.079874	-0.082337	-0.083825
17	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.228675
18	-6.971881	-7.458012	-8.035881	-8.624808	-9.213740	-9.802673	-10.391606	-10.980541	-11.653420

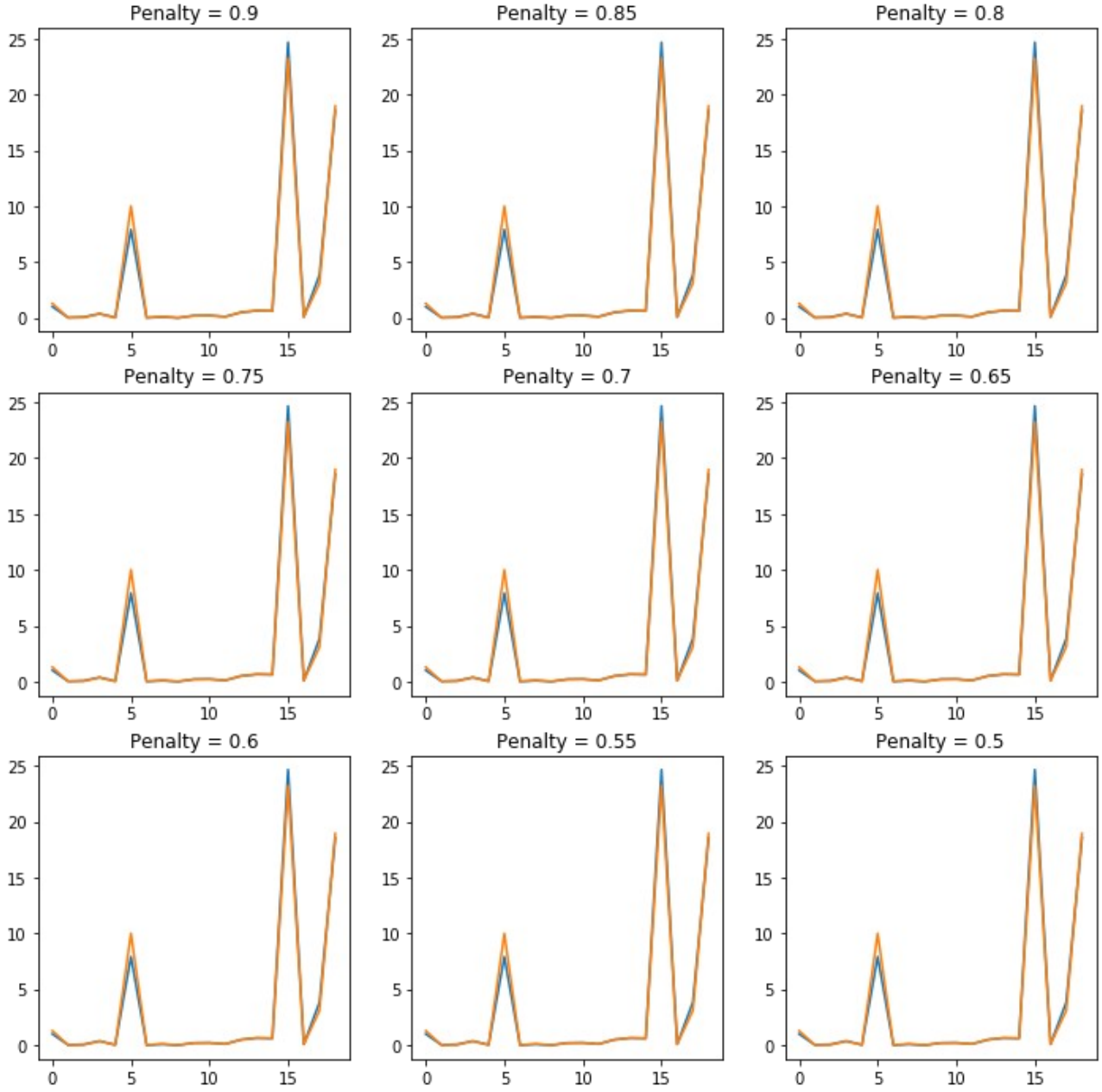
Ridge Regularization

For the following values of penalty : [0.9, 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55, 0.5]
We get the following result using ridge regularization:

```
alpha =  0.9
R squared training set 64.64
R squared test set 51.13
Accuracy  0.3935326951660115

alpha =  0.85
R squared training set 64.64
R squared test set 51.13
Accuracy  0.3935426144205467

alpha =  0.8
R squared training set 64.64
R squared test set 51.13
Accuracy  0.39355253668969314

alpha =  0.75
R squared training set 64.64
R squared test set 51.12
Accuracy  0.39356246197508093

alpha =  0.7
R squared training set 64.64
R squared test set 51.12
Accuracy  0.3935723902783427

alpha =  0.65
R squared training set 64.64
R squared test set 51.12
Accuracy  0.39358232160111073

alpha =  0.6
R squared training set 64.64
R squared test set 51.12
Accuracy  0.39359225594501857

alpha =  0.55
R squared training set 64.64
R squared test set 51.12
Accuracy  0.3936021933116991

alpha =  0.5
R squared training set 64.64
R squared test set 51.12
Accuracy  0.39361213370278625
```

The accuracy increases a little if we decrease the values of the penalty.

Comparing the Ridge regularizations coefficients with OLS coefficients:

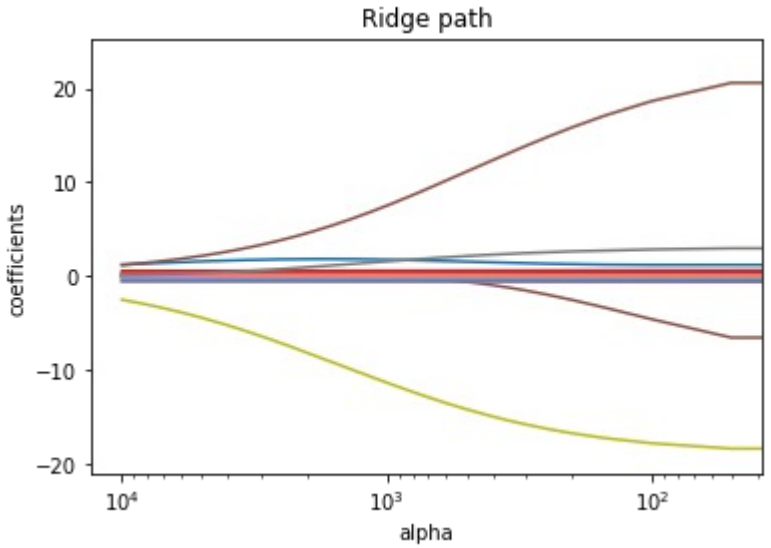


The coefficients are shrinked a little when comparing with the OLS coefficients, but the change when the penalties are changed are not very much visible.

X axis: ith coefficients
Y axis: Valie of the coeeficients

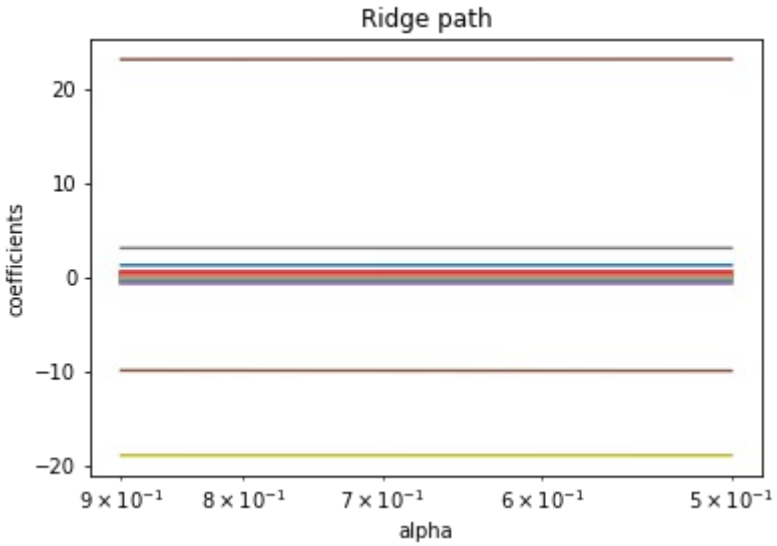
Orange: OLS coefficients
Blue: Lasso Regularized coefficients

Ridge Path:



Here the penalties taken are equispaced between 0 to 10000

Ridge path for penalty : [0.9, 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55, 0.5]



This shows us how the optimum value of each 19 coefficients changes as we change the value of the penalty.
For the manually taken penalties, the coefficients differ very little to be observable.

Ridge Coefficients:

	alpha = 0.90	alpha = 0.85	alpha = 0.80	alpha = 0.75	alpha = 0.70	alpha = 0.65	alpha = 0.60	alpha = 0.55	alpha = 0.50
0	1.005144	1.005251	1.005358	1.005466	1.005574	1.005682	1.005790	1.005898	1.006007
1	0.028266	0.028266	0.028265	0.028265	0.028265	0.028265	0.028265	0.028265	0.028265
2	-0.074818	-0.074817	-0.074816	-0.074815	-0.074814	-0.074813	-0.074812	-0.074811	-0.074810
3	0.372129	0.372143	0.372156	0.372170	0.372184	0.372197	0.372211	0.372224	0.372238
4	0.038822	0.038824	0.038827	0.038829	0.038831	0.038833	0.038835	0.038837	0.038839
5	-7.899910	-7.903995	-7.908082	-7.912173	-7.916266	-7.920363	-7.924462	-7.928565	-7.932670
6	0.008583	0.008580	0.008577	0.008574	0.008571	0.008567	0.008564	0.008561	0.008558
7	-0.082279	-0.082277	-0.082275	-0.082273	-0.082271	-0.082268	-0.082266	-0.082264	-0.082262
8	-0.002910	-0.002910	-0.002910	-0.002910	-0.002910	-0.002910	-0.002910	-0.002911	-0.002911
9	-0.205117	-0.205114	-0.205112	-0.205110	-0.205108	-0.205106	-0.205104	-0.205102	-0.205099
10	0.209917	0.209923	0.209929	0.209935	0.209941	0.209947	0.209953	0.209959	0.209965
11	0.106144	0.106134	0.106124	0.106114	0.106105	0.106095	0.106085	0.106075	0.106065
12	-0.490767	-0.490751	-0.490736	-0.490720	-0.490705	-0.490689	-0.490674	-0.490658	-0.490642
13	0.650113	0.650130	0.650147	0.650164	0.650181	0.650199	0.650216	0.650233	0.650250
14	-0.618779	-0.618797	-0.618816	-0.618835	-0.618854	-0.618872	-0.618891	-0.618910	-0.618929
15	24.661254	24.664427	24.667602	24.670778	24.673955	24.677134	24.680313	24.683494	24.686677
16	-0.088175	-0.088174	-0.088172	-0.088170	-0.088169	-0.088167	-0.088165	-0.088164	-0.088162
17	3.852845	3.853013	3.853181	3.853349	3.853517	3.853685	3.853853	3.854021	3.854189
18	-18.567563	-18.568139	-18.568716	-18.569293	-18.569870	-18.570446	-18.571023	-18.571600	-18.572176

We can see the coefficients shrinkes more as we increase the penalty.