CS F320 - Foundations of Data Science

Assignment 2

BY

Name of the Student	ID Number
Rikil Gajarla	***2017A7PS0202H***
Kasuba Badri Vishal	***2017A7PS0270H***

Foundations of Data Science Report

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

I Semester, 2019-20

Abstract

This assignment implements Polynomial Regression using Gradient Descent and Regularization. The assignment involves predicting the altitude using longitude and latitude. The dataset used consists of 434874 data points. 80% of the data is used to train the model and the remaining 20% is used to test the developed model

Python is used to implement the Polynomial regression model using numpy, pandas and matplotlib libraries

Gradient Descent

Gradient Descent is an iterative method used to minimize the loss/error on the training data to best fit the model. Main concept behind Gradient Descent involves in minimizing the loss function which is half of the sum of squares of errors.

Gradient Descent takes steps towards the nearest minima of the error function. In the case of linear regression, the error function is a convex figure and hence, we only have one global minima. Therefore, gradient descent is bound to converge to that point. To find a local minimum of a function using gradient descent, one takes steps proportional to the *negative* of the gradient of the function at the current point.

Parameters:

Degree 1 :

Loss function = $\frac{1}{2}\sum_{n=1}^{N}(y_n - f(x_n))^2$

Initial weights = (1, 1, 1)

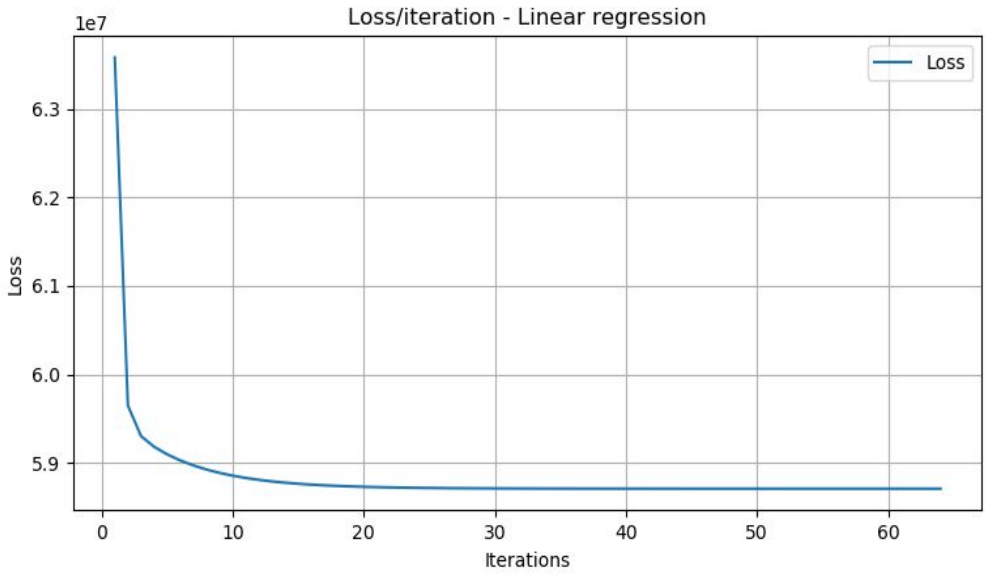
Learning rate = 3.5×10^{-6}

Stopping criteria = 1×10^{-3}

Results:

	MSE	**RMSE**	**R2**
Training error	337.5224	18.37178	0.02618
Testing error	337.2080	18.36322	0.02751

Time taken: 0.57758



Part B - Comparison and Analysis of Models

Results :

Degree : 1 , *learning_rate*=2.23e-6, *stopping_criteria*=1e-4

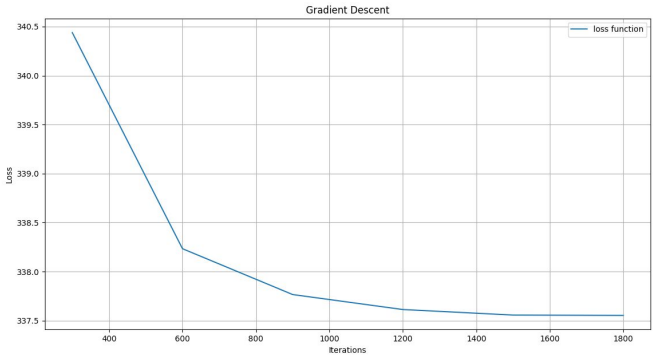
Time to find weights: 5.858181953430176
[21.35423968 13.19894656 -14.05347319]
Train Error(MSE): 337.52539821574777
Test Error(MSE): 337.2219430990912
Train Error(R2): 0.026177728653153265
Test Error(R2): 0.027474540053599505

Degree : 1 , *learning_rate*=2.23e-7, *stopping_criteria*=1e-4
Results:
Weights [21.51301202 12.51184159 -13.58477797]

	MSE	**RMSE**	**R2**
Training error	337.553	18.37	0.0261

Testing error	337.274	18.36	0.0273

Time taken: 74.02531814575195



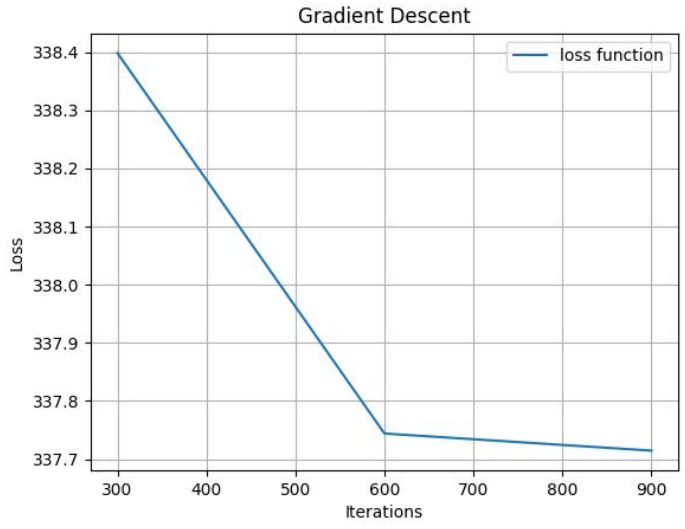
Degree : 2 , *learning_rate*=2.23e-7, *stopping_criteria*=1e-4

Time to find weights: 29.525357246398926

[21.51226631 5.39851129 -7.06343138 -7.06004627 4.54530233 3.1476919]

Results:

	MSE	**RMSE**	**R2**
Training error	337.714	18.376	0.0256
Testing error	337.417	18.369	0.0269



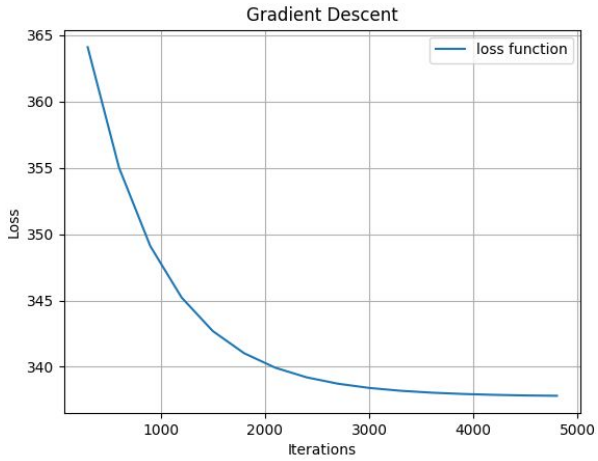
\

Degree : 2 , *learning_rate*=2.23e-8, *stopping_criteria*=1e-4

```
Time to find weights: 98.31647419929504
[21.27678821  4.98433274 -6.494396   -6.50220394  4.21548885  3.37302525]
```

Results:

	MSE	**RMSE**	**R2**
Training error	337.814	18.380	0.0253
Testing error	337.541	18.372	0.0265

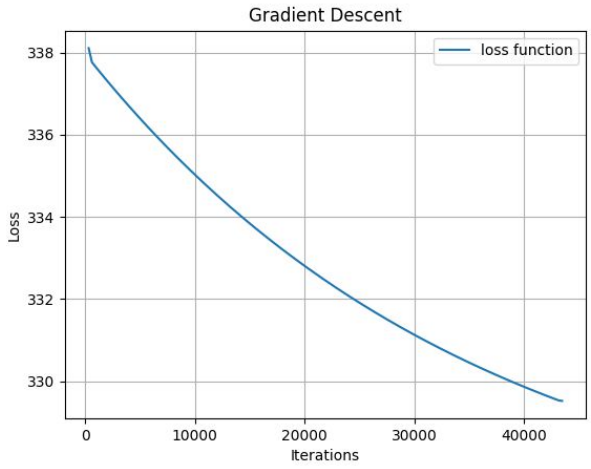


Degree : 3 , *learning_rate*=2.23e-7, *stopping_criteria*=1e-4

```
Time to find weights: 1112.9748485088348
[ 16.45175085  49.96678582  -8.25853255  -6.59578139  44.49343251
 -16.0658742   -4.95516167  39.60088502 -19.23205026 -85.09796613]
```

Results:

	MSE	**RMSE**	**R2**
Training error	329.518	18.153	0.0493
Testing error	329.526	18.153	0.0497

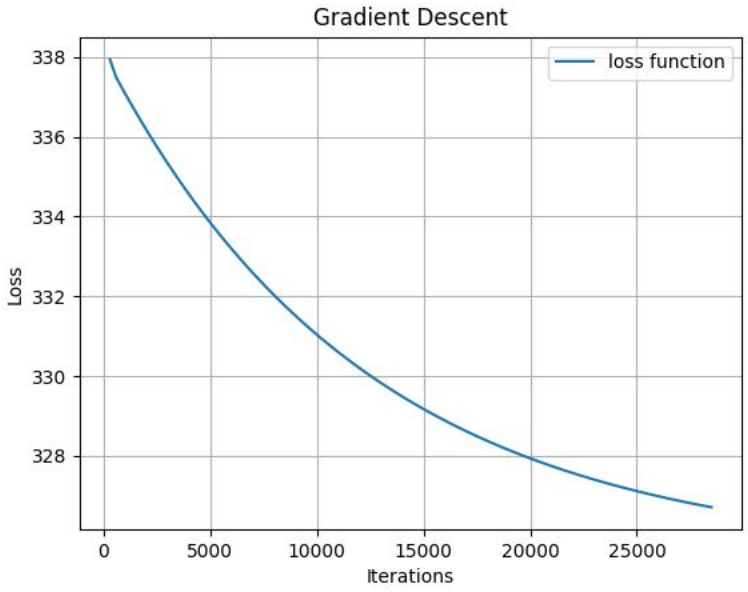


Degree : 4 , *learning_rate*=2.23e-7, *stopping_criteria*=1e-4

```
Time to find weights: 761.8493151664734
[ 13.93889513  38.82611824  -7.48972142  -6.33210685  35.44824952
   4.89698549  -5.18911776  32.44446707   3.05490698 -31.16643011
  -4.06088475  29.77565075   1.34440469 -32.48908517 -68.74405381]
```

Results:

	MSE	**RMSE**	**R2**
Training error	326.715	18.075	0.0574
Testing error	326.873	18.080	0.0573

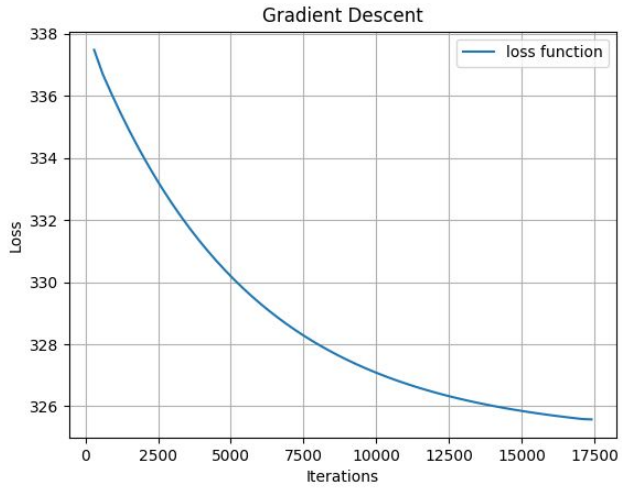


Degree : 5 , *learning_rate*=2.23e-7, *stopping_criteria*=1e-4

```
Time to find weights: 563.9071927070618
[ 12.87685154  26.6088444   -6.33911104  -5.62093548  24.52935872
   9.56184956  -4.91158062  22.68105745   8.4734704   -8.81314604
  -4.21112649  21.03962417   7.46510373  -9.56424108 -28.19431118
  -3.51965168  19.5829765    6.53174521 -10.27251172 -28.77329095
 -48.23587854]
```

Results:

	MSE	**RMSE**	**R2**
Training error	325.578	18.044	0.0606
Testing error	325.803	18.050	0.0604

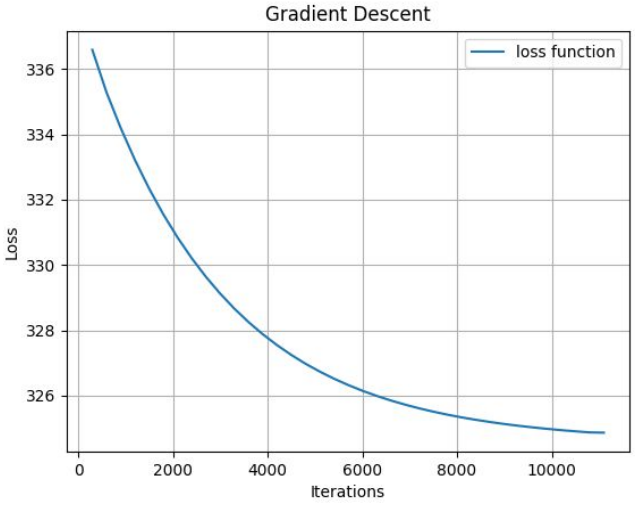


Degree : 6 , *learning_rate*=2.23e-7, *stopping_criteria*=1e-4

```
Time to find weights: 501.582839012146
[ 12.38825813  18.55128095  -5.34632265  -4.88737036  17.1624978
   9.3027084   -4.43395814  15.92591629   8.59488621  -0.78931421
  -3.98613694  14.82526664   7.9387825   -1.26401856 -11.54560461
  -3.54395678  13.84606872   7.33117805  -1.7113248  -11.90119329
 -22.76973059  -3.10746679  12.97570066   6.76876162  -2.13228966
 -12.23972437 -23.0511958  -34.25895588]
```

Results:

	MSE	**RMSE**	**R2**
Training error	324.866	18.024	0.0627
Testing error	325.130	18.031	0.0623



Part C - Gradient Descent Method along with Regularization

L1 Regularization

L1 Regularization, also known as least absolute shrinkage and selection operator (also Lasso) tries to reduce the overfitting of weights to the given data by putting a modulus constraint on the freedom of weights. The constraint is

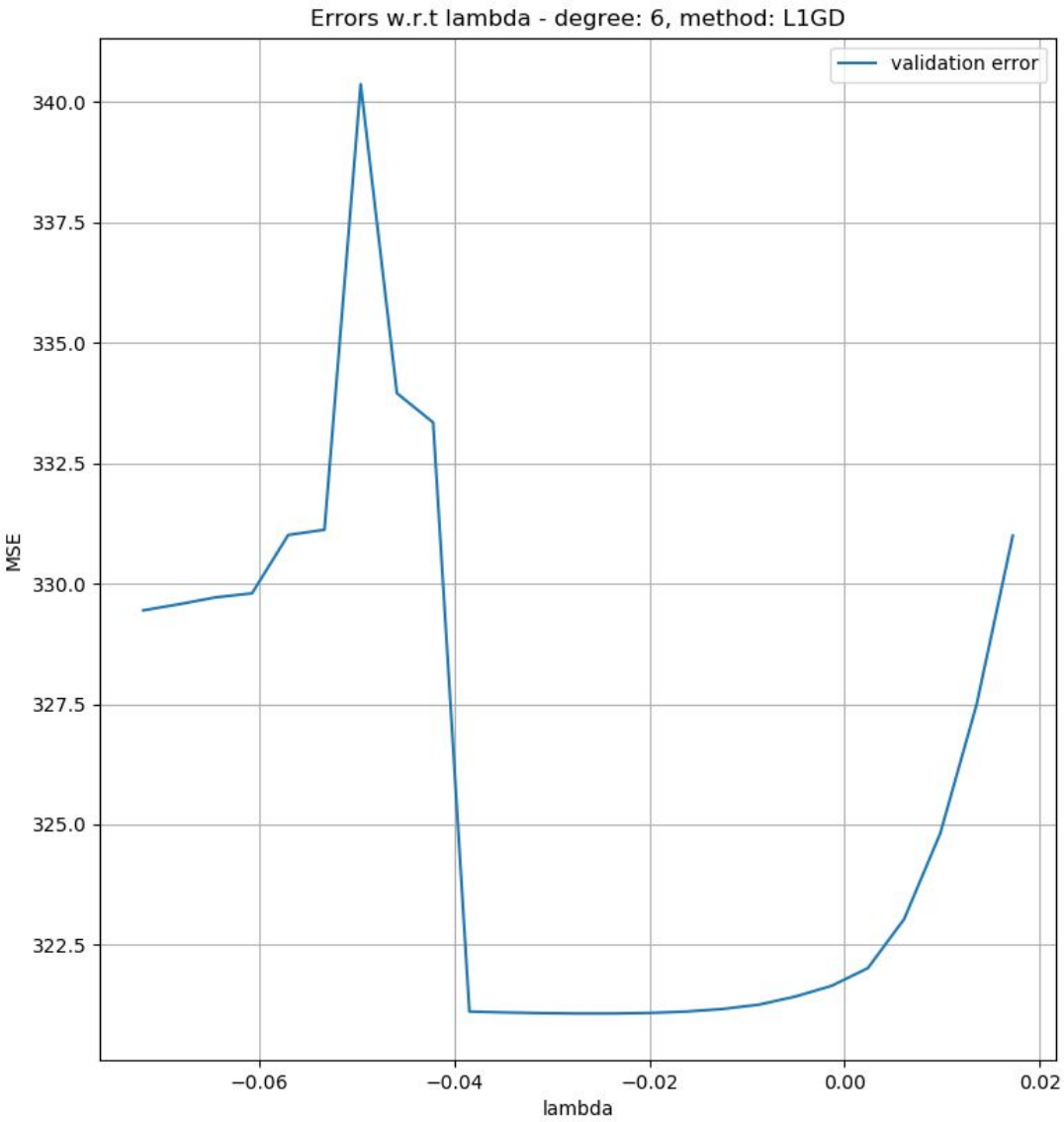
$$\lambda \sum_{i=1}^{D} |W_i| = 1$$

lambda for degree 6 = -0.03736

Result:

	MSE	**RMSE**	**R2**
Train error	324.67932	18.0188601193	0.06324
Test error	324.9320	18.025870298	0.06291

Plot for lambda vs error:



L2 Regularization

L2 Regularization, also known as Tikhonov regularization or ridge regularization tries to reduce the overfitting of weights to the given data by putting a square constraint on the freedom of weights. The constraint is

$$\lambda \sum_{i=1}^{D} W^2 = 1$$

lambda for degree 6 = -0.0373671

Result:

	MSE	**RMSE**	**R2**
Train error	326.58833	18.0717550338	0.057733
Test error	326.8099	18.0778842789	0.057502