COL341 - Assignment 1
Linear Regression

Simran Mahawar- 2020CS10387

1. Basic Implementation

learning rates, [0.1, 0.01, 0.001]
Used both stopping criteria - maxit and reltol

Implemented Linear Regression from Scratch by using gradient descent.
Make functions for gradient descent, mean squared error, mean absolute error.
Function gradient_descent_maxit uses maximum iterations = 1000 to give theta, minimum loss.
Function gradient_descent uses and tolerance maximum iterations = 1000 to give theta, uses stopping criterion when the cost function (loss) on validation data drops below a threshold.

We have to train model using our training data and find theta/ coefficients(b_0, b_1, b_2, …, b_n).

$$Y_i = b_0 + b_1 X_{1i} + b_2 X_{2i} + b_n X_{ni}$$

Y_i = dependent variable
b_0 = Intercept
$b_1 \ldots b_n$ = Coefficient of Regression
$X_{1i} \ldots X_{ni}$ = independent variable

```
def gradient_descent(X,y,rate, threshold, X_val, y_val):
   Theta = theta_initial(X)
   N = len(X)
   rel_cost = -sys.maxsize
   i = 0
   trainLoss = []
   validationLoss = []
   while rel_cost < threshold and i < 1000:
       gradients = 2/N * X.T.dot(X.dot(Theta) - y)
       theta_new = Theta - rate * gradients
       rel_cost = RelDec(X_val, y_val, Theta, theta_new)
       trainLoss.append(Mean_Squared_Error(X, y, theta_new))
       validationLoss.append(Mean_Squared_Error(X_val, y_val, theta_new))
       theta = theta_new
       i = i + 1
   return theta, trainLoss, validationLoss
```

I trained the model by using these 2 types of gradient descent and computed MSE, MAE by using 3 kinds of leaning rate

```
If we use threshold and keep the learning rate =  0.001
MSE of training data:  0.8915006158550065
MAE of training data:  0.7679174221687569
MSE of validation data:  0.9941722321908697
MAE of validation data:  0.794122763257961
If we don't use threshold and keep the learning rate =  0.001
MSE of training data:  0.3891677686216799
MAE of training data:  0.5014068042425852
MSE of validation data:  0.5634600567431234
MAE of validation data:  0.5629263256519705
```

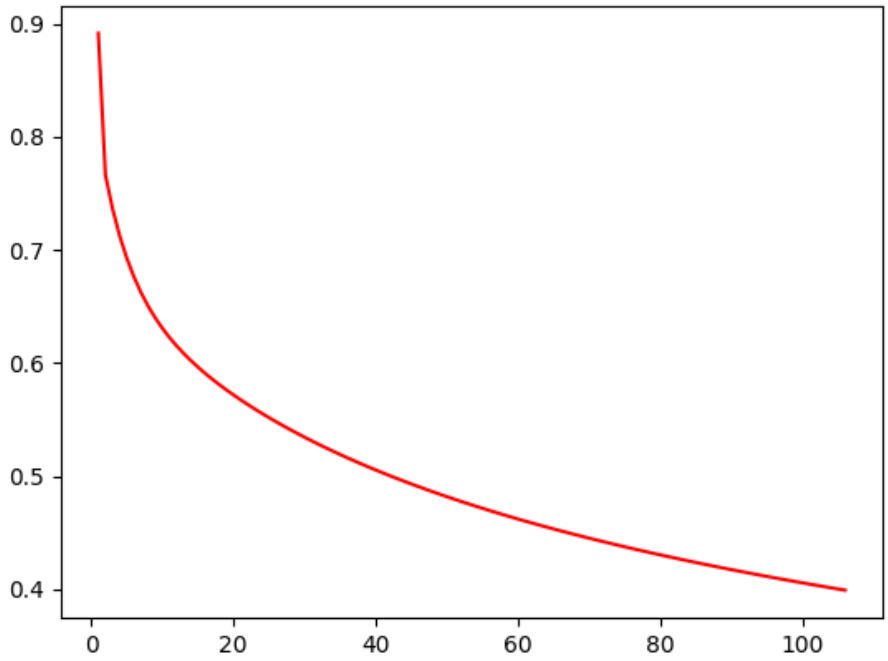
Overflow occurs when learning rate = 0.01 and 0.1

```
If we use threshold and keep the learning rate =  0.01

MSE of training data:  2731.492983024991

MAE of training data:  51.41585445410522

MSE of validation data:  2785.126445457005

MAE of validation data:  52.257682904386826

/home/simran/miniconda3/lib/python3.9/site-packages/numpy/core/_methods.py:180: RuntimeWarning: overflow
encountered in reduce
  ret = umr_sum(arr, axis, dtype, out, keepdims, where=where)
/home/simran/Documents/COL341/A1/run.py:8: RuntimeWarning: overflow encountered in square
  return np.square(X.dot(theta) - y).mean()
If we don't use threshold and keep the learning rate =  0.01

MSE of training data:  30.819444444444443

MAE of training data:  5.347222222222222

MSE of validation data:  31.0

MAE of validation data:  5.380952380952381
```

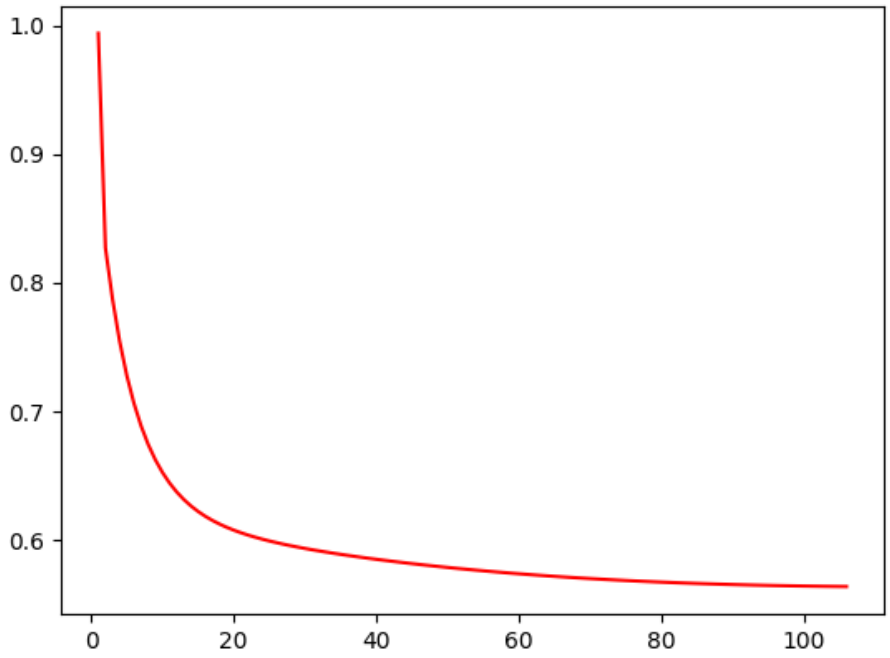
```
If we use threshold and keep the learning rate =  0.1

MSE of training data:  327032.852516894

MAE of training data:  562.2835445410523

MSE of validation data:  332990.7367680798

MAE of validation data:  571.0054004724398

/home/simran/Documents/COL341/A1/run.py:8: RuntimeWarning: overflow encountered in square
  return np.square(X.dot(theta) - y).mean()
/home/simran/miniconda3/lib/python3.9/site-packages/numpy/core/_methods.py:180: RuntimeWarning: overflow
encountered in reduce
  ret = umr_sum(arr, axis, dtype, out, keepdims, where=where)
If we don't use threshold and keep the learning rate =  0.1

MSE of training data:  30.819444444444443

MAE of training data:  5.347222222222222

MSE of validation data:  31.0

MAE of validation data:  5.380952380952381
```

When the rate is 0.001

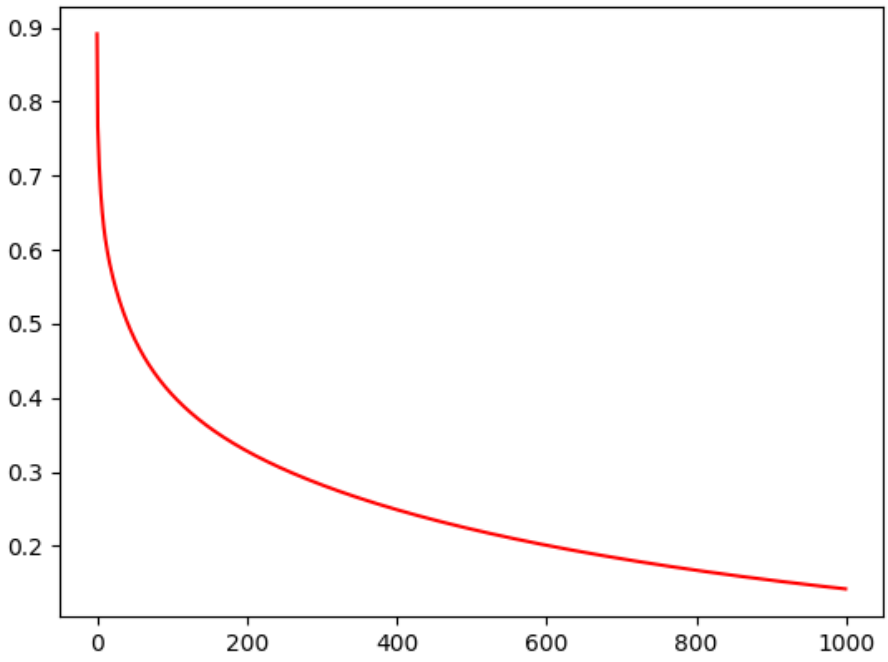
Training error with threshold = 0.0001



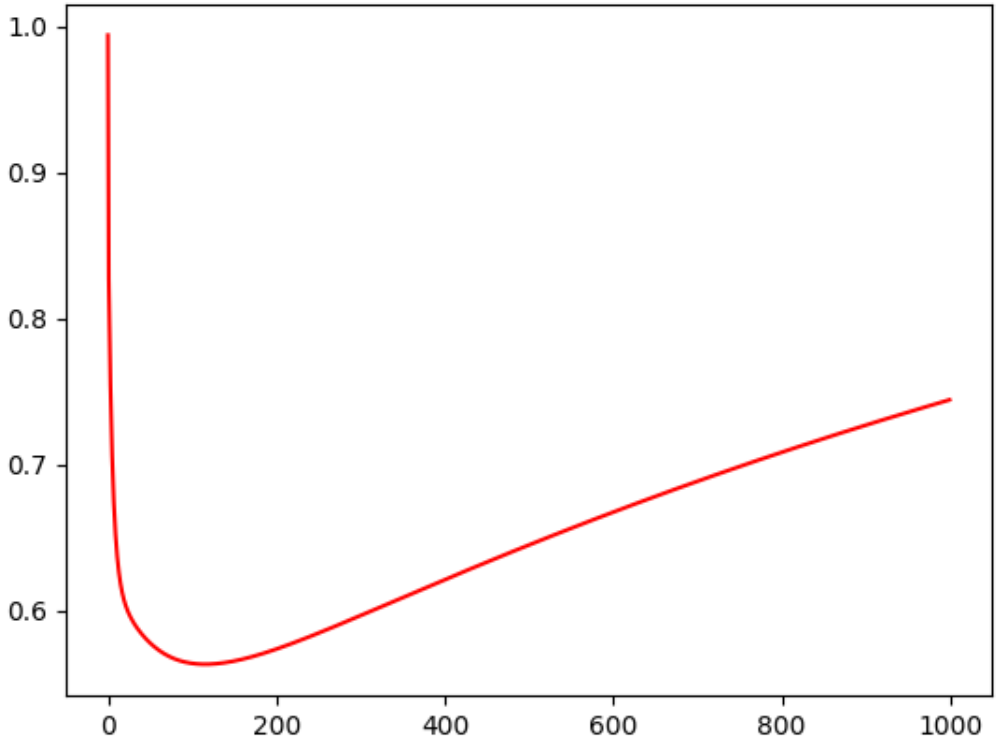
Validation error with threshold = 0.0001



Training error when maximum number of iteration is 1000



Validation error when maximum number of iteration is 1000



2. Ridge Regression

Ridge regression is an extension of linear regression where the loss function is modified to minimize the complexity of the model. We add a regularisation term. Ridge helps normalize ("shrink") Linear Regression Coefficient Estimates. This indicates that the predicted parameters are pushed towards zero to improve their performance on fresh data sets. It allows you to employ sophisticated models while avoiding overfitting.

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\theta(x^{(i)}), y^{(i)}) + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

I made small changes in gradient descent function of linear regression. I add one another term in computing gradient.

```
def ridge_gradient_descent_maxit(X,y,rate, lambda_value, max_iter, X_val,
y_val):
   theta = theta_initial(X)
   bestTheta = theta
   minimum_mse = 100
   N = len(X)
   trainLoss = []
   validationLoss = []
   for i in range(max_iter):
       gradients = np.add((2/N * (X.T.dot(X.dot(theta) - y))),
((2*lambda_value)/N)*(theta.T))
       theta = theta - rate * gradients
       curr_mse = Mean_Squared_Error(X_val, y_val, theta)
       if curr_mse < minimum_mse:
           minimum_mse = curr_mse
           bestTheta = theta
       validationLoss.append(curr_mse)
       trainLoss.append(Mean_Squared_Error(X, y, theta))
   return bestTheta, trainLoss, validationLoss
```

I computed error values for learning rate = 0.0001, 0.001, 0.01

threshold = -0.001

max_iter= 1000

lambda_value = 5

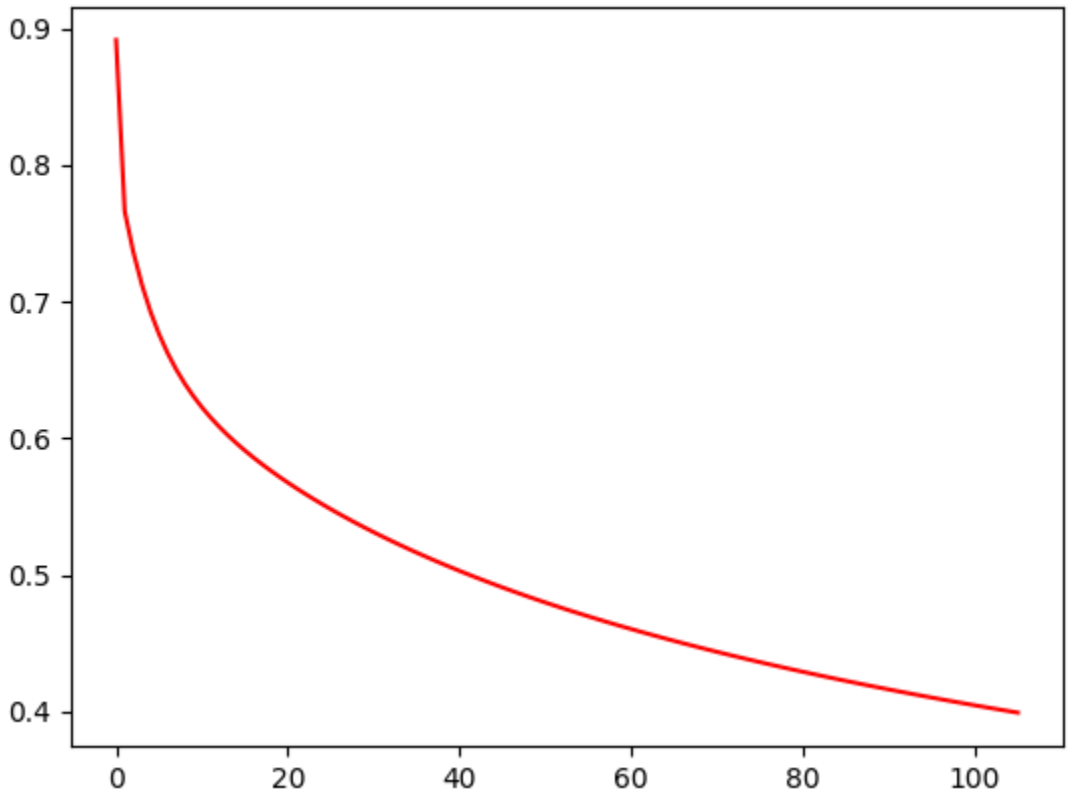
```
If we use threshold and keep the learning rate =  0.0001

MSE of training data:  0.6134818695367497

MAE of training data:  0.6106002077076442

MSE of validation data:  0.6376172233542763

MAE of validation data:  0.6299569496839837

If we don't use threshold and keep the learning rate =  0.0001

MSE of training data:  0.4059841359655576

MAE of training data:  0.5099383650698437

MSE of validation data:  0.5645067693242595

MAE of validation data:  0.5665774881363326
```

```
If we use threshold and keep the learning rate =  0.001

MSE of training data:  0.48412100334420427

MAE of training data:  0.5526155873920438

MSE of validation data:  0.5794623397000855

MAE of validation data:  0.5793937919303972

If we don't use threshold and keep the learning rate =  0.001

MSE of training data:  0.38925311669961304

MAE of training data:  0.5014009967560359

MSE of validation data:  0.563814101796403

MAE of validation data:  0.5629951971182701
```

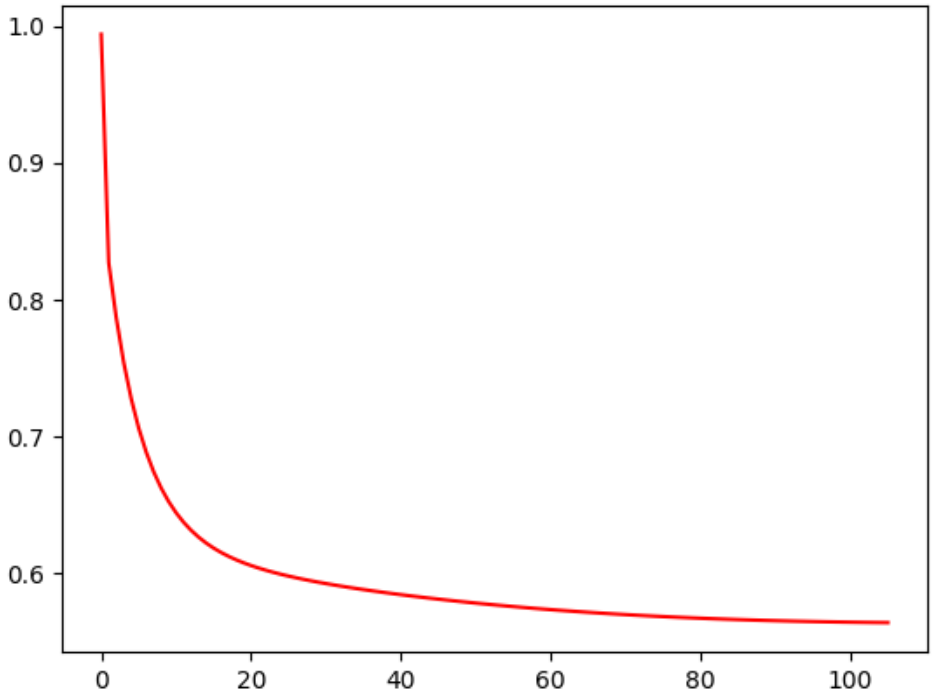
Overflow encountered for learning rate = 0.1

```
If we use threshold and keep the learning rate =  0.01

MSE of training data:  2731.492983024991

MAE of training data:  51.41585445410522

MSE of validation data:  2785.126445457005

MAE of validation data:  52.25768290438682

/home/simran/miniconda3/lib/python3.9/site-packages/numpy/core/_methods.py:180: RuntimeWarning: overflow encountered
  ret = umr_sum(arr, axis, dtype, out, keepdims, where=where)
/home/simran/Documents/COL341/A1/run.py:20: RuntimeWarning: overflow encountered in square
  return np.square(X.dot(theta) - y).mean()
If we don't use threshold and keep the learning rate =  0.01

MSE of training data:  30.819444444444443

MAE of training data:  5.347222222222222

MSE of validation data:  31.0

MAE of validation data:  5.380952380952381
```

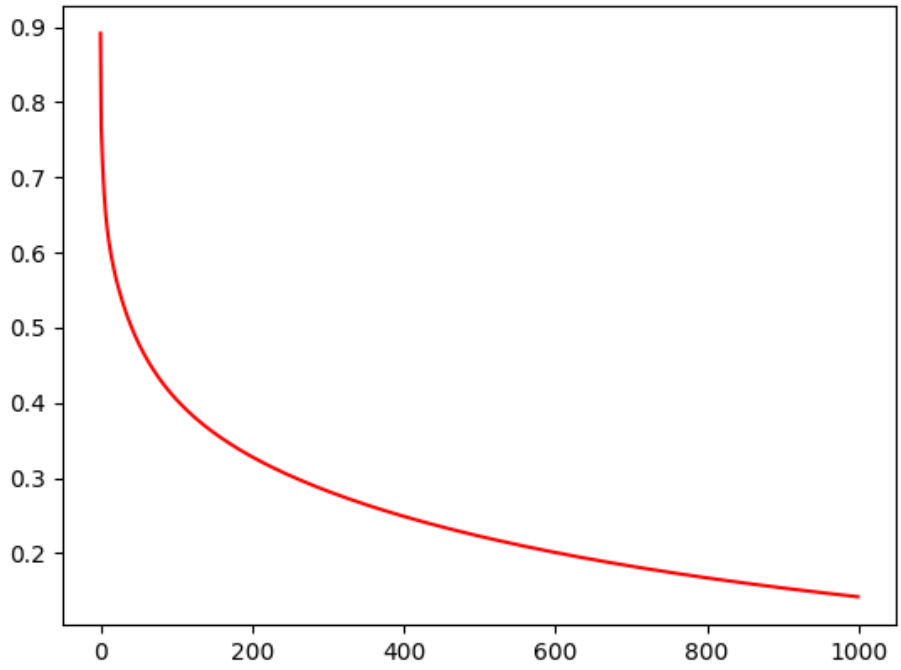
Training error with threshold = 0.0001



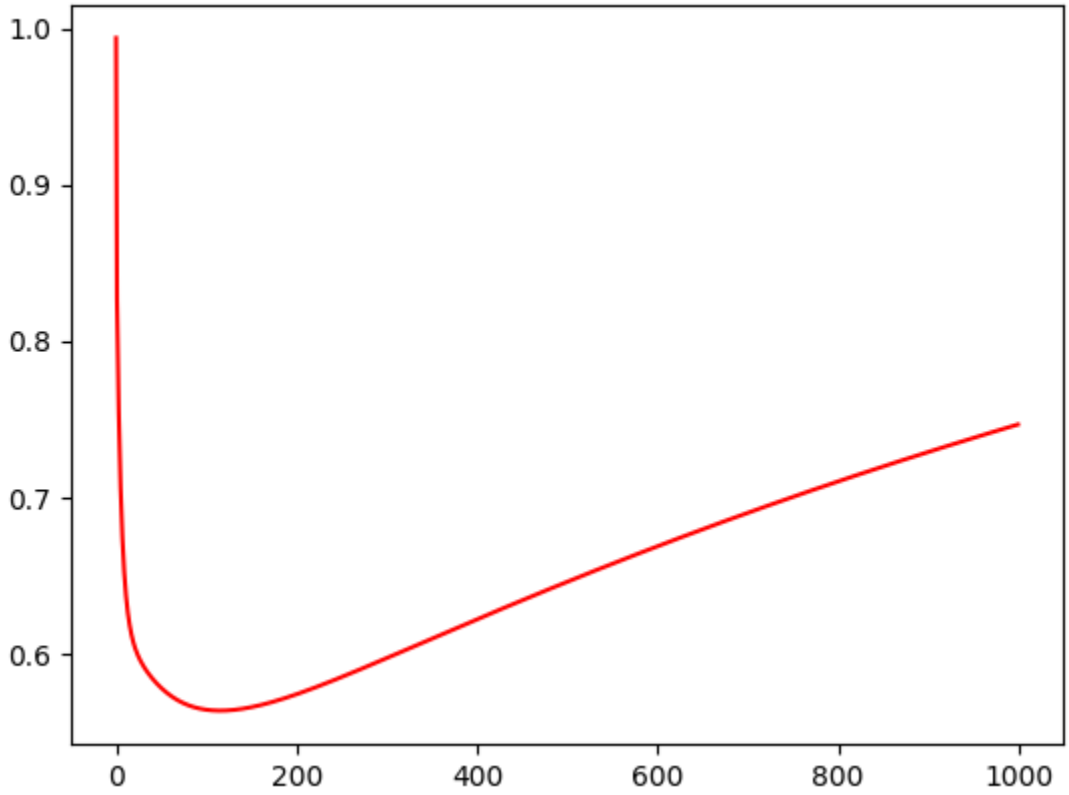
Validation error with threshold = 0.0001



Training error when maximum number of iteration is 1000



Validation error when maximum number of iteration is 1000



3. Using Scikit-Learn Library

```
from sklearn.linear_model import LinearRegression
 ml = LinearRegression()
 ml.fit(X_tr, Y_tr)
 pred = ml.predict(X_val)
```

Imported sklearn library and trained the model using train data.
After running that model on validation data, we got pred as the resulting values of y for validation data. Computed errors on that.

MSE = 0.8178311302215785
MAE = 0.7116702931834721

4. Feature Selection

Used the method SelectKBest to find the 10 best features from these 2048 features. Trained linear regression model using these 10 features and computed the MSE and MAE

The error is increasing after using only 10 best features

MSE new is = 1.961505865701351
MAE new is = 1.1036291546367805

After using selectFromModel and keeping
alpha=0.1, tol=0.001,max_features=10

MSE after using ridge is = 1.0225732879854281
MAE after using ridge is = 0.8308415462378169

5. Classification

Logistic regression can be used as a binary classifier. The cost function for logistic regression is as follows:

$$J(\theta) = -y \log(h_\theta(x)) - (1-y) \log(1 - h_\theta(x))$$

the probability of the score
being equal to one of the classes for each r in 1 to 8 as:

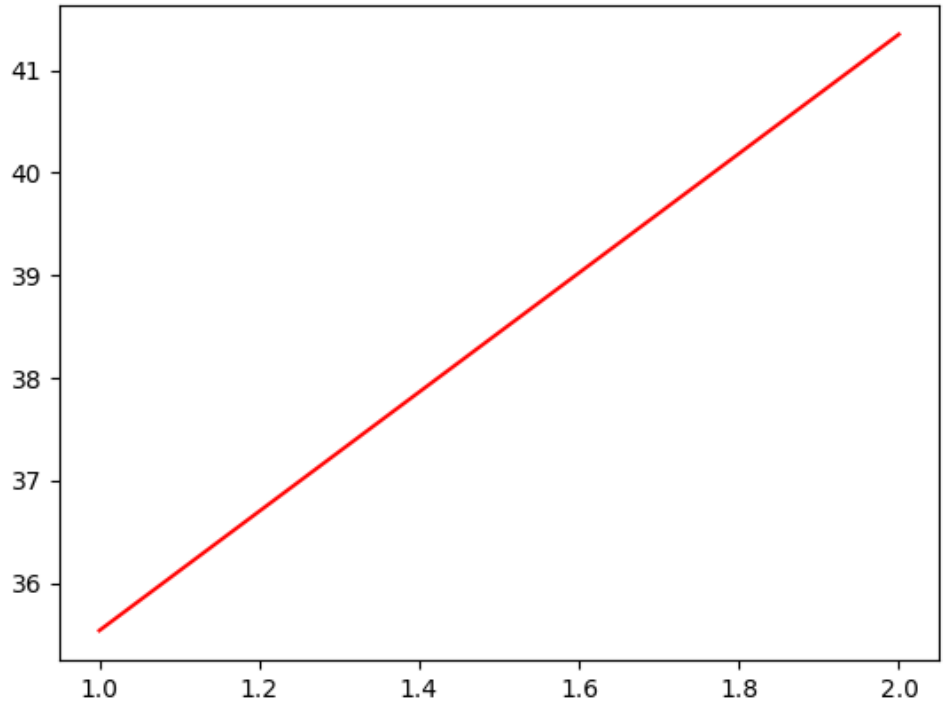
$$h_{\theta_r}(x) = e^{(\theta_r)^T X} / (1 + \sum_{p=1}^{8} e^{(\theta_p)^T X})$$

6. Visualization

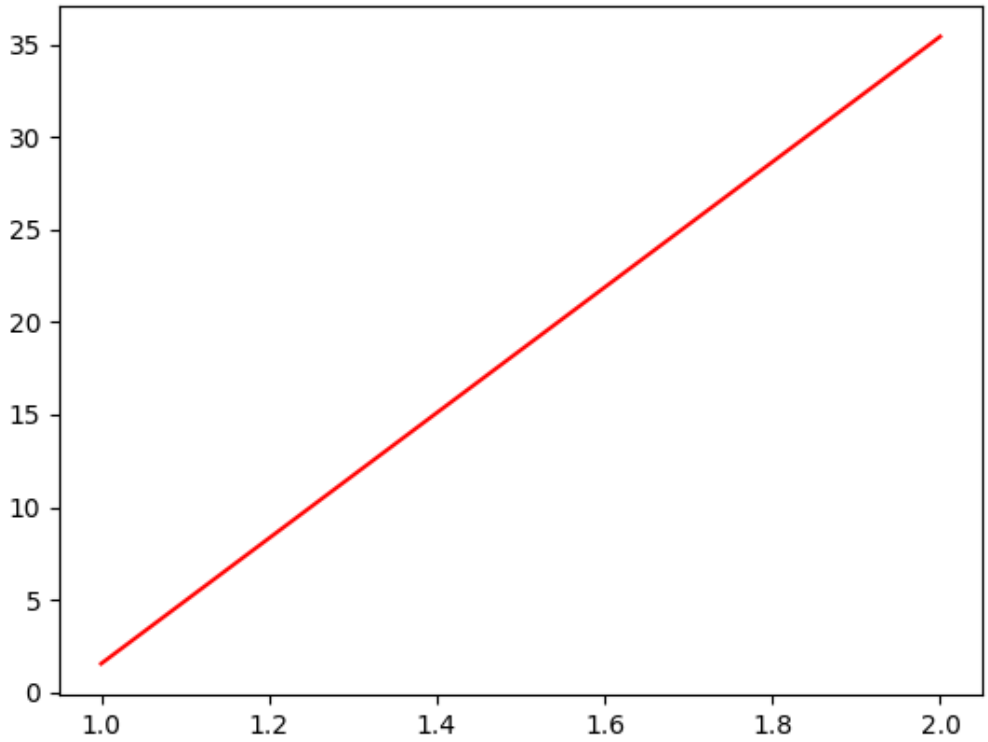
After normalising the training set

In linear regression

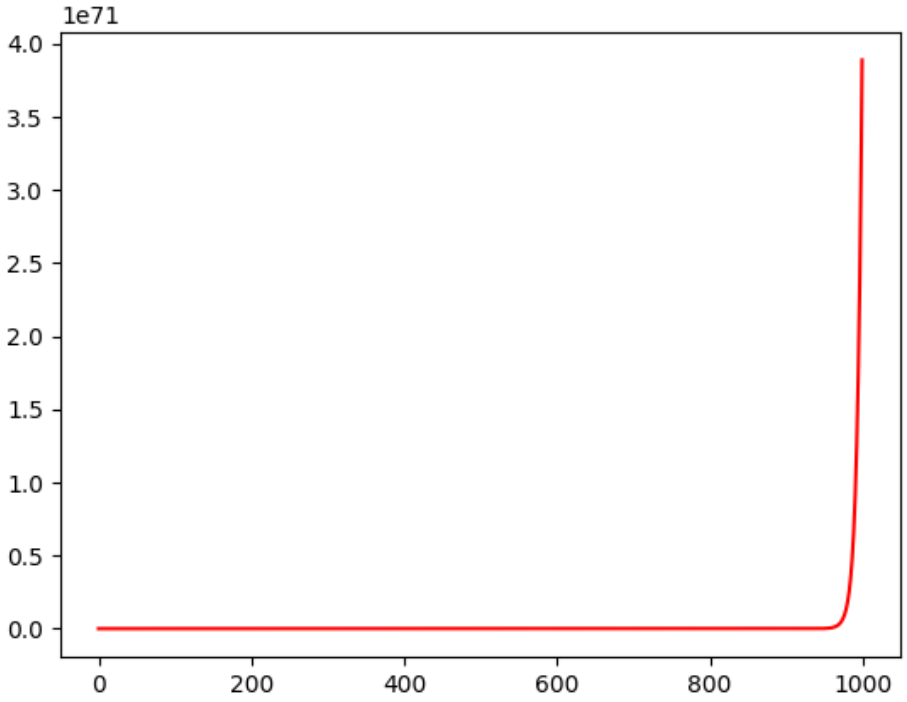
Training error with threshold = 0.0001



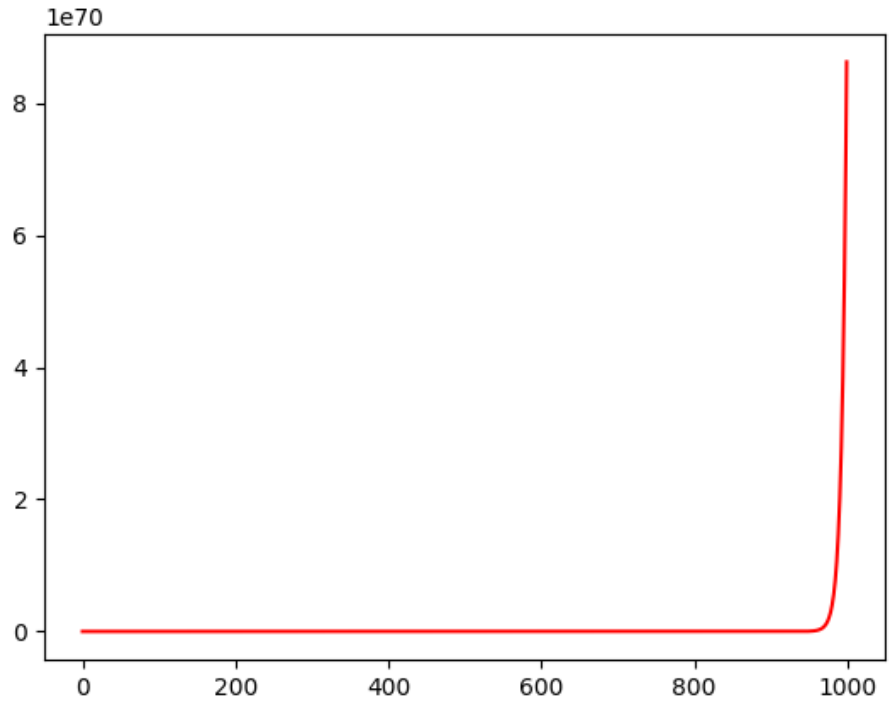
Validation error with threshold = 0.0001



Training error when maximum number of iteration is 1000. Error is increasing after 1000 iterations. It is happening due to overfitting



After normalising the training set

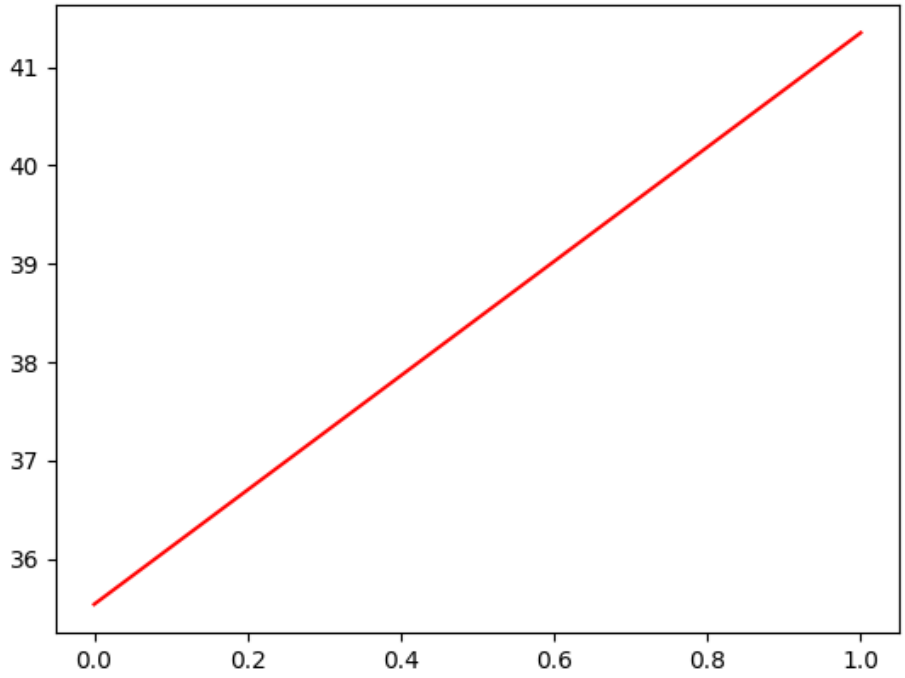



```
If we use threshold and keep the learning rate =  0.001
MSE of training data:  41.35089641784343
MAE of training data:  6.0594798363450275
MSE of validation data:  35.430450990437166
MAE of validation data:  5.70001835429616
If we don't use threshold and keep the learning rate =  0.001
MSE of training data:  30.819444444444443
MAE of training data:  5.347222222222222
MSE of validation data:  31.0
MAE of validation data:  5.380952380952381
```

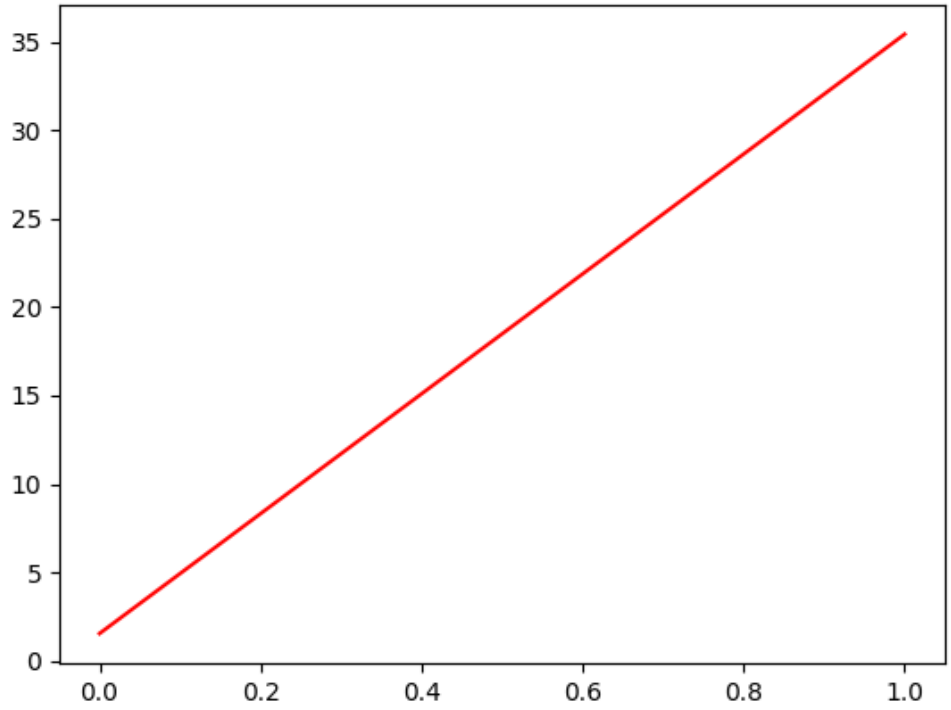
In ridge regression

```
If we use threshold and keep the learning rate =  0.001
MSE of training data:  41.35089738015883
MAE of training data:  6.059479829086687
MSE of validation data:  35.430495440767345
MAE of validation data:  5.700022205928744
If we don't use threshold and keep the learning rate =  0.001
MSE of training data:  35.536939707243405
MAE of training data:  5.405879100115787
MSE of validation data:  1.5405695465416498
MAE of validation data:  0.9525650226677198
```

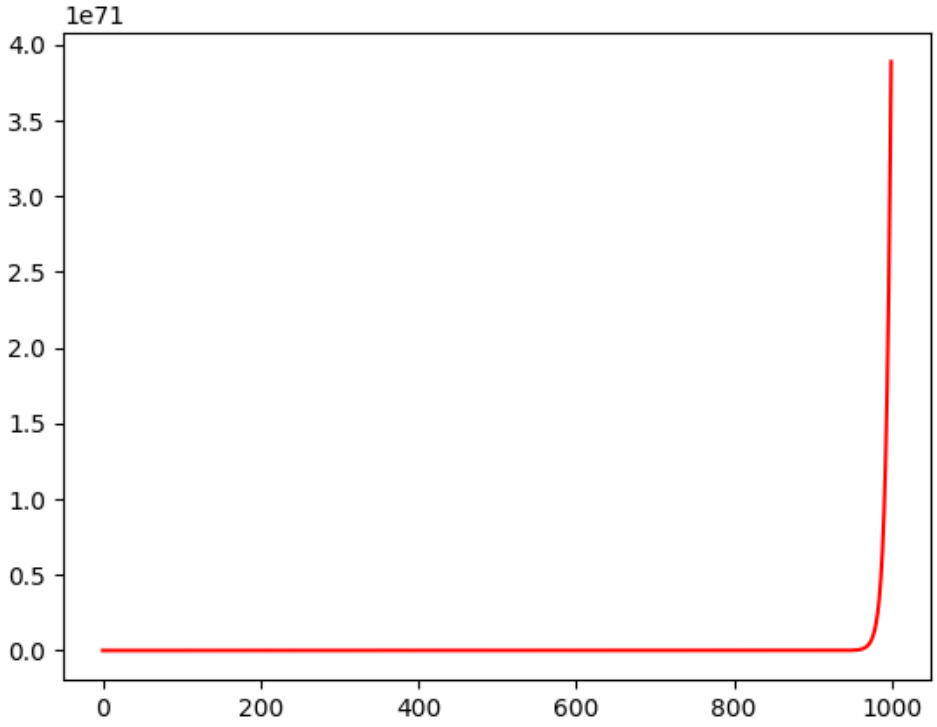
Training error with threshold = 0.0001



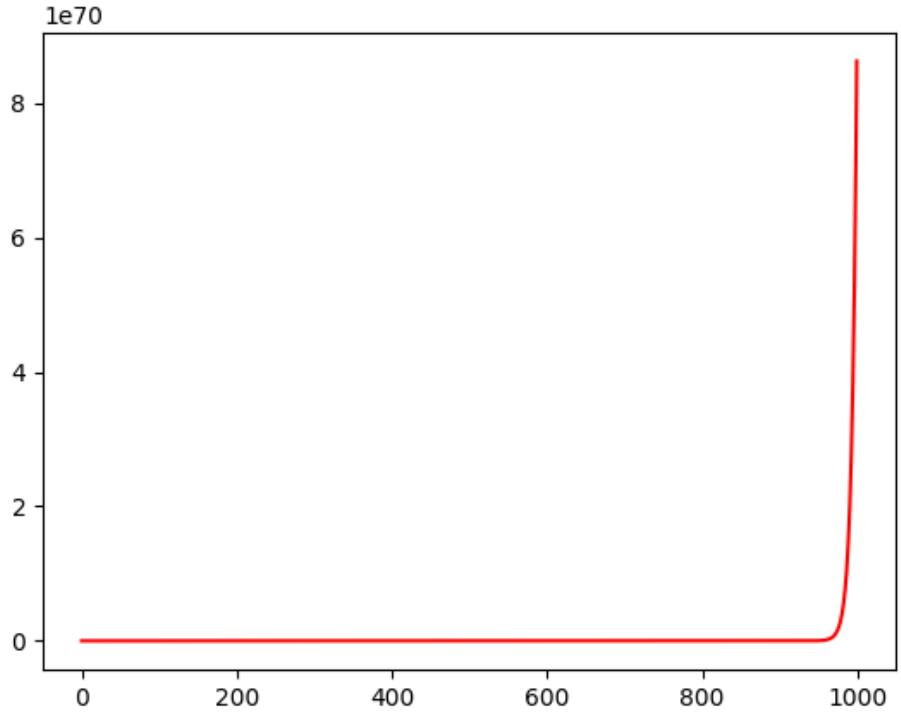
Validation error with threshold = 0.0001

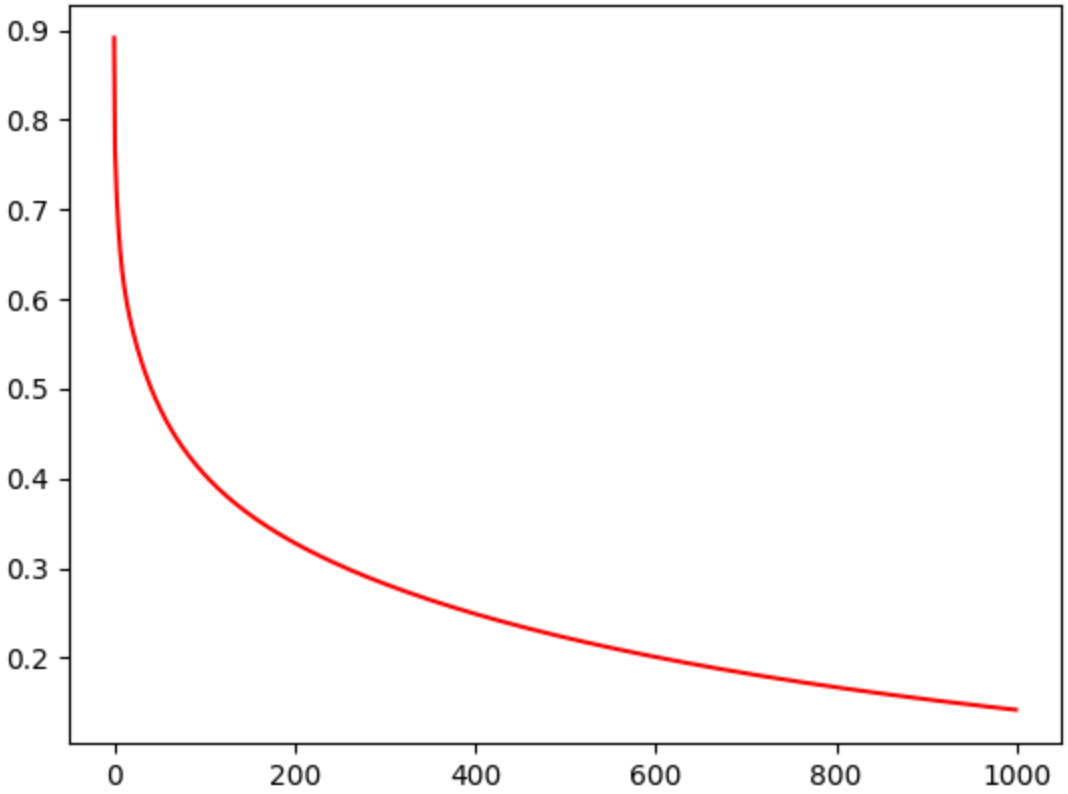


Training error when maximum number of iteration is 1000



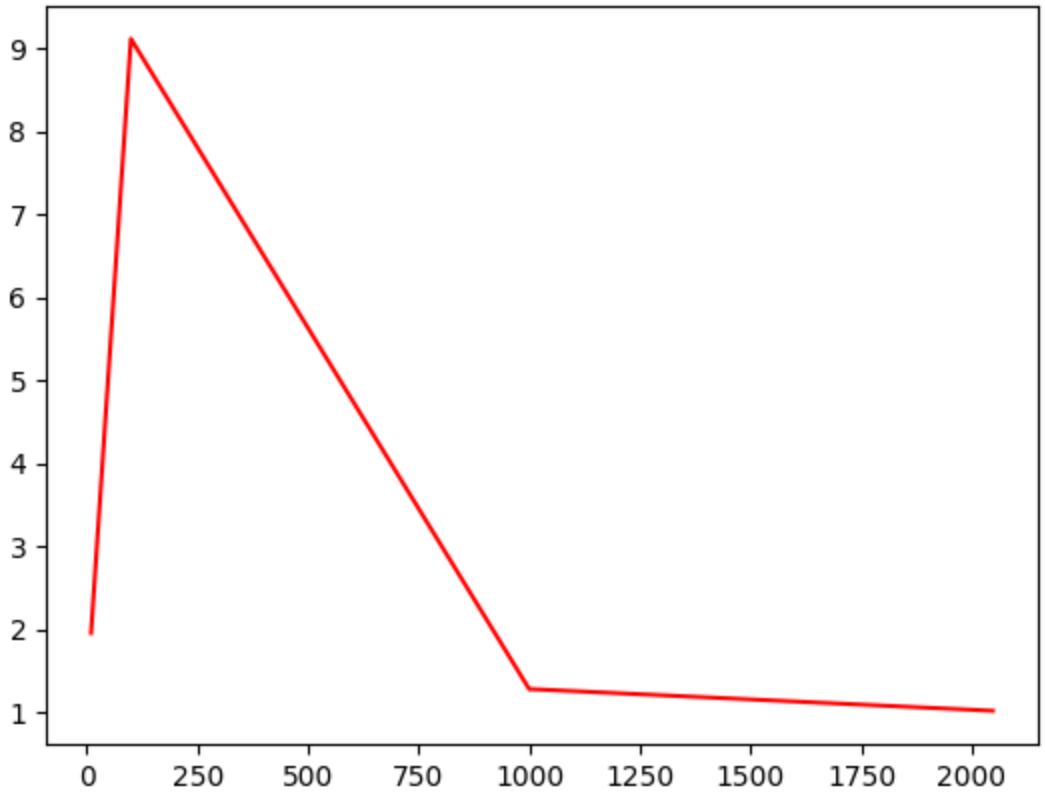
Validation error when maximum number of iteration is 1000





Changing the number of best features and computing MSE

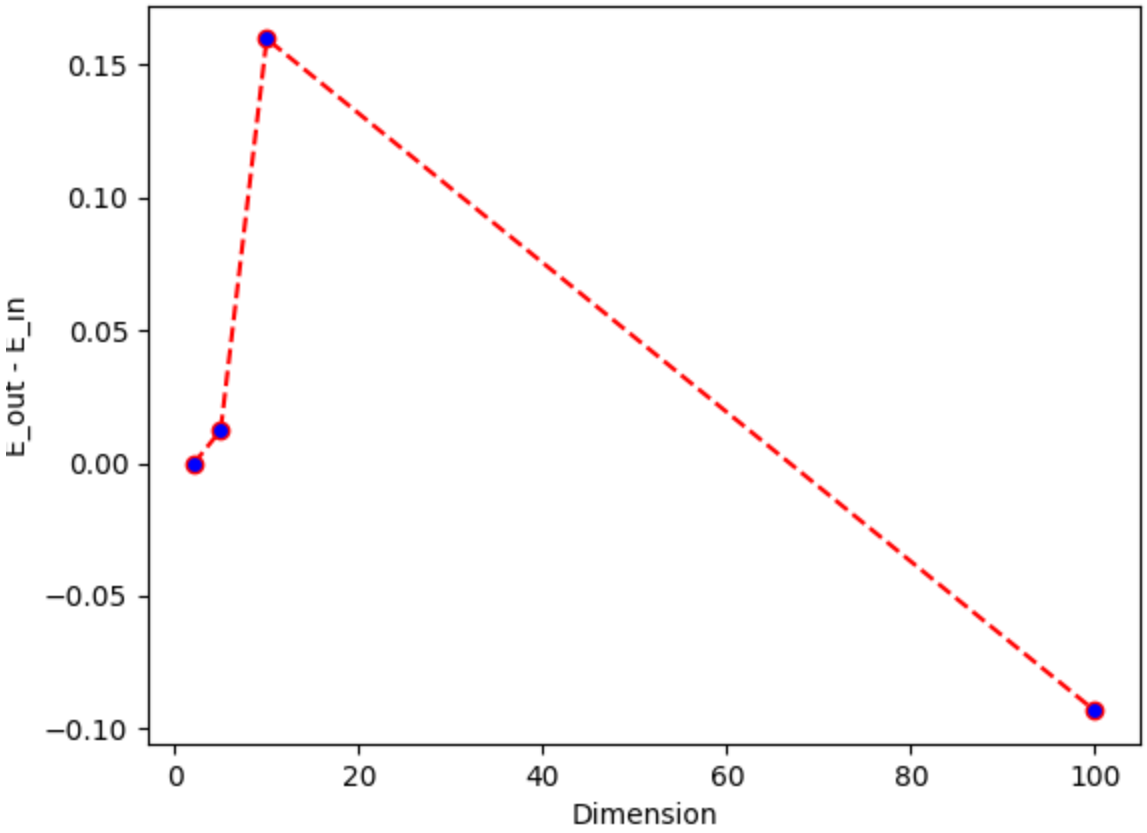
```
number of features is  10
MSE new is =  1.961505865701351
MAE new is =  1.1036291546367805
==========================================================
number of features is  100
MSE new is =  9.110990602922502
MAE new is =  2.378622698808923
==========================================================
number of features is  1000
MSE new is =  1.2870117232197504
MAE new is =  0.9071764363814406
==========================================================
number of features is  2048
MSE new is =  1.0250695782192834
MAE new is =  0.8321119591875601
==========================================================
```



7. Generalization Analysis

$$E_{out} = E_{in} + \mathcal{O}(\sqrt{\frac{d}{N}})$$

For each dimension, I trained model using the train and calculate MSE to find E_{in} and E_{out} out using predictions on train and test files respectively.
And plotted the value of E_{in} and E_{out} out against the dimensions - [2, 5, 10, 100].


0.15
0.10
0.05
0.00
−0.05
−0.10
E_out - E_in
0
20
40
60
80
100
Dimension