REPORT ON

ASSIGNMENT 1

ANURAG BEHERA

UTTAM SINGH

IMPORTANT INSTRUCTION :

Gradient descent code contains regularization implementation , where regularization factor is kept 0 , in case of model without regularization

Stochastic Gradient Descent

Data is divided into training and validation sets. Training data contains 80 % of the original data whereas validation data that is used for testing contains the other 20%. R^2 and RMS is calculated on testing/validation set.

Weights are randomly initialized.

Weight update rule

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

Stochastic Gradient Descent class is implemented which take various arguments such as

- alpha (learning rate)
- no of iterations
- regularization_factor (although it is set to 0 as mentioned in the question)
- stop criterion
  - based on weights
    - Training Is stopped if the gradient is very small
  - based on cost
    - Training is stopped if the decrease in cost is very small signaling its near its minima and further iteration wont necessarily decrease cost
- stop rate

Model 1:

Parameters

alpha=0.01, no of iteration=5000, reg_factor=0, stop_rate=10^{-7}

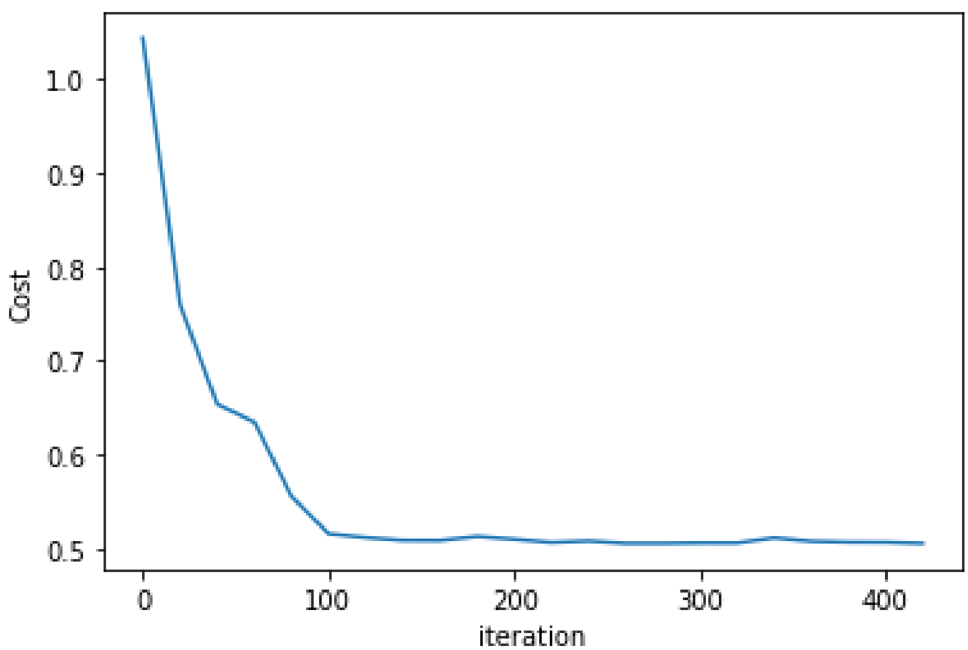
Results:

Model ran until 420 iteration before stopping

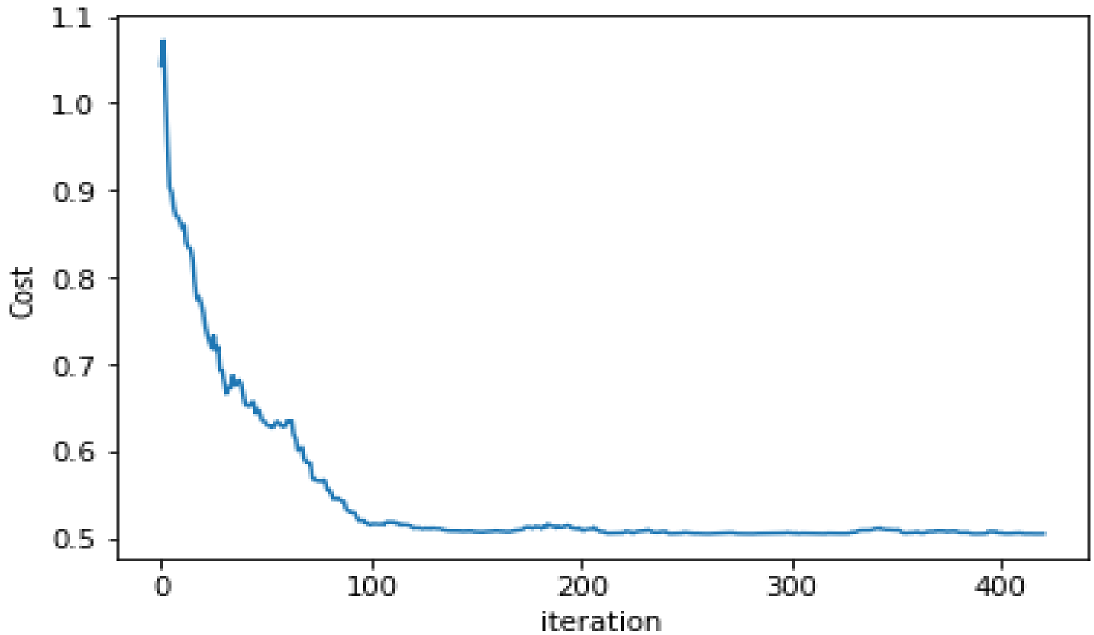
R2 score	0.03129808
RMSE	0.9115818060933101

Plots

Cost plotted after every 20^{th} Iteration



Cost plotted after every iteration:



Model 2:

Parameters

alpha=0.01, no of iteration=5000, reg_factor=0, stop_rate=10^{-8}

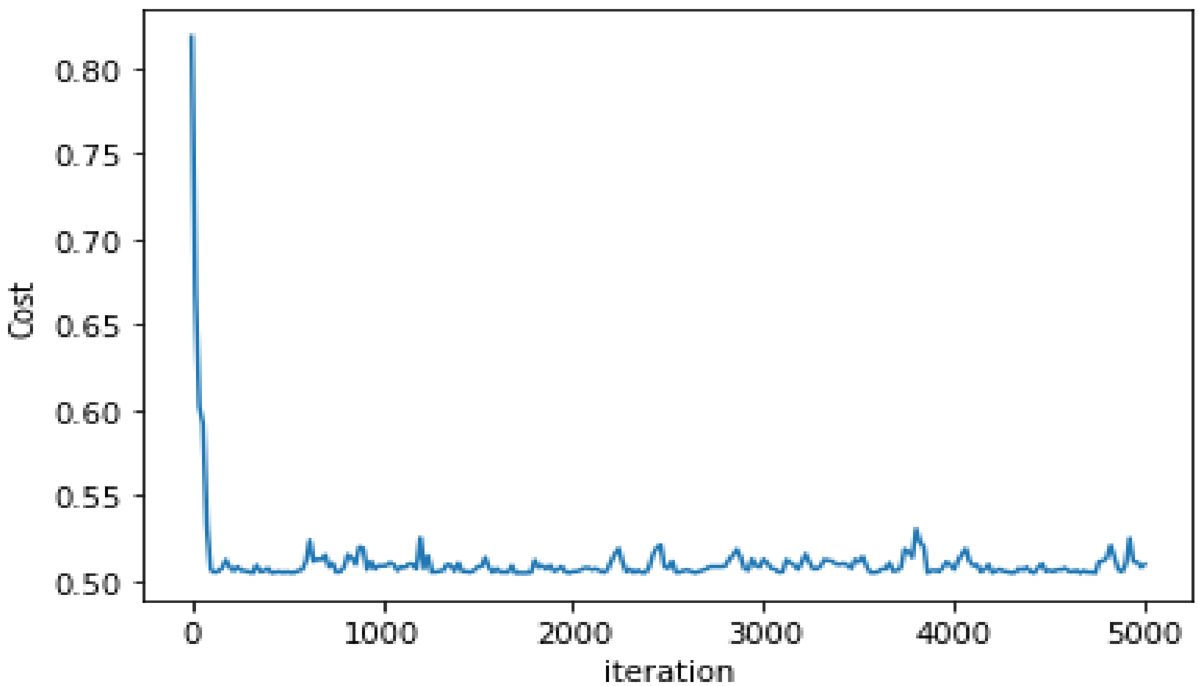
Results

Model ran until 5000 iteration before stopping

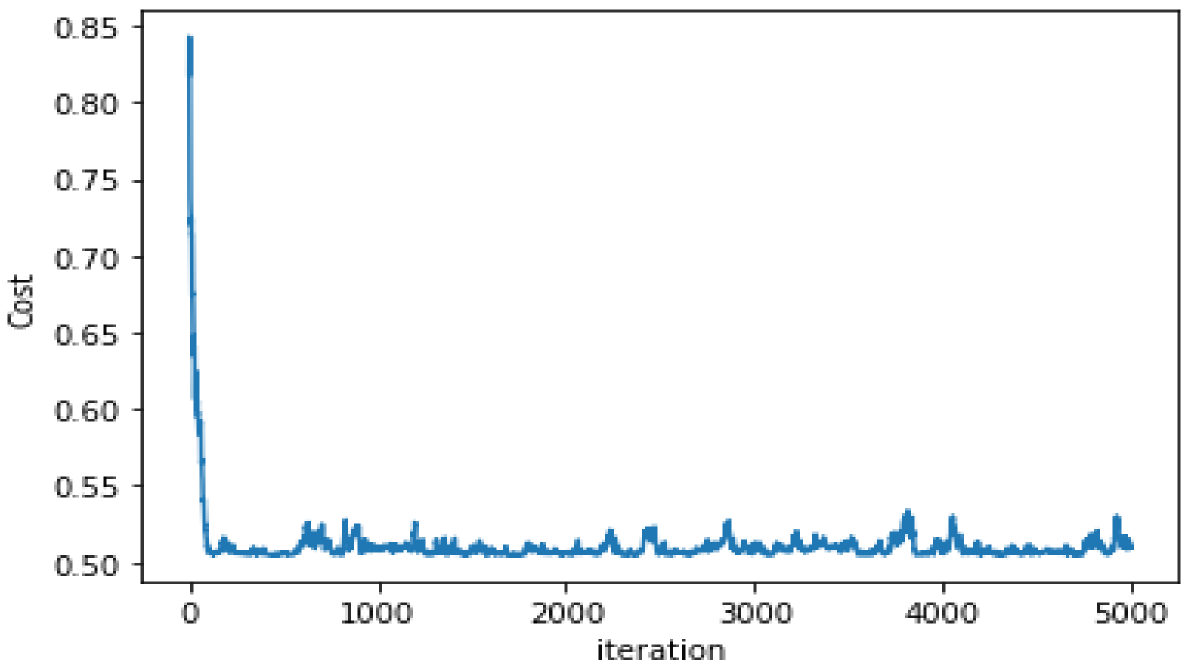
R2 score:	0.01104182
RMSE:	0.9210634150520459

Plots

Cost plotted after every 20th Iteration



Cost plotted after every iteration



Conclusion

Stopping rate 10^{-8} actually leads to overfitting and while 10^{-7} approximately converges to minima value not only faster (420 vs 5000 iteration) but also leads to better R2 score and RMSE.

Plot of cost oscillates a lot as expected.

Linear Gradient Descent

Data is divided into training and validation sets. Training data contains 70 % of the original data whereas validation data that is used for testing contains the other 30%. R^2 and RMS is calculated on testing/validation set.

Weights are randomly initialized.

Linear Gradient Descent class is implemented which take various arguments such as

- alpha (learning rate)
- no of iterations
- regularization_factor (although it is set to 0 as mentioned in the question)
- stop criterion
  - based on weights
    - Training Is stopped if the gradient is very small
  - based on cost
    - Training is stopped if the decrease in cost is very small signaling its near its minima and further iteration wont necessarily decrease cost
- stop rate

Model 1:

Parameters

alpha=0.01, no of iteration=5000, reg_factor=0, stop_rate=0.5

stopping criterion=weights

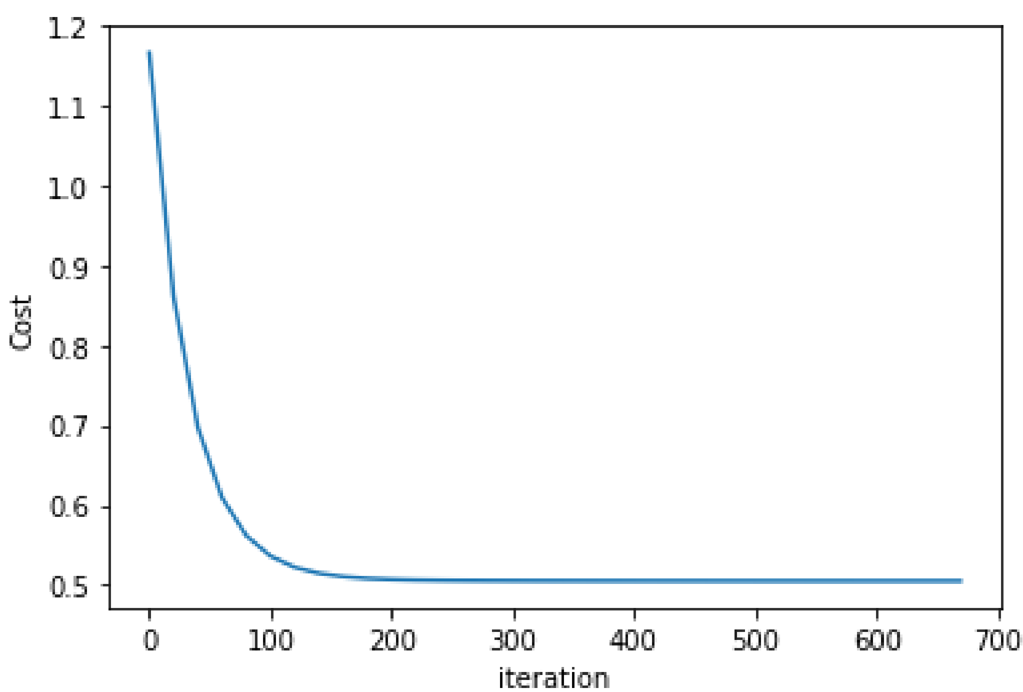
Results

Model ran until 670 iteration before stopping

R2 score	0.03282973
RMSE	0.910860856280399

Plot

Cost plotted after every 20th iteration



Model 2:

Parameters

alpha=0.01, no of iteration=5000, reg_factor=0, stop_rate=10^{-7}

stopping criterion=cost

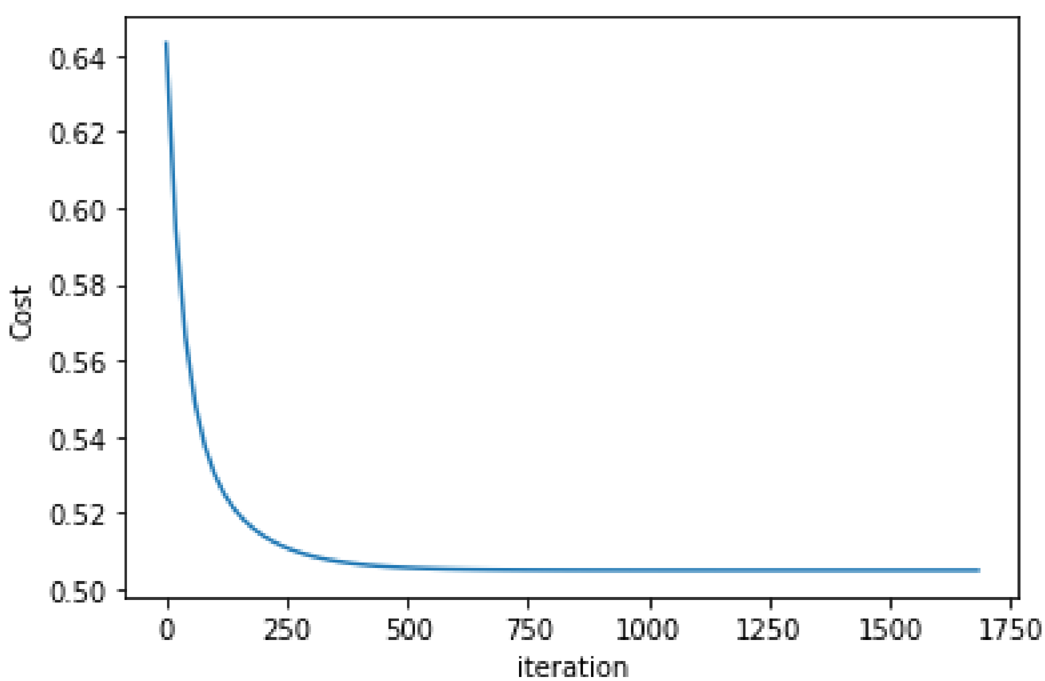
Results

Model ran until 1683 iteration before stopping

R2 score	0.03309544
RMSE	0.9107357290389554

Plot

Cost plotted after every 20th iteration



Gradient Descent with regularization

L1 norm

$$\sum_{i=1}^{n}(Y_i - \sum_{j=1}^{p} X_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

Cost function

$$\hat{\beta}^{\text{lasso}} = \underset{\beta}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^{N} \Big(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\Big)^2 + \boxed{\lambda \sum_{j=1}^{p} |\beta_j|} \right\}$$

Gradient

Parameters

alpha=0.01, no of iteration=5000, reg_factor=0.1, stop_rate=10^{-7}

stopping criterion=cost

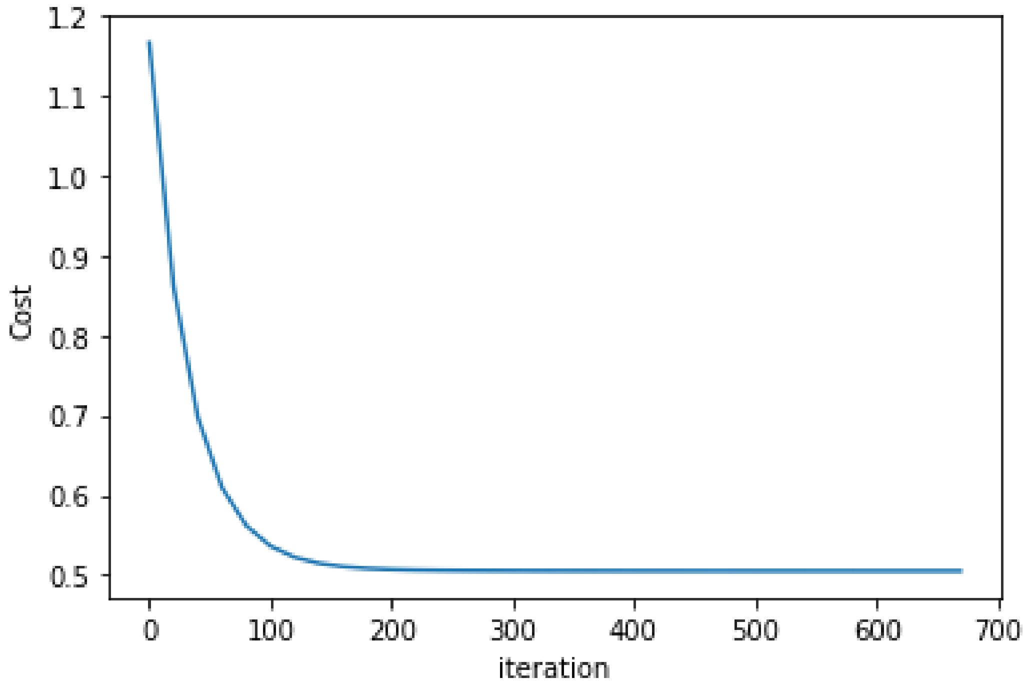
Results

Model ran until 670 iteration before stopping

R2 score	0.03309542
RMSE	0.9107357355480036

Plots:

Cost plotted after every 20th iteration



Validation loss against the regularization coefficient:

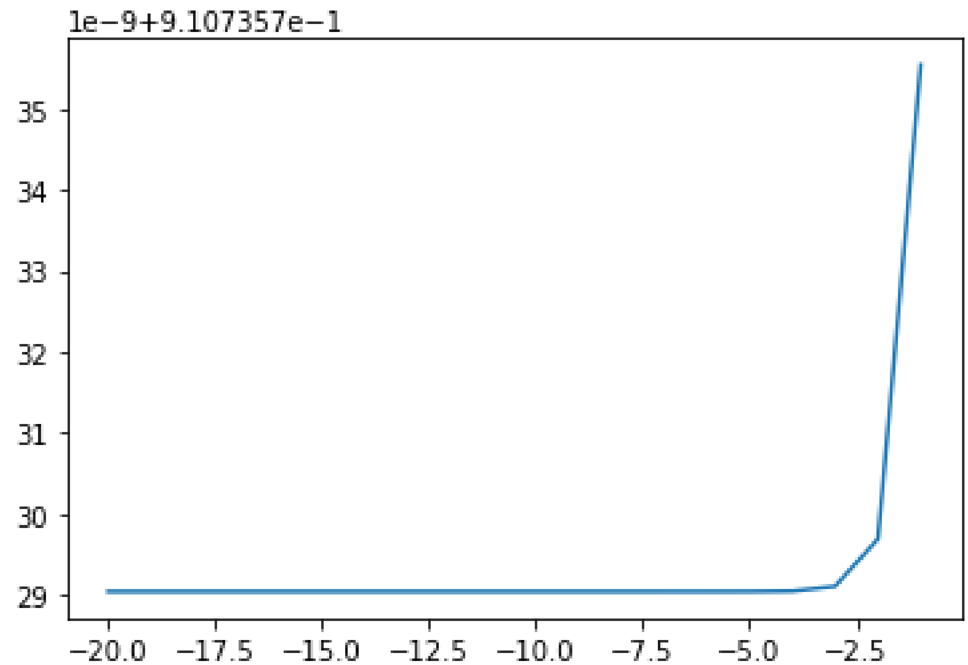
RMSE

[0.9107357290389554,0.9107357290389554,0.9107357290389553,

0.9107357290389618,0.9107357290390203,0.9107357290396062,

0.9107357290454643,0.9107357291040452,0.9107357296898559,

0.9107357355480036])



Regularization coefficient is 10^x

We see cost is fairly the same for regularization coefficient 10^{-3} and then it shoots upwards indicating

10^{-3} as an ideal choice of regularization factor

L2 norm

$$\sum_{i=1}^{n}(y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

Cost function

$$\hat{\beta}^{\text{ridge}} = \underset{\beta}{\text{argmin}}\left\{\sum_{i=1}^{N}\left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \boxed{\lambda \sum_{j=1}^{p} \beta_j^2}\right\}$$

Gradient

Parameters

alpha=0.01, no of iteration=5000, reg_factor=0.1, stop_rate=10^{-7} stopping criterion=cost

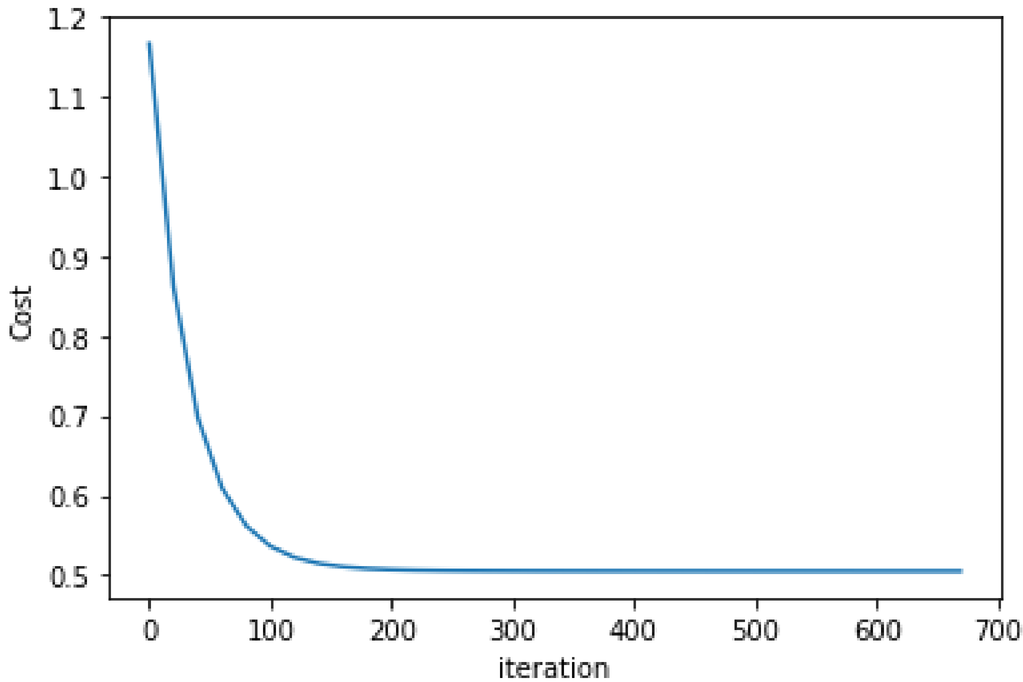
Results

Model ran until 670 iteration before stopping

R2 score	0.03309543
RMSE	0.9107357316340248

Plots:

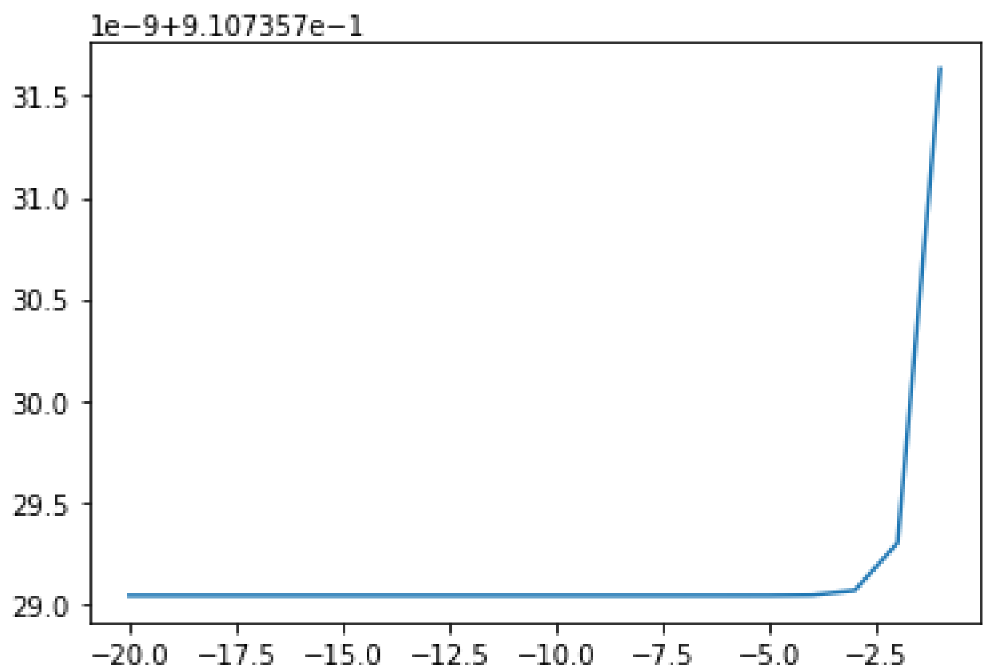
Cost plotted after every 20^{th} iteration



Validation loss against the regularization coefficient

RMSE

[0.9107357290389554, 0.9107357290389554,0.9107357290389554,
0.9107357290389578,0.9107357290389813,0.9107357290392147,
0.9107357290415504,0.910735729064906,0.9107357292984618,
0.9107357316340248]



Regularization coefficient is 10^x.

Similar to l1 norm we see cost is fairly the same for regularization coefficient 10^{-3} and then it shoots upwards indicating 10^{-3} as an ideal choice of regularization factor

Conclusion

l1 vs l2

- For regularization factor = 0.1, training on the dataset provided we observe that both l1 and l2 converge to min cost fairly quickly due to early stopping and surprisingly for the random weights that were initialized, both l1 and l2 led to early stopping at 670^{th} iteration
- Validation loss vs regularization factor plot is virtual similar for both of them, the RMSE cost varies albeit the difference is very small

Normal Equations Method

Data is divided into training and validation sets. Training data contains 70 % of the original data whereas validation data that is used for testing contains the other 30%.

It is implemented by making a class vectorized_linear_gradient_descent which can also take regularization factor as an argument

$$\theta = \left(X^T X\right)^{-1} . \left(X^T y\right)$$

Results

R2 score	0.03284265
RMSE	0.9108547718760206

Stochastic Gradient Model 1	R2 score	0.03129808
	RMSE	0.9115818060933101
Stochastic gradient Model 2	R2 score	0.01104182
	RMSE	0.9210634150520459
Linear gradient descent Model 1	R2 score	0.03282973
	RMSE	0.910860856280399
Linear gradient descent Model 2	R2 score	0.03309544
	RMSE	0.9107357290389554
L1 regularization	R2 score	0.03309542
	RMSE	0.9107357355480036
L2 regularization	R2 score	0.03309543
	RMSE	0.9107357316340248
Normal equation	R2 score	0.03284265
	RMSE	0.9108547718760206

L1 :

RMSE	R^2
0.9107357290389554	0.03309544
0.9107357290389554	0.03309544
0.9107357290389553	0.03309544
0.9107357290389618	0.03309544
0.9107357290390203	0.03309544
0.9107357290396062	0.03309544
0.9107357290454643	0.03309544
0.9107357291040452	0.03309544
0.9107357296898559	0.03309543
0.9107357355480036	0.03309542

L2 :

RMSE	R^2
0.9107357290389554	0.03309544
0.9107357290389554	0.03309544
0.9107357290389554	0.03309544
0.9107357290389578	0.03309544
0.9107357290389813	0.03309544
0.9107357290392147	0.03309544
0.9107357290415504	0.03309544
0.910735729064906	0.03309544
0.9107357292984618	0.03309543
0.9107357316340248	0.03309543