Assignment 2-linear regression

ICS485 (Machine learning) Term: 191

*__NOTE__ : This is an update to part b of the assignment due to the postpone of the deadline and for the new information revealed, part A will be as is in the previous submission

Hatim Alfahmi - 201467840

1. Documents all the results in the report.

q_Two:

```
MSE of built-in linear regression(training):
1.3927486422724714e-28
MSE of gradient descent solver for ridge regression
(training):  1.3353171354017344
MSE of built-in solver for ridge regression (training):
0.03480412397500122
MSE of built-in linear regression(test):
50.98181057763657
MSE of gradient descent solver for ridge regression
(test):  4.8859572319037925
MSE of built-in solver for ridge regression (test):
4.590164991621038
```

q_Four:

```
MSE of built-in linear regression(training):
1.3927486422724714e-28
MSE of gradient descent solver for lasso regression
(training):  0.011272278703378033
MSE of built-in solver for lasso regression (training):
8.78299643035392
MSE of built-in linear regression(test):
50.98181057763657
MSE of gradient descent solver for lasso regression
(test):  2.7277095698900076
MSE of built-in solver for lasso regression (test):
9.566046105001256
```

2. Try with a large value of C (e.g., 20) for lasso and check the weights and MSE. What do you observe?

When enlarging the c value we notice that the error becomes greater the larger the value of C

3. Compare the coefficients (parameter values) for ridge and lasso for the best setup. What do you observe? Can you explain?

With ridge the parameters are small positive numbers and it can go a bit below zero as you can see in the picture below, where in lasso as you can see below the parameters has larger variance

Ridge :

```
[ 0.0222661   0.03500623  0.07191945  0.02218244  0.24769803  0.01214546
  0.14473069 -0.01992597  0.18875785 -0.02290988  0.134311    0.18711323
  0.03131055  0.17820117 -0.01128704  0.17403624 -0.07672466 -0.02477998
 -0.07055908  0.25280915  0.0067114   0.05443752  0.02493596  0.0885071
 -0.1146895   0.09639893 -0.00609383  0.00665273 -0.03373151  0.04578291
  0.05360968  0.03679608 -0.01392632 -0.03288788  0.04643155 -0.00814038
  0.0481336   0.00562323 -0.02910642 -0.09163953  0.01248366  0.00702026
  0.06239537  0.00200517 -0.02015205  0.02118849 -0.01678329  0.01961642
  0.01152845 -0.045262    0.01001699  0.00721633 -0.01314242 -0.06097233
  0.09689571 -0.05941824  0.04960649  0.09246913  0.06165309 -0.03391731
  0.08526875  0.03205506 -0.04255685  0.04198559 -0.08697497 -0.01678209
  0.0108758  -0.07073486  0.07839249  0.04189966  0.0391842  -0.00573017
  0.0225923  -0.02295584  0.0002834   0.07499254  0.00161236 -0.03003299
  0.00744248 -0.02785378  0.01520529 -0.04464682 -0.02503134  0.00377554
  0.08559781 -0.0316764   0.06919203 -0.03276227  0.04562744  0.13800154
  0.00715751  0.03239    -0.01060639 -0.0705888  -0.01556905 -0.01685448
  0.03197218 -0.0645504   0.05707945 -0.02971918]
```

Lasso :

```
[-0.00000000e+00 -1.50798198e+00  1.13466197e+01  0.00000000e+00
  1.86440209e+01 -1.31814722e-01  7.40525516e+00  0.00000000e+00
  1.82858609e+01  1.25218859e+00  1.02856046e+01  2.35393148e+01
  2.60031372e-01  5.35660365e+00 -1.93524388e+00  1.80215691e+01
 -2.58103637e+00  7.98310649e+00  4.69037502e-01  2.25316505e+01
  1.25561708e-03  6.50603215e-01  2.22065525e+00  0.00000000e+00
 -2.31511451e+00 -1.24544796e-01 -2.02187889e-01 -0.00000000e+00
  6.63213923e-01 -0.00000000e+00  2.28138237e+00 -1.64843300e+00
  0.00000000e+00 -1.03737697e+00 -1.82969768e+00 -0.00000000e+00
 -2.51591612e-01  2.06030888e+00  3.94268476e+00  6.65239711e-01
 -1.14403815e+00  1.96873244e+00 -0.00000000e+00  1.23596098e+00
  8.46980931e-02  2.06990825e+00  2.48416475e+00  1.25671947e+00
 -6.20250911e-01 -1.20015565e+00  6.50496406e-02  0.00000000e+00
  1.45181669e+00  9.47158286e-01  1.60796896e-01  0.00000000e+00
  2.46212317e+00  8.43309773e-01 -0.00000000e+00  0.00000000e+00
  0.00000000e+00 -2.56428224e-01  0.00000000e+00  6.89069583e-01
 -1.36960960e+00  4.64352823e-01  1.91495454e+00 -8.19150691e-02
  0.00000000e+00  5.86846828e-01  2.03407959e+00 -0.00000000e+00
 -2.24282776e+00 -0.00000000e+00 -0.00000000e+00  0.00000000e+00
  2.19958032e+00  2.48082360e+00  1.52247037e+00  6.85468781e-01
 -1.90238300e+00 -5.35446406e-01 -8.00331073e-01 -1.99658030e+00
  0.00000000e+00  1.25690968e+00 -0.00000000e+00 -5.99917270e-01
 -0.00000000e+00  0.00000000e+00  9.20692709e-01 -0.00000000e+00
  0.00000000e+00 -5.06394742e-01  2.67814238e-01  1.74462684e+00
  8.09357365e-01  4.88762542e-01  0.00000000e+00 -1.73459981e+00]
-0.035626418603313226
```

4. Compare MSE of linear, ridge, and lasso. What do you observe?

The MSE in liner regression is very large as you can see in question 1. However, the built-in solver for ridge regression did good **(4.59)**, and fortunately my implementation of gradient descent solver for ridge regression did very close with **(4.88)**.

On the other hand the gradient descent solver for lasso regression did very good with **(2.73)**

5. Which among the ridge and lasso gives the best results on the test? Can you explain why?

The best results comes from lasso with **(2.73)**. However, I did some research and I found out the ridge is supposed to perform better than lasso unless all but a few of the coefficients in linear regression are nearly zero and the rest are large (which is not the case). So taking that into consideration it might be the reason but I still do not think so, and I believe that maybe there is a flaw in the implementation of lasso that lead to this behavior. And since the built-in solver for lasso regression gave MSE of **(9.57)** I think that the best might be ridge with **(4.88)**

6. Can the lasso regression retrieve the 10 features which were used in the equation for y? List them.

We tried to do so and we failed because we were not trained to use the library in time