Assignment 1

Assignment of
ELL 784: Introduction to Machine Learning

by

Lovlesh Roy

2021ESN2618

Under the Guidance of
Prof. Sumeet Agarwal

INDIAN INSTITUTE OF TECHNOLOGY DELHI

NEW DELHI - 110016

SEPTEMBER 2022

Table of Contents

Part 1A

1. Using 20 data points from training data. Created a design matrix by creating feature vector containing the power of x from 1 to 20.

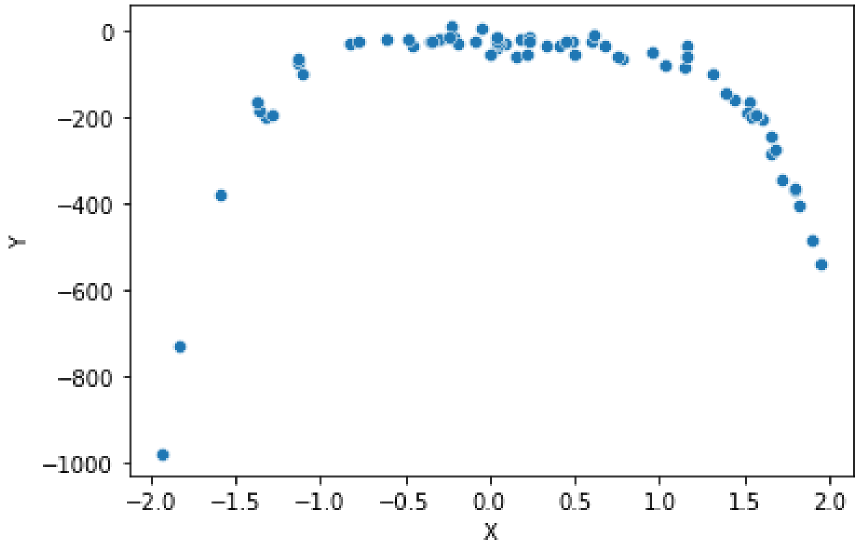


Fig 1: Plot of X and Y (70 data points)

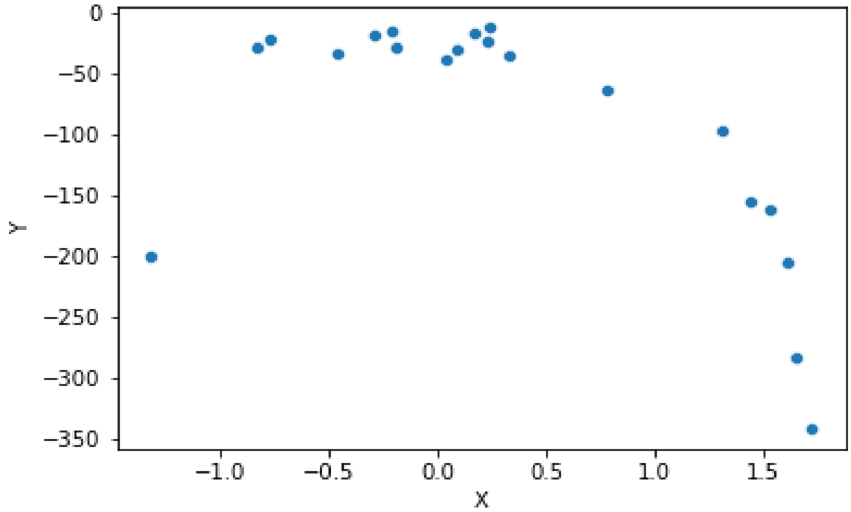


Fig 2: Plot of X and Y (20 data points)

❖ **With 20 Data Points**

1. **1st Order Polynomial**

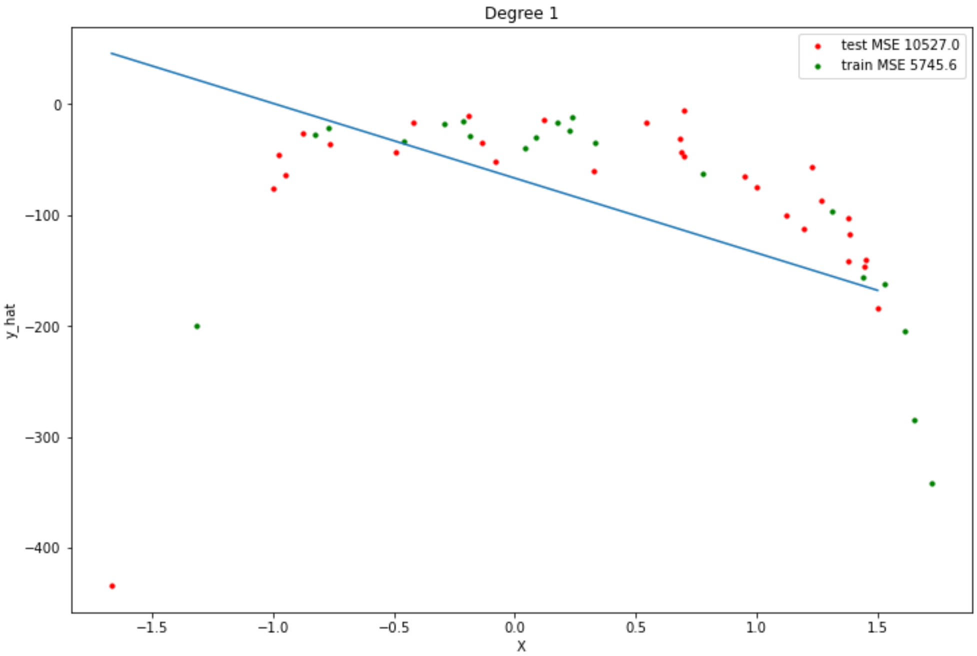


Fig 3: Plot of X and Y

2. **2nd Order Polynomial**

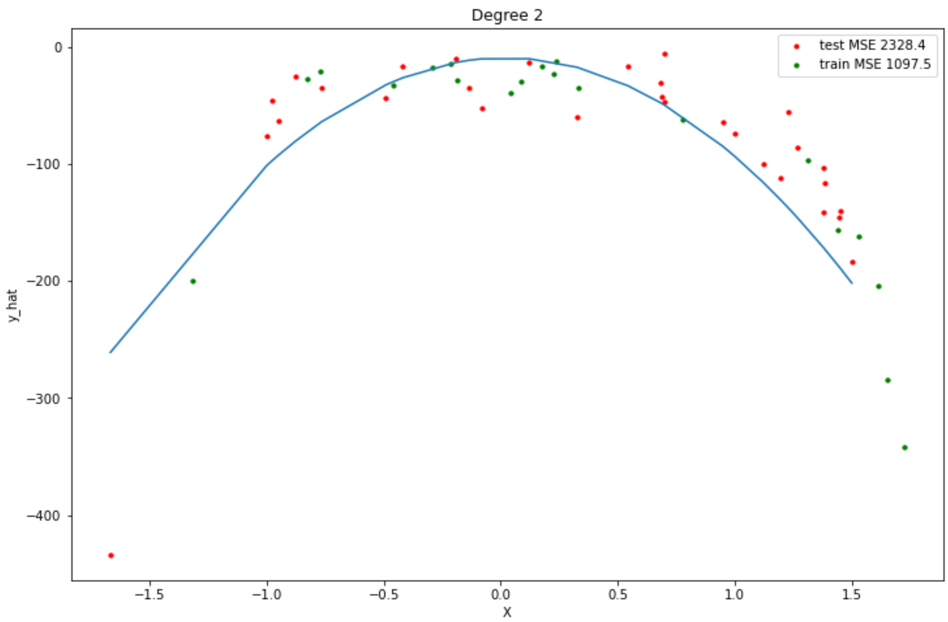


Fig 4: Plot of X and Y

3. **5th Order Polynomial**

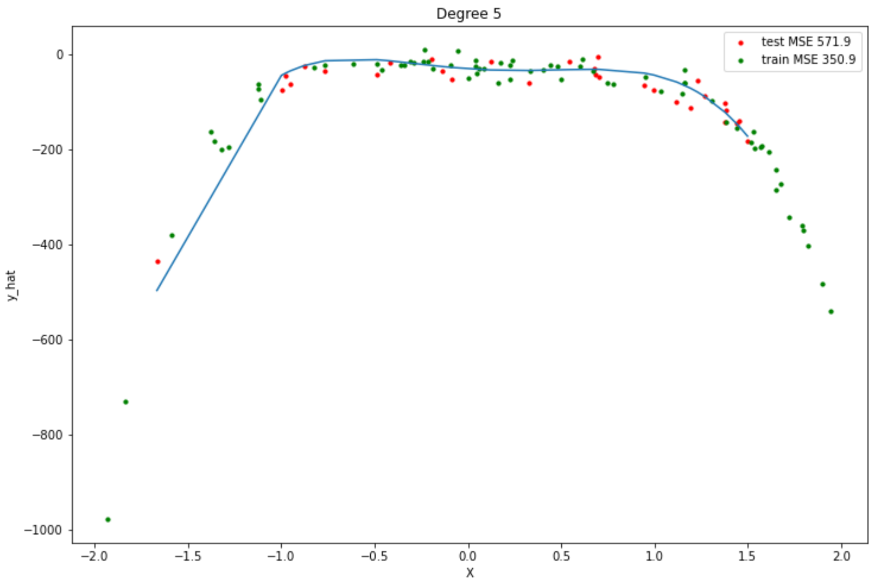


Fig 5: Plot of X and Y

❖ **RMSE plot for train and test data vs degree of polynomial**

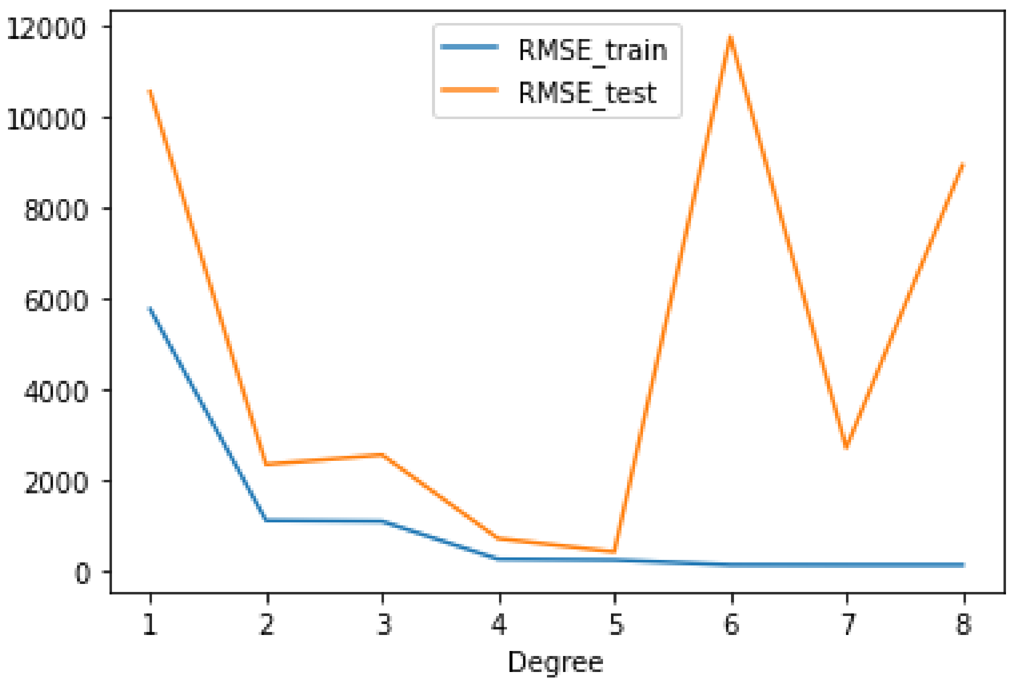


Fig 6: RMSE of train and test dataset vs order of polynomial

From the above plot it can be seen that the 5th order polynomial as a model results in least test error.

- **Analytic Method – Moore-Penrose pseudoinverse**
  For 5th Order polynomial

```
w = np.dot(np.linalg.pinv(X),Y_train)
```

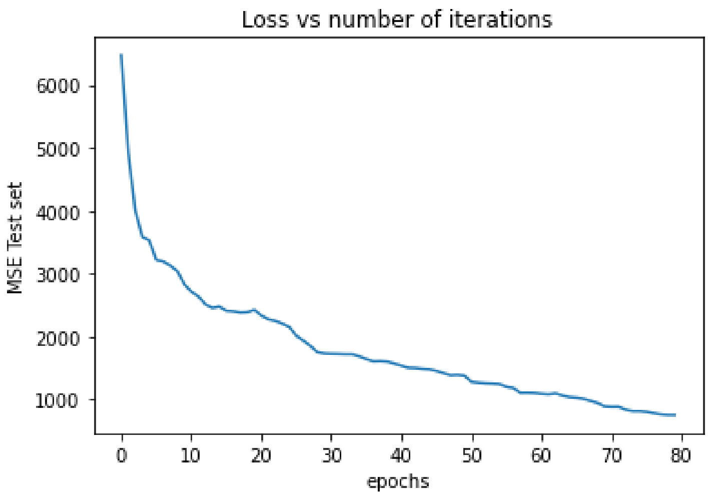
```
w
```

```
matrix([[-25.43180047],
        [-41.73363067],
        [ 12.58554419],
        [ 69.08368564],
        [-47.51844755],
        [-13.38767989]])
```

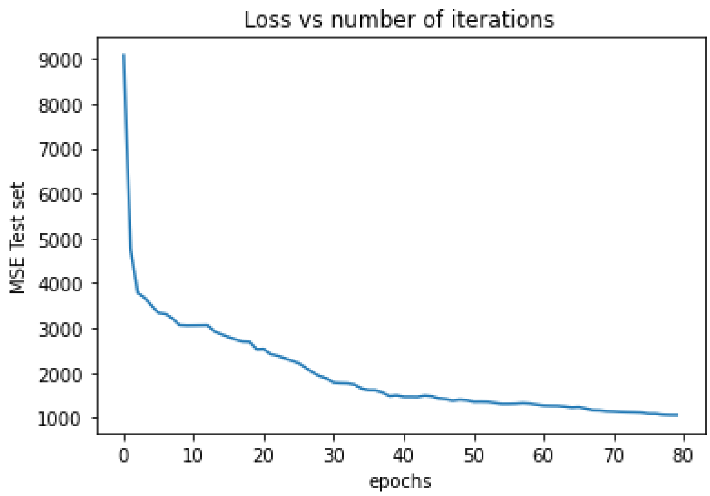
```
coefff= [[-43.40922505,  11.33637161  ,72.74017468 ,-46.48216736, -14.70086315]]
inter= [-25.32400518] #20 Data points
```

Fig 7: Pseudoinverse method for finding the weight matrix

- **Mini batch Gradient Descent Approach**
  As the batch size increases. The plot of test data loss become more smoother



a)



b)

Fig 8: Loss vs number of iterations a) batch size =1, b) batch size =3(learning rate = 0.00025)

- **Comparison of Gradient Descent and Analytic Approach for 5th Order Polynomial**

S.No.	Weights	Analytic Approach	Mini Batch Gradient Descent
1	W0	-25.43180047	-1.768151343539648
2	W1	-41.73363067	4.08281145
3	W2	12.58554419	-13.89811157
4	W3	69.08368564	4.21684195
5	W4	-47.51844755	-28.74041435
6	W5	-13.38767989	-2.90751481
7	Test MSE	407.4	1793.26

Table 1: Solution by Analytic and Mini batch gradient descent approach

❖ **Regularisation – Ridge Regression with 20 data points**

In the least square objective function, l2 norm of weights is added. If we minimize this objective function to fit the curve, its known as ridge regression.

Along with regularisation, comparison with linear, lasso and elastic net model is done.

	y_test	y_hat_linear	y_hat_ridge	y_hat_lasso	y_hat_net
0	-52.110296	-20.936962	-23.936440	-22.866576	-36.634001
1	-35.382308	-38.654300	-45.225751	-43.202845	-46.397639
2	-433.925695	-383.437672	-406.635526	-406.578634	-133.821107
3	-63.249048	-73.181566	-73.665878	-74.003821	-54.971827
4	-116.558703	-117.711720	-127.174528	-123.546285	-138.668867

Fig 9: Comparison of predicted value, where y_test is the actual label

Order of polynomial	model_lin	model_ridge	model_lasso	model_net
1	-41.173774	-7.944801	-18.438717	1.138990
2	-4.576080	1.073629	0.000000	-6.610062
3	78.261316	32.687842	47.799020	0.622407
4	-38.142833	-41.394550	-41.131273	-18.192735
5	-18.782147	-5.632147	-9.773972	-5.146369

Fig 10: comparison of weights with various regularised models

As regularisation parameter lambda increases, weights are penalized more in the objective function. Hence the value of weights decreases while minimizing the objective function. To visualize this change in weights due to change in lambda, a plot is generated.

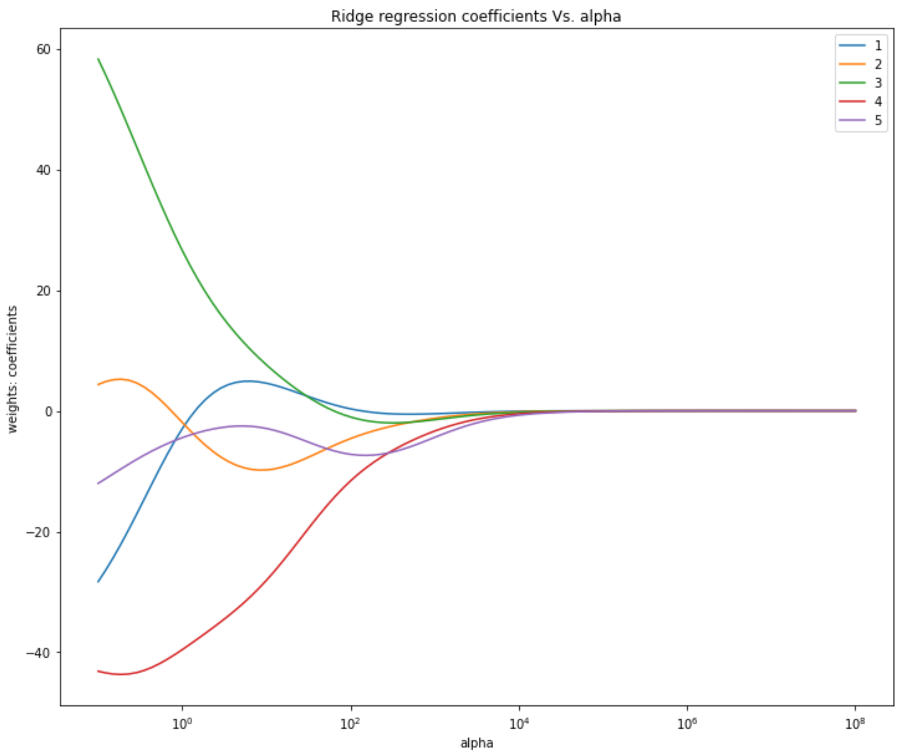


Fig 11: weights vs alpha, where alpha is

- **With 70 Data points**

1. **1st Order Polynomial**

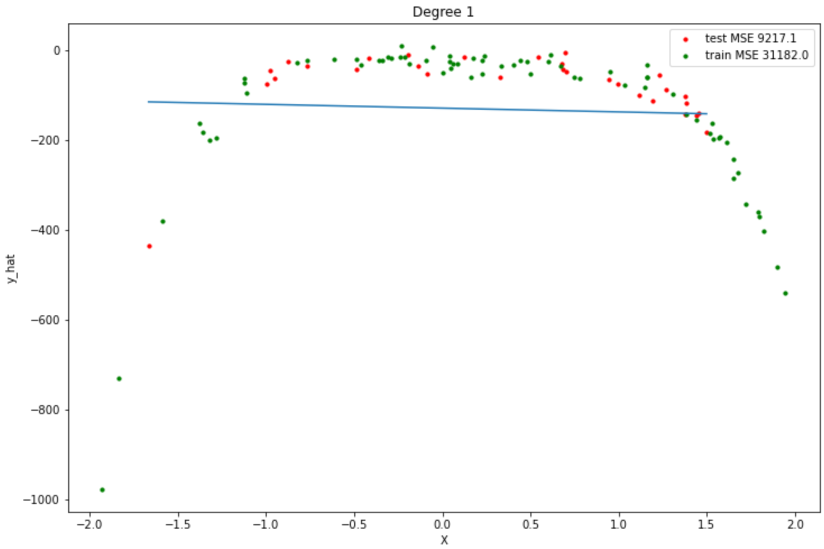


Fig 12: Plot of X and Y

2. **2nd Order Polynomial**

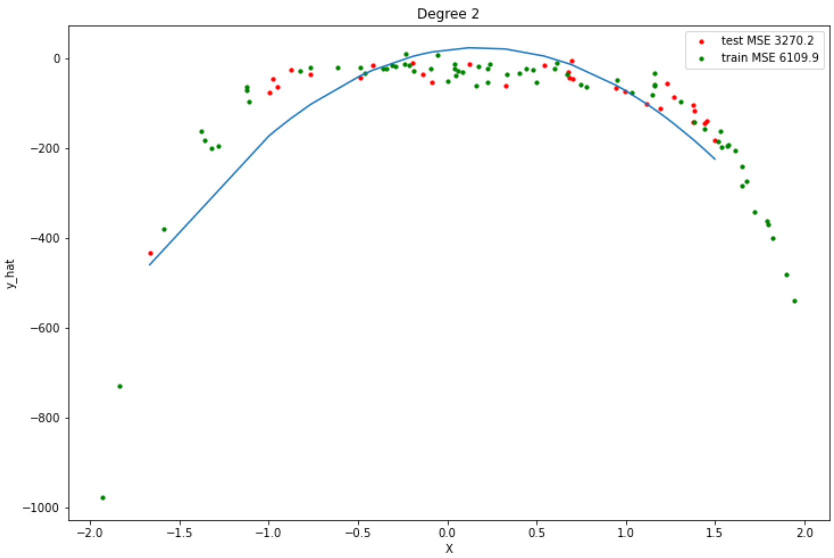


Fig 13: Plot of X and Y

3. **5th Order Polynomial**

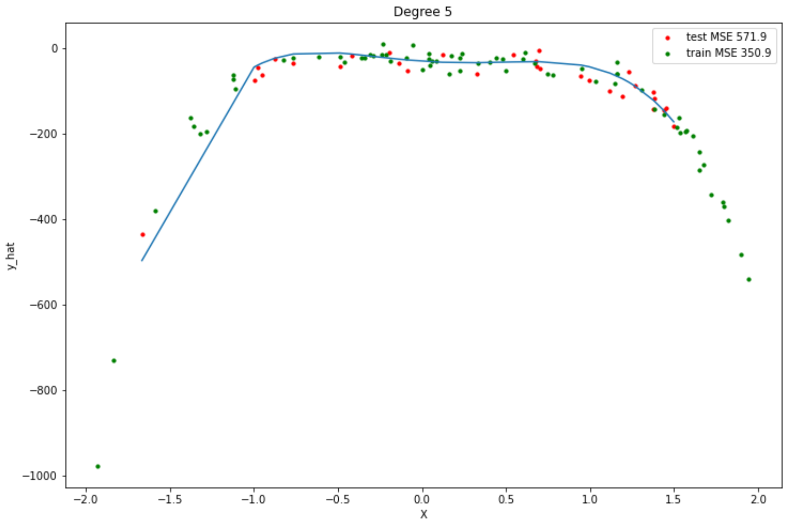


Fig 14: Plot of X and Y

❖ **RMSE plot for train and test data vs degree of polynomial**

	Degree	RMSE_train	RMSE_test
0	1	31182.02	9217.10
1	2	6109.88	3270.17
2	3	5317.40	3471.18
3	4	365.05	614.48
4	5	350.91	571.88
5	6	304.78	405.52
6	7	294.48	394.68
7	8	293.66	387.32

Fig 15: Train and test RMSE vs the order of polynomial

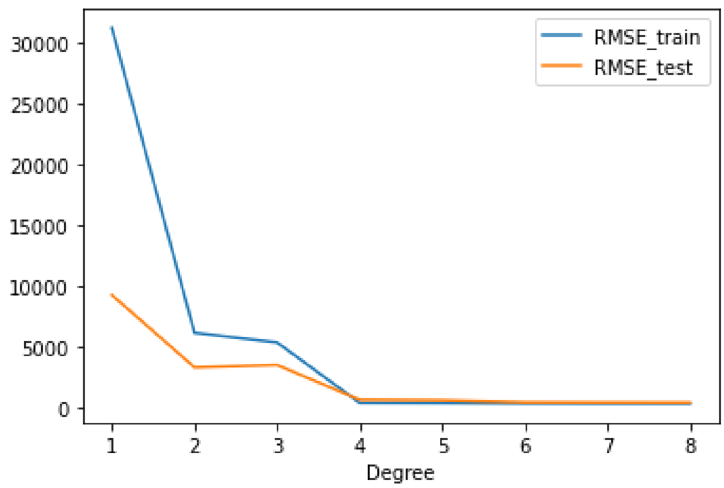


Fig 16: RMSE of train and test dataset vs order of polynomial

Here we can see that, after 6th order polynomial the test error does not decreases as the order of polynomial increases.

❖ **Analytic Method – Moore-Penrose pseudoinverse**
For 5th Order polynomial (70 Data points)

```
w = np.dot(np.linalg.pinv(X),Y_train)
```

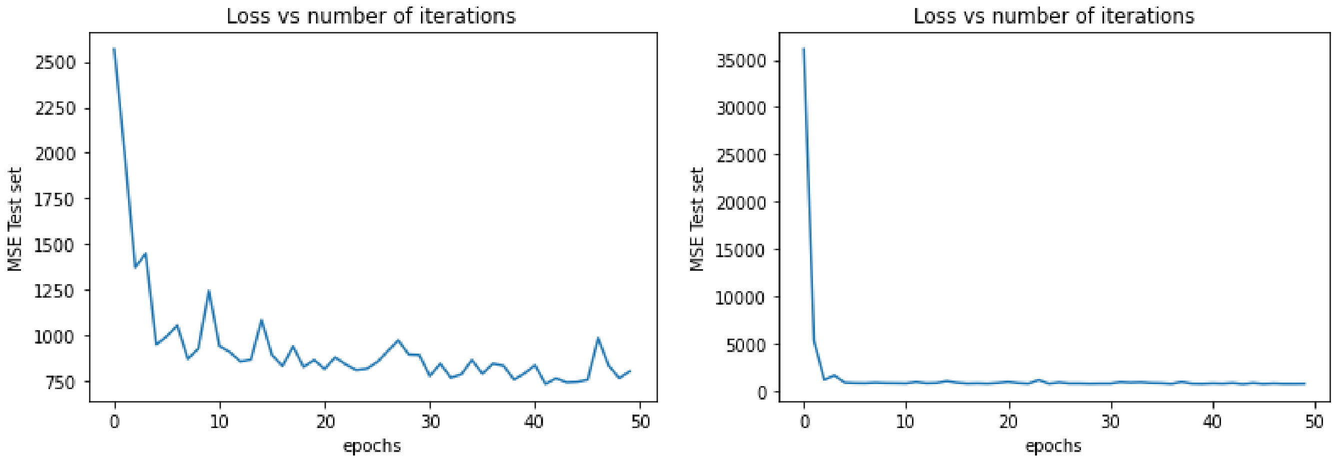
```
w
```

```
matrix([[-30.31555344],
        [-27.93812721],
        [ 50.01264287],
        [ 24.76543577],
        [-63.81418432],
        [  3.49299527]])
```

```
coefff = [[-27.93812721  ,50.01264287  ,24.76543577, -63.81418432 ,  3.49299527]
inter = [-30.31555344] #with 70 Data points
```

Fig 17: Pseudoinverse method for finding the weight matrix

❖ **Mini batch Gradient Descent Approach (70 Data points)**
As the batch size increases. The plot of test data loss become more smoother



a) b)

Fig 18: Loss vs number of iterations a) batch size =2, b) batch size =6(learning rate = 0.00045)

❖ **Comparison of Gradient Descent and Analytic Approach for 5th Order Polynomial**

S.No.	Weights	Analytic Approch	Mini Batch Gradient Descent
1	W0	-30.31555344	-12.516954
2	W1	-27.93812721	-5.76291
3	W2	50.01264287	11.31735
4	W3	24.76543577	9.251
5	W4	-63.81418432	-53.510
6	W5	3.49299527	7.544
7	Test MSE	517	761.857030

Table 2: Solution by Analytic and Mini batch gradient descent approach

❖ **Regularisation – Ridge Regression with 20 data points**

In the least square objective function, l2 norm of weights is added. If we minimize this objective function to fit the curve, its known as ridge regression.

Along with regularisation, comparison with linear, lasso and elastic net model is done.

	y_test	y_hat_linear	y_hat_ridge	y_hat_lasso	y_hat_net
0	-52.110296	-27.432663	-26.042923	-27.292987	-17.733095
1	-35.382308	-14.149082	-19.040387	-27.425861	-38.063800
2	-433.925695	-498.610459	-497.681583	-491.902564	-488.335957
3	-63.249048	-35.889764	-41.317389	-47.920396	-65.881305
4	-116.558703	-123.696378	-127.488966	-130.997566	-146.875297

Fig 19: Comparison of predicted value, where y_test is the actual label

	Order of polynomial	model_lin	model_ridge	model_lasso	model_net
0	1	-28.340096	-19.866442	-0.000000	-0.000000
1	2	48.702702	41.572584	39.399324	-1.256295
2	3	26.016043	18.801158	0.083198	3.955978
3	4	-63.480220	-61.268354	-60.414810	-46.850297
4	5	3.173484	4.553677	8.342193	6.792650

Fig 20: comparison of weights with various regularised models

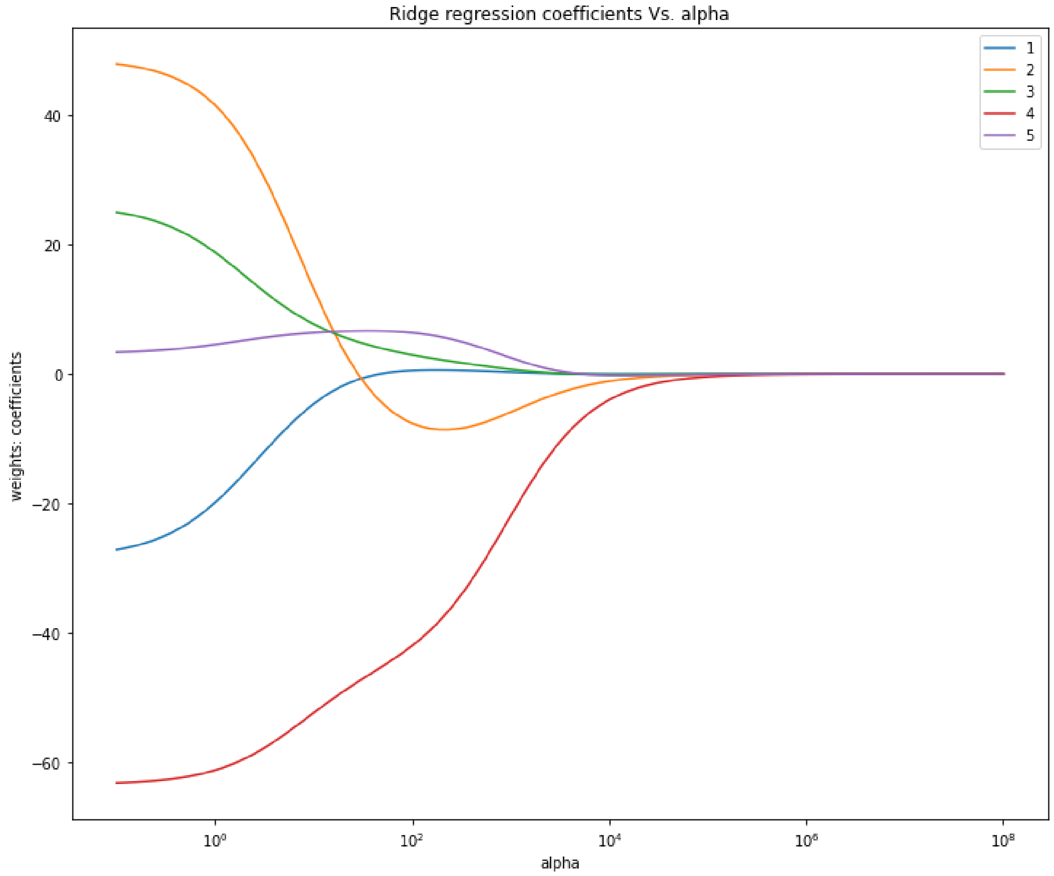


Fig 21: Value of weights vs change in alpha, where alpha refers to lambda

- **Estimation of Polynomial**

S.No.	Weights/ intercept	Values
1	Intercept	-27.965683476721097
2	W0	-19.866442
3	W1	41.572584
4	W2	18.801158
5	W3	-61.268354
6	W4	4.553677

Table 3: Weights and intercept value for the estimated polynomial

Part 1B

- **Plot of given Data**

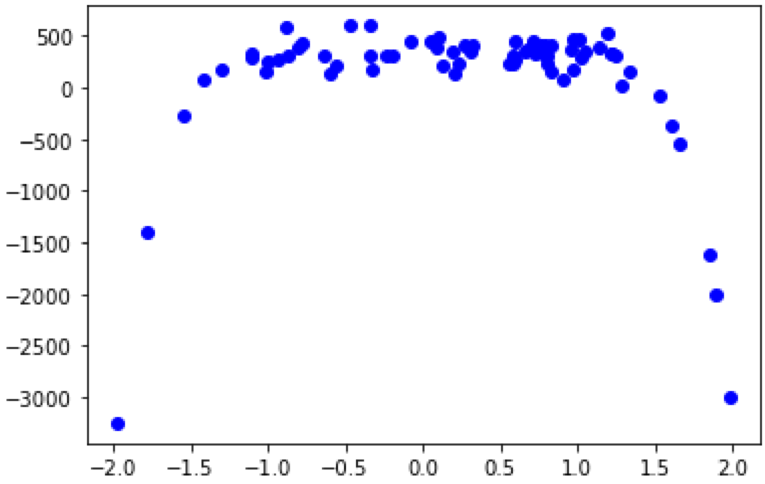


Fig 22 : Plot of X vs Y

- **Error for different order of polynomial**

	Degree	RMSE_train	RMSE_test
0	1	501321.16	971145.87
1	2	161741.39	171861.91
2	3	161490.84	169016.70
3	4	23962.84	42711.28
4	5	23788.62	43064.58
5	6	12106.79	12701.20
6	7	12094.37	12354.53
7	8	12088.68	11996.36
8	9	11908.37	13846.89
9	10	11817.28	13025.72

Fig 23 : Train Test error for different order polynomial

It can be seen from above plot that 8th order polynomial have least test error

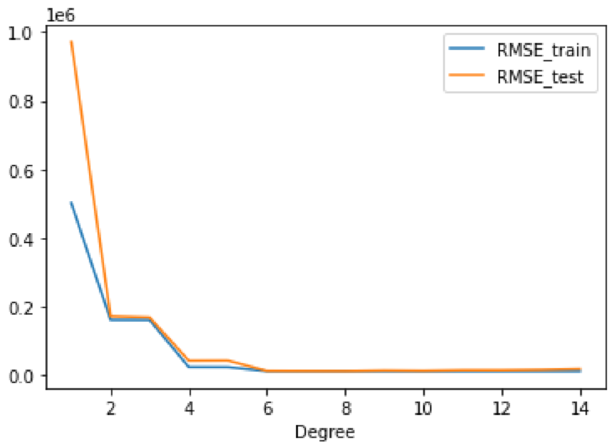


Fig 24 :Plot of Train Test error for different order polynomial

- **Noise distribution for 8th order polynomial**
  Here noise is defined as the difference between actual y test values and predicted y values

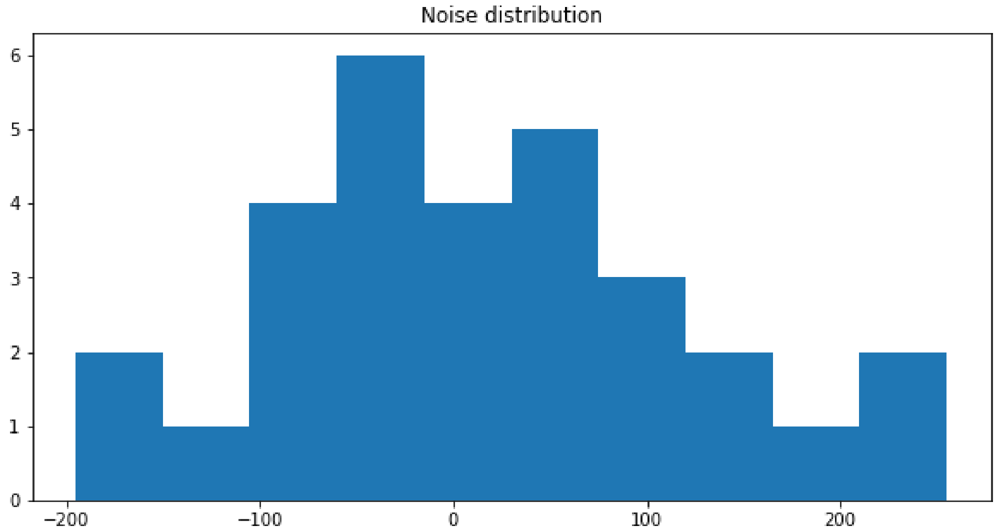


Fig 25 :Plot of Noise, this is similar to chi square noise distribution

Part 2

Data Analysis and Methodology

Dataset of 110 rows is provided. The given set of features are 'id' and 'value'.

a) Feature engineering
Extract the value of date and year from 'id' feature. Generated a design matrix for the date and year feature separately. For date, the design matrix contains power of date from 1 to 9 and for year, the design matrix contains power of year from 1 to 5.
b) Trying different subset of design matrix created for the date and year feature and it was observed that model containing date feature with order of polynomial from 1 to 9 with the year feature performed better. L2 regularization was using along with linear regression.
c) In built python library for ridge regression was used for this.

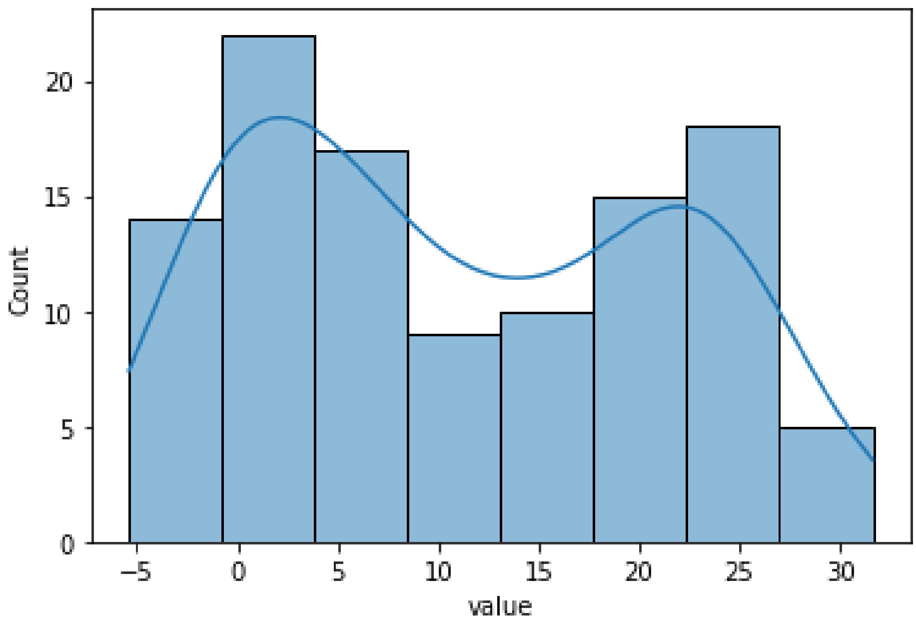


Fig 26: Distribution of label (histplot by using seaborn library)

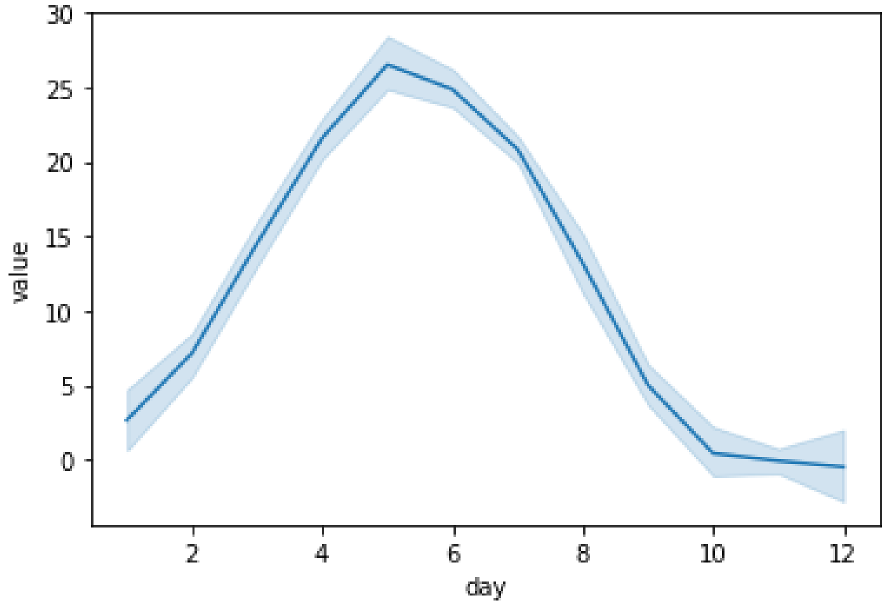


Fig 27: Lineplot of day vs value

❖ **RMSE for Ridge Regression without tuning hyper parameter Lambda**

```
MSE_test = np.mean(np.square(df_predictions['y_test'] - df_predictions['y_hat_ri
RMSE_test = np.sqrt(MSE_test)
np.round(RMSE_test,5)
```

```
3.31561
```

Fig 28: RMSE for ridge regression

❖ **RMSE for Ridge Regression with tuning hyper parameter Lambda**

```
MSE_test = np.mean(np.square(df_predictions['y_test'] - df_predictions['y_hat_ri
RMSE_test = np.sqrt(MSE_test)
np.round(RMSE_test,5)
```

```
1.95195
```

Fig 29: RMSE for ridge regression (lambda =10)

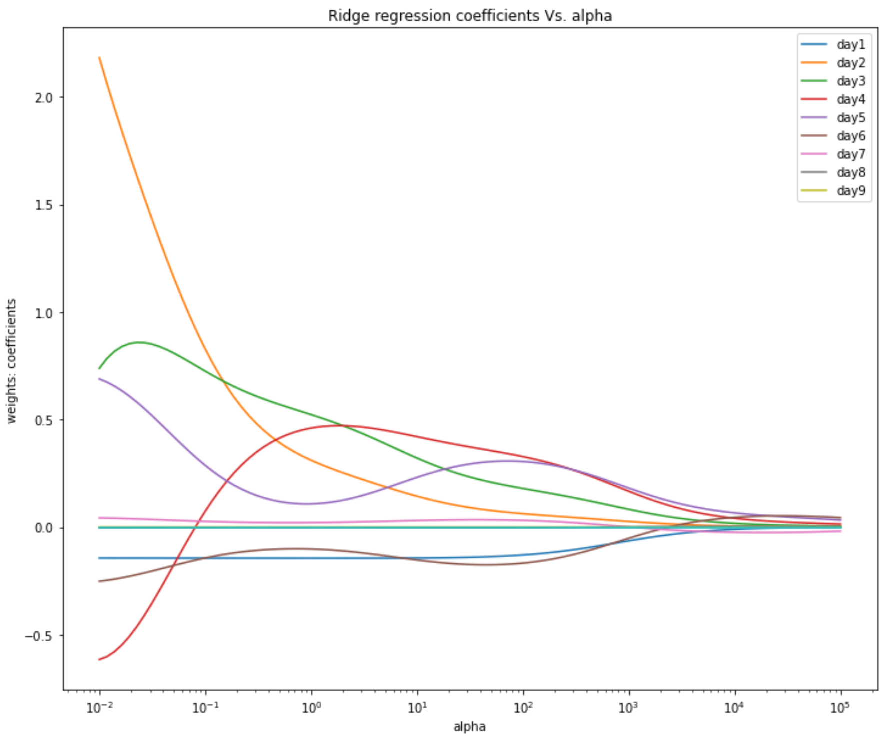


Fig 30: Value of weights vs change in alpha, where alpha refers to lambda

❖ Predicted values

id	value
5/1/10	26.328472
4/1/09	22.208721
9/1/13	4.618891
1/1/06	2.904471
2/1/07	7.412829
8/1/12	12.278132
6/1/14	25.059085
3/1/08	14.827978
12/1/04	0.251985
7/1/11	20.232444

Fig 31: Predicted values