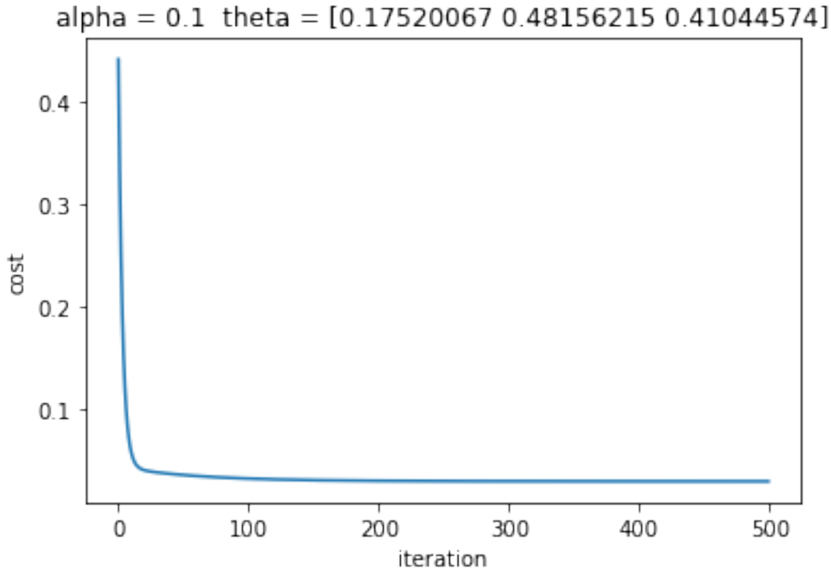
ECON 425 Homework Assignment 2

Madelyn Caufield

Part I

The following results were produced from a linear regression made from scratch. A gradient descent algorithm was developed to find the values of coefficients of a function that minimized a cost function. To implement this, I chose different values of alpha and numbers of iterations in attempts at getting the lowest possible cost. I normalized the data using 1/3 functions provided, and split the data into training and testing sets. I further split the testing and training data into their associated X and Y values (X training, Y training, X testing, Y testing). Starting at ALPHA = 0.1; MAX_ITER = 500, I decreased values of ALPHA, observing how the convergence curve begins to converge at a slower rate (discussed more below). I then acquired the training error in order to calculate the average error and standard error. The final test was the R^2, in which I used the split data to calculate. The graphs below show the convergence curves for different values of alpha with different iterations. Three different data normalization methods are also displayed for ALPHA = 0.01.

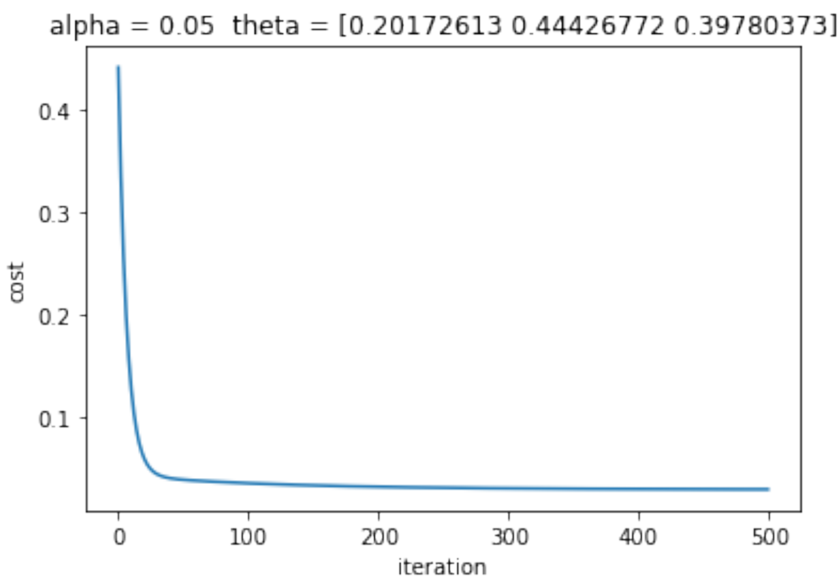
ALPHA = 0.1; MAX_ITER = 500




```
results: 0.1727749787556687 (0.12688442297897462)
Out of Sample R^2:  -0.06856196284761995
```

Beginning with ALPHA = 0.1; MAX_ITER = 500, the convergence curve converges too quickly, represented by a sharp L shape.

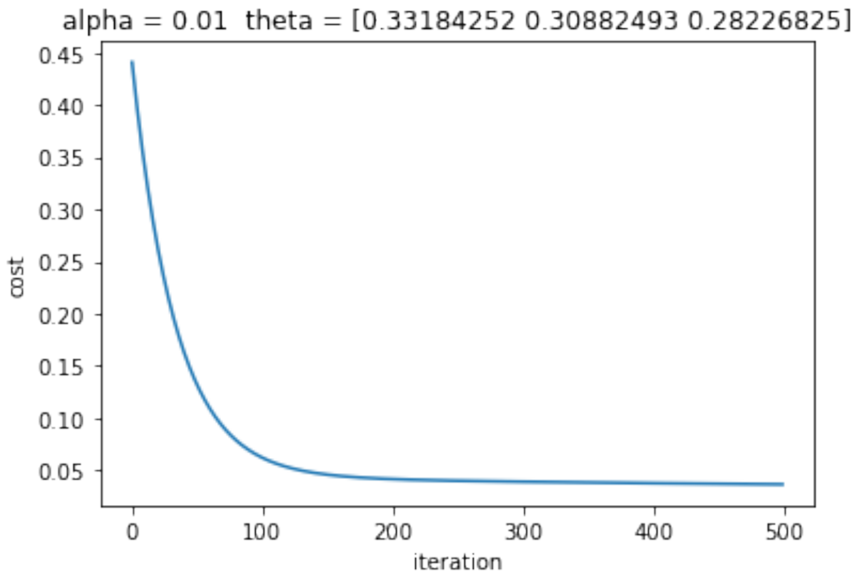
ALPHA = 0.05; MAX_ITER = 500




```
results: 0.16752026423804484 (0.12395549180620766)
Out of Sample R^2:  -0.009894526864077546
```

As we decrease alpha, holding the same number of iterations, we see the curve gradually converge at a slower rate, with a slightly better R^2.

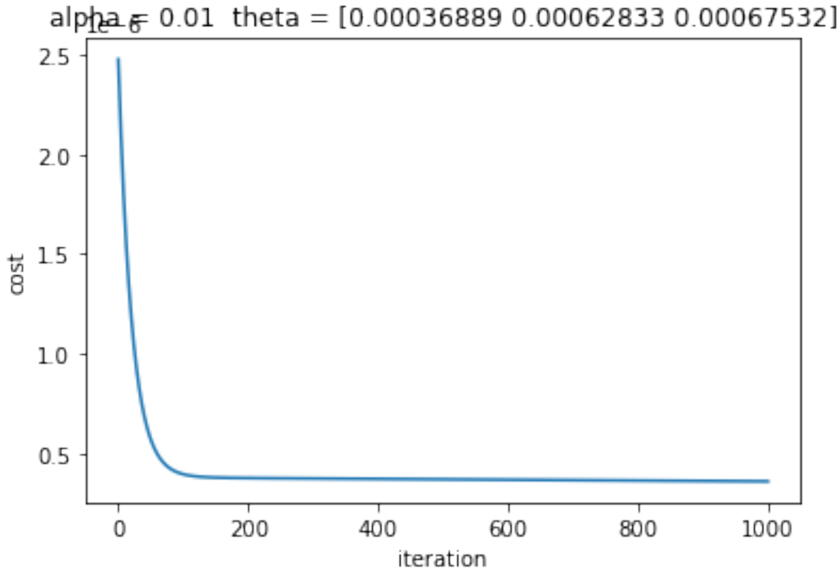
ALPHA = 0.01; MAX_ITER = 500




```
results: 0.147017033893325 (0.11722709952151875)
Out of Sample R^2:  0.17781082178660557
```

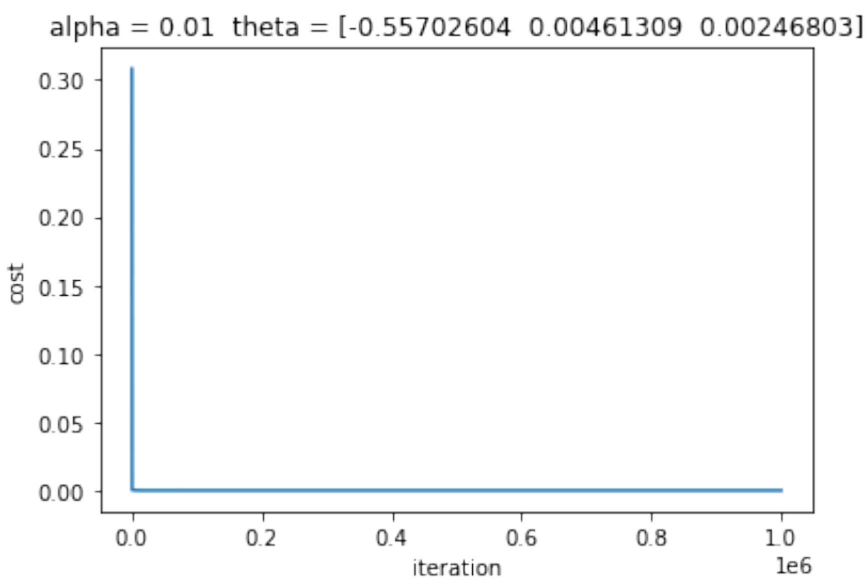
Once we decrease alpha all the way to 0.01, we see the highest R^2 out of all the different variations of the parameters ALPHA and MAX_ITER. It is also worth noting here that we see a very similar average error and standard deviation at ALPHA = 0.01 (for both matrix scaling and Min-Max scaling).

ALPHA = 0.01; MAX_ITER = 1,000 with Min-Max Scaling




```
results: 0.0003595715040199083 (0.0003056954095199245)
Out of Sample R^2:  0.077926949892274
```

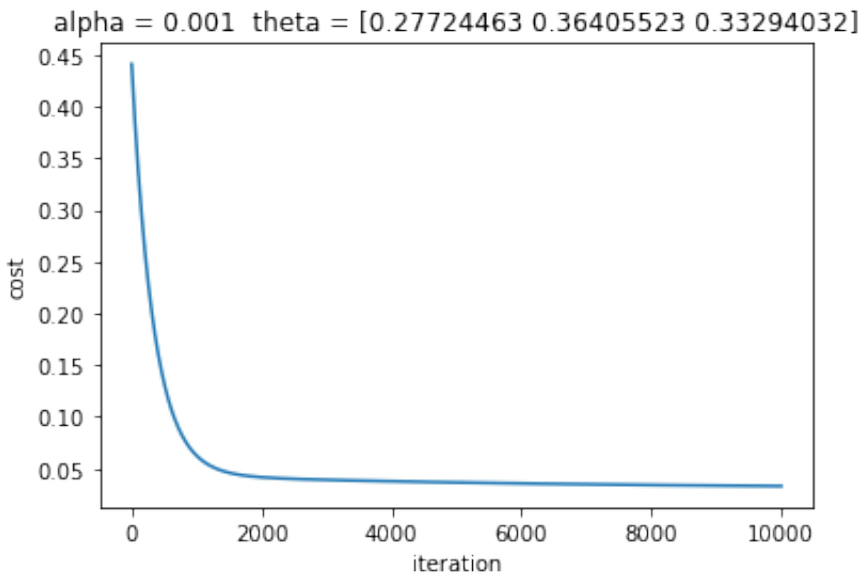
ALPHA = 0.01; MAX_ITER = 1,000,000 with Mean Normalization




```
results: 0.00041415278567939783 (0.0003015541340733733)
Out of Sample R^2:  -0.08648402780990039
```

Maintaining the ALPHA = 0.01 value that gave us our highest R^2 previously, we increase the number of iterations for the regression subject to mean normalization. Increasing the number of iterations to 1,000,000 still gave us a negative R^2, although this is a signficant improvement form the R^2 given from ALPHA = 0.01, NUM_ITER = 500 with mean normalization (not pictured).

ALPHA = 0.001; MAX_ITER = 10,000




```
results: 0.15402481172478028 (0.11898441804239651)
Out of Sample R^2:  0.11909952680158775
```

However, once we further decrease the value of alpha to 0.001, making number of iterations 10,000, we see a nicely converging convergence curve, with an R^2 that is similar to the R^2 of the regression with an alpha of 0.01 and number of iterations at 500. For ALPHA = 0.001, too small of iterations (500) and too large of iterations (100,000) gave us a negative R^2 (not pictured).

From the above output, we also observe the three different normalization methods under ALPHA = 0.01. The rescaleNormalization function performs Min-Max scaling by scaling the data to a fixed range (X-Xmin)/(Xmax-Xmin). The rescaleMatrix function is also performing Min-Max scaling except it is subtracting the minimum of each column in the numerator and denominator. The meanNormalization function replaces x_i with x_i - x_u in order to force features into having an approximately zero mean. For the rescaleMatrix function, a number of iterations of 500 was enough to achieve the highest R^2 possible, Min-Max Scaling required MAX_ITER = 1,000 to achieve it's highest R^2, and mean normalization required MAX_ITER = 1,000,000 to produce a smaller negative R^2. An important observation under each of the variations of the parameters is that R^2 is negative or extremely low for all of the models. We can attribute this to the models fitting worse than a horizontal line.

Part II

Below we have the test R^2 for each of the four linear regression methods.

	Method	Test R^2
0	Linear Regression	-0.09
1	Ridge Regression	0.17
2	LASSO	-0.09
3	Elastic Net	0.10

From the table above, we can see that the Ridge Regression method provided us with the highest R^2 of 0.17. The regression model from Part I with ALPHA = 0.01; MAX_ITER = 500 also gave us an R^2 of 0.17. Although these R^2 are not high, they show us that given our selected parameters, these were the best methods. Linear Regression and Ridge Regression both provided us with the same R^2 of -0.09.

Linear Regression

```
Training set score: 0.63
Test set score: -0.09
Out of Sample R^2:  -0.08691482476661982
```

Ridge Regression

```
Training set score: 0.53
Test set score: 0.17
Out of Sample R^2:  0.17166414541808483
```

LASSO

```
Training set score: 0.61
Test set score: 0.07
Number of features used: 2
Out of Sample R^2:  0.06928308026977537
```

Elastic Net

```
Training R^2: 0.56
Test R^2: 0.10
```