1. **Part-1**

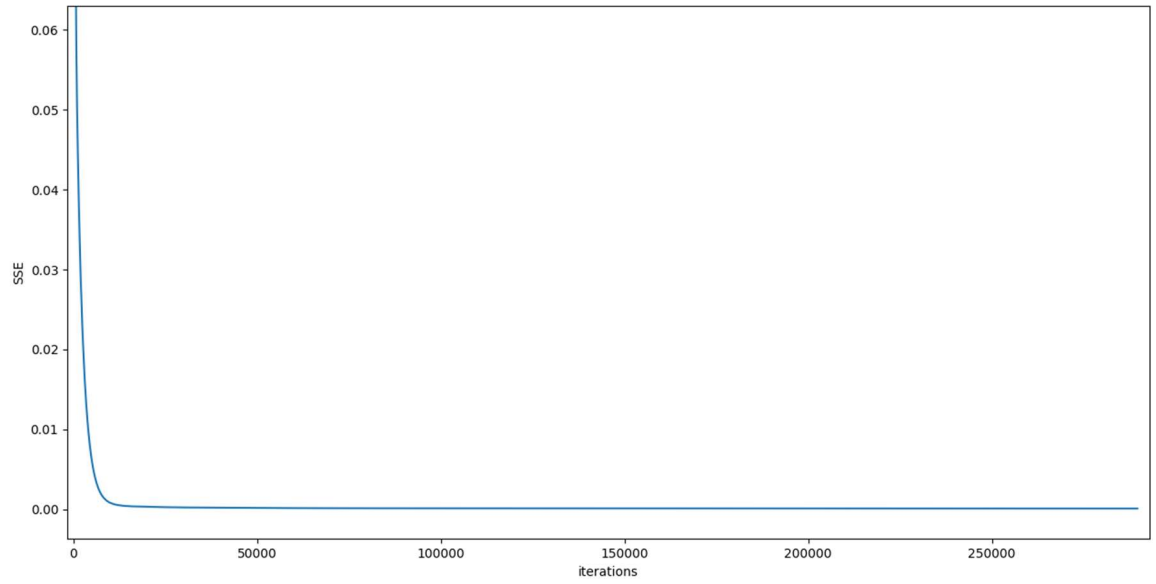
Name: Guruvu Surya Sai Prakash
Entry No: 2019EE10481

1-a:

Moore's Penrose Inverse:

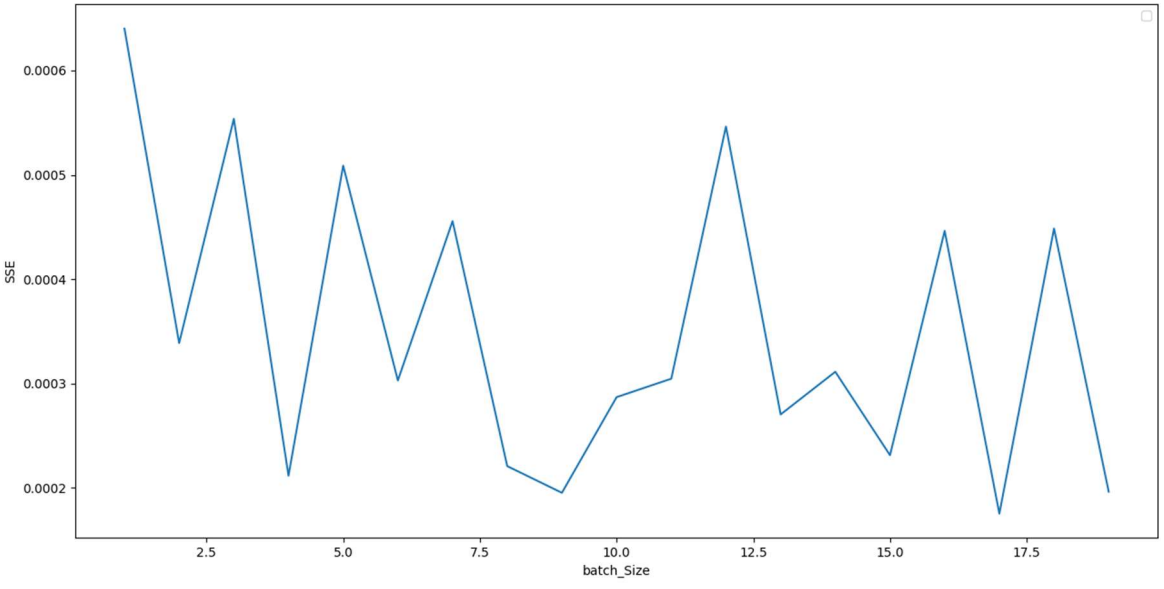
Batch gradient descent:

Variation of SSE with no of iterations: (Below plot is for m=6)



From, the graph it is shows SSE decreases as no of iterations increases. This implies that our learning rate is good enough to capture the minimum value.

Variation of SSE with Batch-size: (Below plot is for m=6)



From the graph it shows that SSE is almost the same for different batch sizes. So, this implies the idea of not taking all the error gradients for arriving at the output is a good idea.

Finding the degree from 20 points and doing cross-validation:

Error functions used:

1. Mean Square Error: It is given by

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)^2}_{\text{The square of the difference between actual and predicted}}$$

The smaller the value of MSE, the better is the fit.

2. R-squared Error: It is given by

$$R^2 = \frac{SSR}{SST} = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2}$$

The greater the value of R2(max=1), the better is the fit.

The difference between MSE and R2 is that MSE gets pronounced based on whether the data is scaled or not. This is where R-Squared comes to the rescue. R-Squared is also termed as the standardized version of MSE. R-squared represents the fraction of variance of response variable captured by the regression model rather than the MSE which captures the residual error.

Now, Taking first 20 points as training set. Remaining points as validation(test) set. I started my analysis.

Variation of MSE and R2 with degree(m) with no regularization:

1. **Variation of MSE with m:**

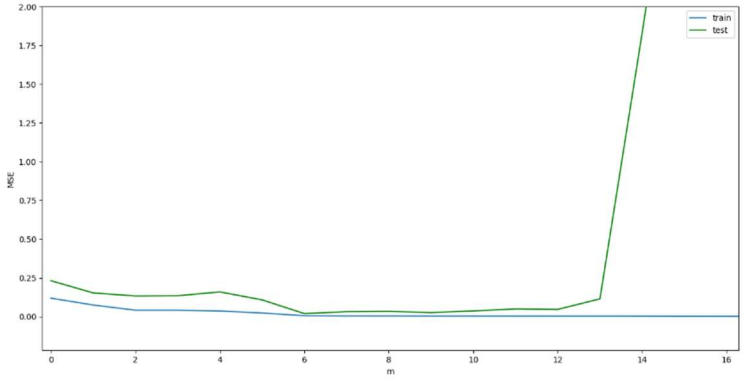
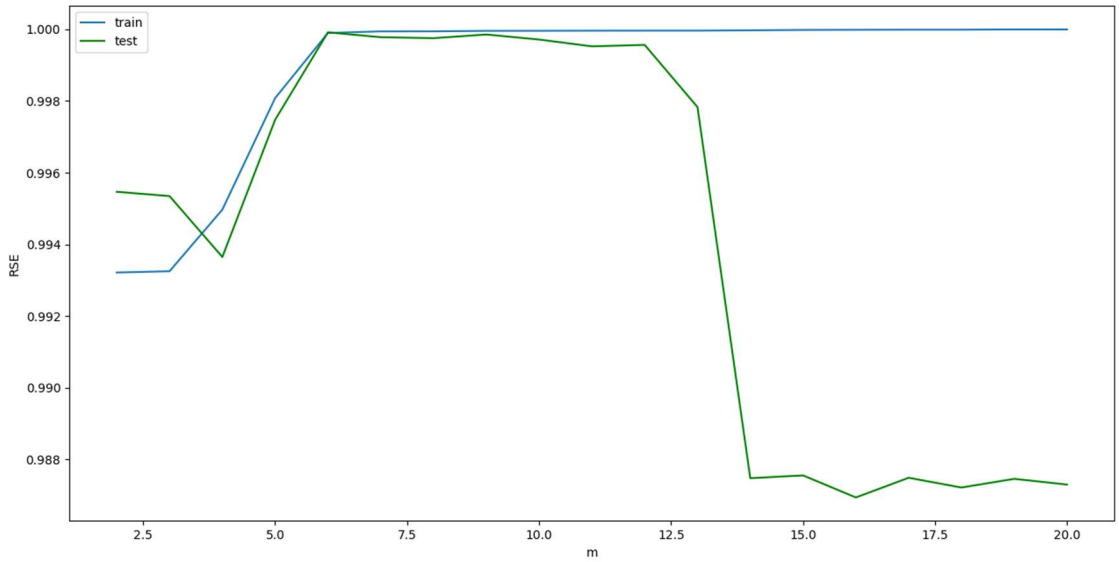


Fig1: Variation of MSE with m for training Variation of MSE with m

2. **Variation of R2 with m:**

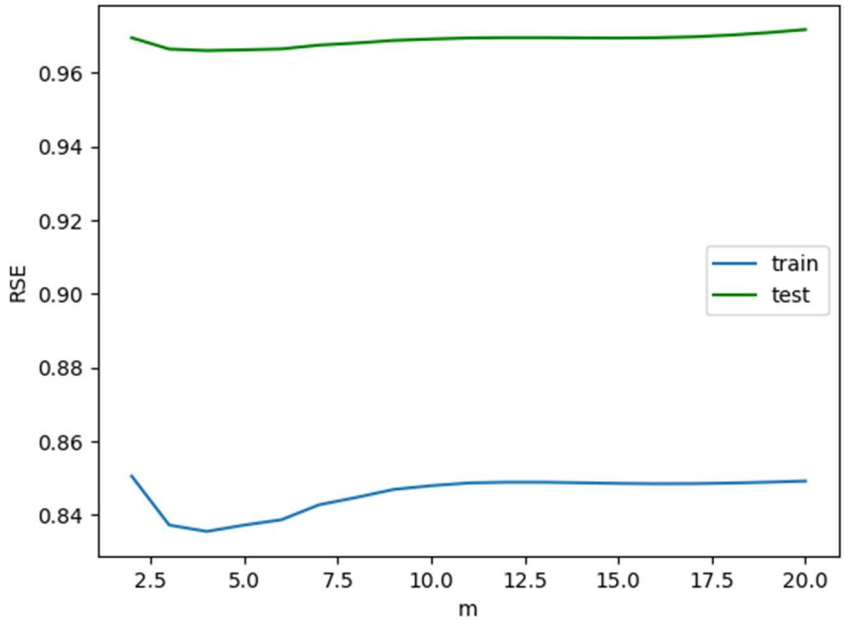


From, the plot we could see that for m >=14, the models start overfitting.

From the above two plots we get the best fit for m=6, with R2_score=0.999919 and noise variance= 0.00039

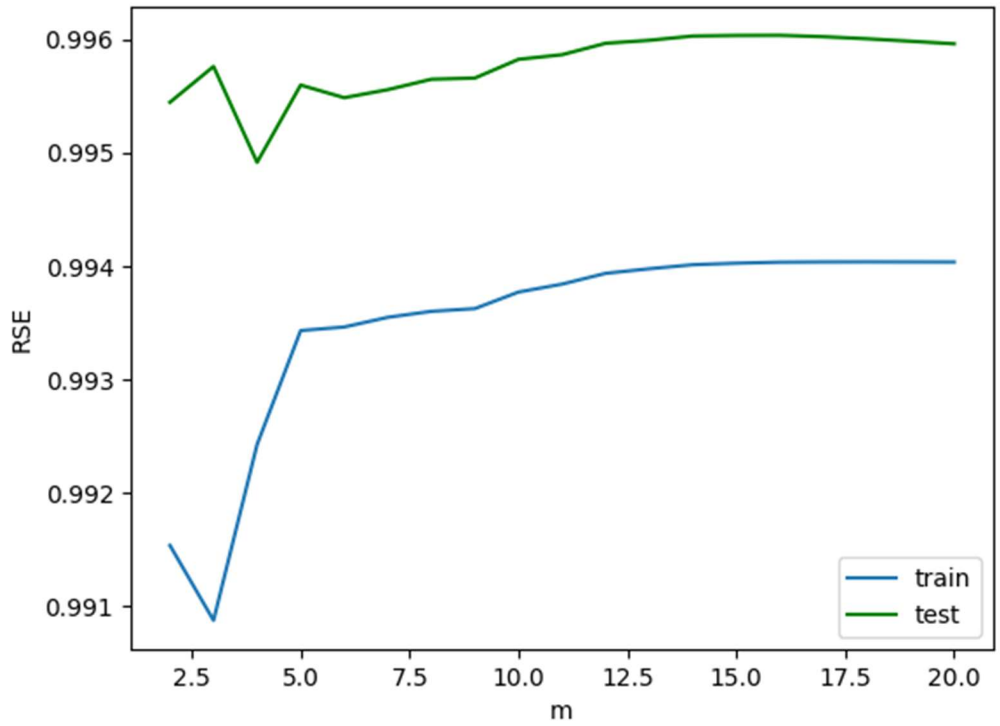
Variation of R2 with degree(m) with regularization:

1. **Lambda=1:**



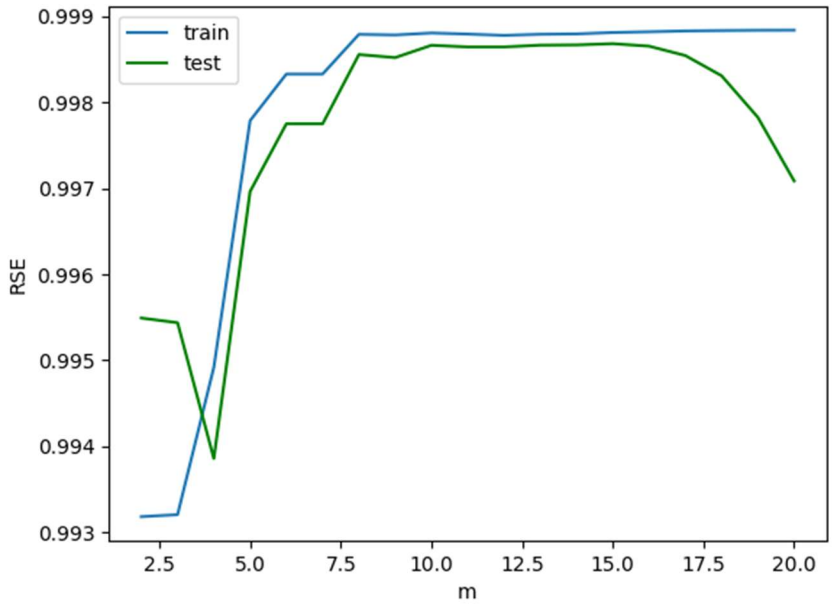
Here, the best R2_score for training set is 0.85 (regularization dominates) and for testing set is 0.97 (less than that for m=6 with no regularization).

2. **Lambda=0.01:**



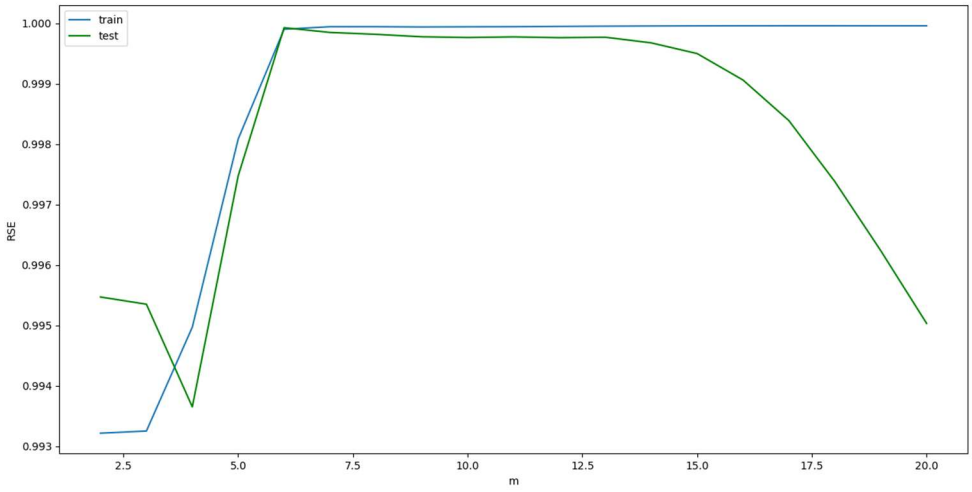
Here, the best R2_score for training set is 0.994 (for m=18) and for testing set is 0.996 (for m=16). This looks fine in terms of R2_score, But the complexity has increased and we got an even better R2_score for m=6 without regularization.

3. **Lambda=0.001:**



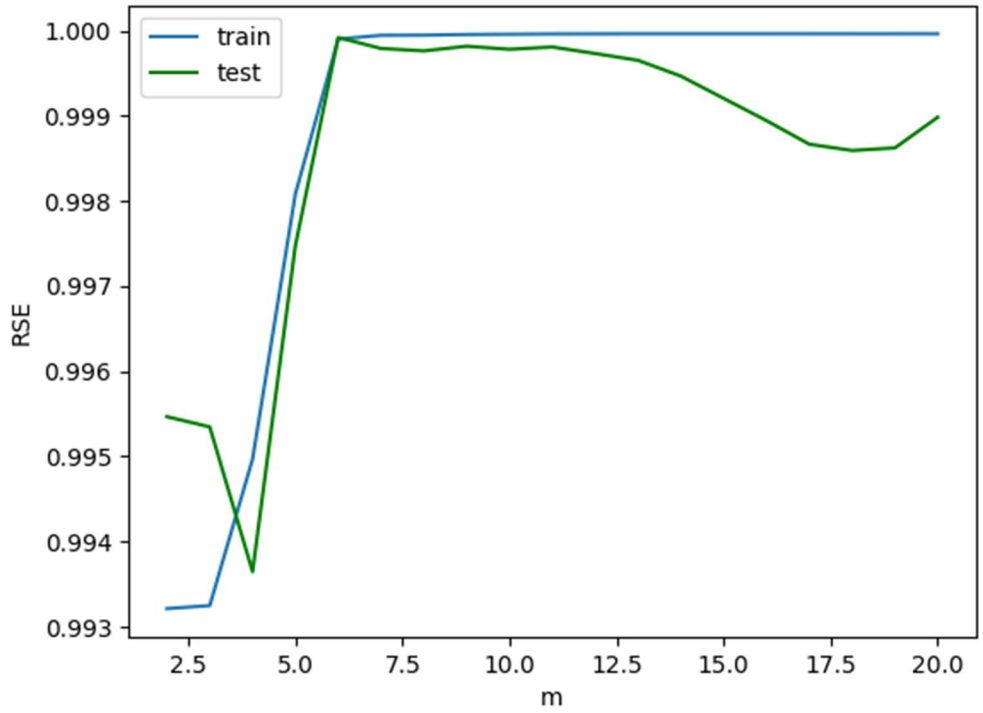
Here, the best R2_score for training set is 0.998 (for m=15) and for testing set is 0.998 (for m=20).Here, also just as before the model complexity is more for best R2_score and also this score is less than that for m=6 without regularization.

4. **Lambda=0.00001**



Here, we get best R2_score for testing set is 0.99992 (for m=6) and for testing set is 0.999921 (for m=6). So, this is better fit than without regularization and model complexity is also same.

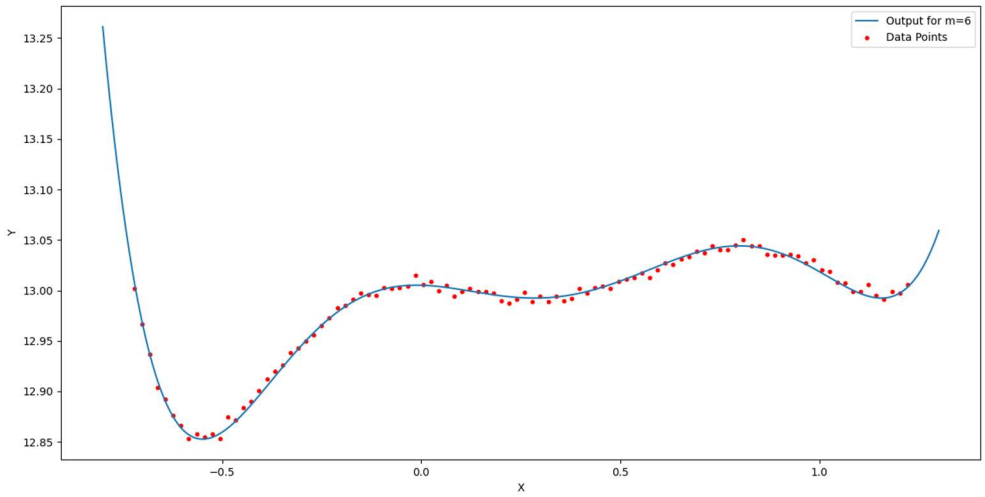
5. **Lambda=0.0000001:**



Here, the R2_score is same as without regularization. So, the effect of regularization is almost vanished.

So, for best fit **m=6** and **lambda=0.00001.**
The Noise variance is = 0.0003807.

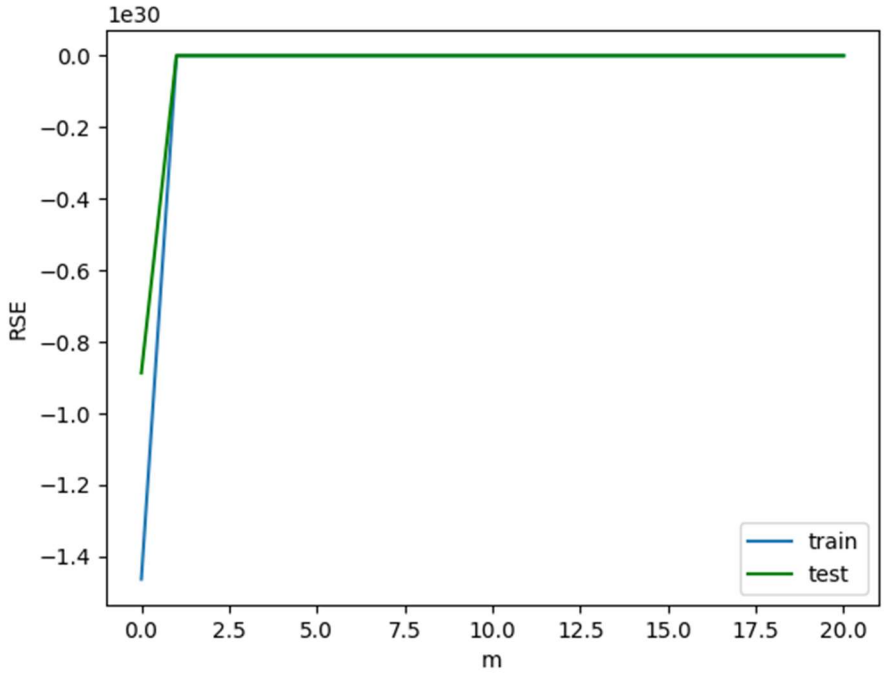
Plot for m=6 and lambda=0.00001:



The value of w vector is= **[13.00498 -0.00877 -0.43725 1.154789 0.368528 -2.10005 1.037099]**

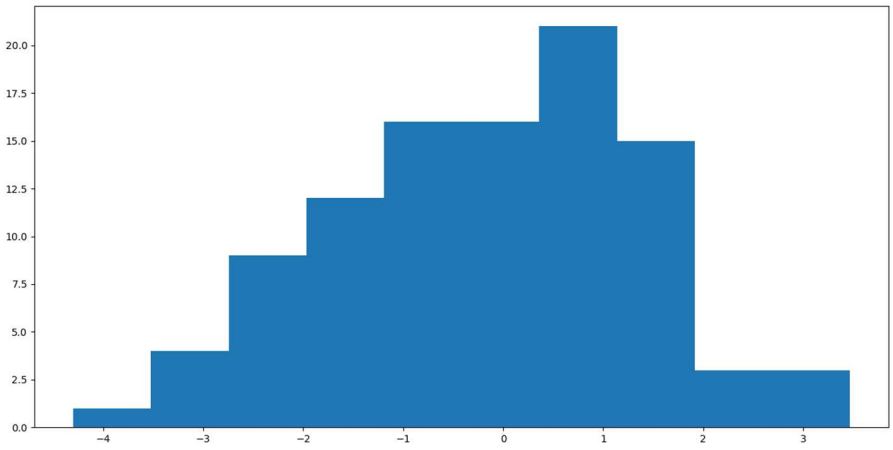
3. Part1B:

Given, a data with non-gaussian noise. I tried to fit to a polynomial and found the degree of best fit as 8.



After that from the found w vector and there by the polynomial. I plotted the histogram of noise.

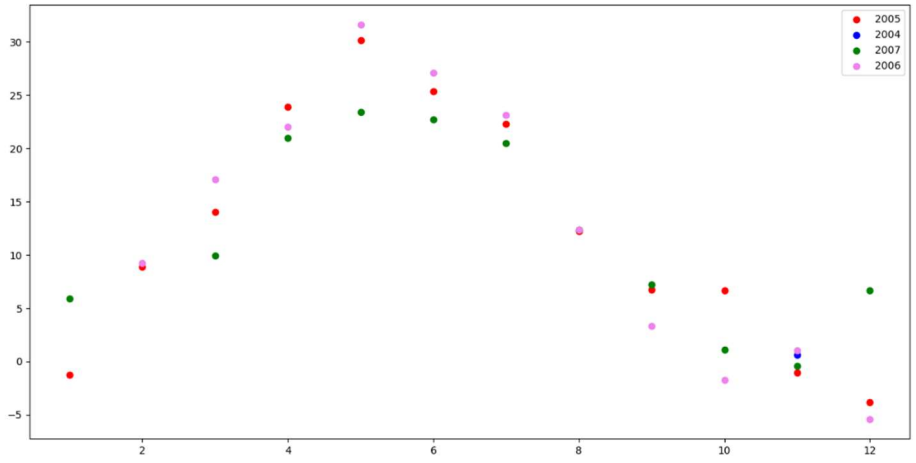
Which is as shown in in the figure.



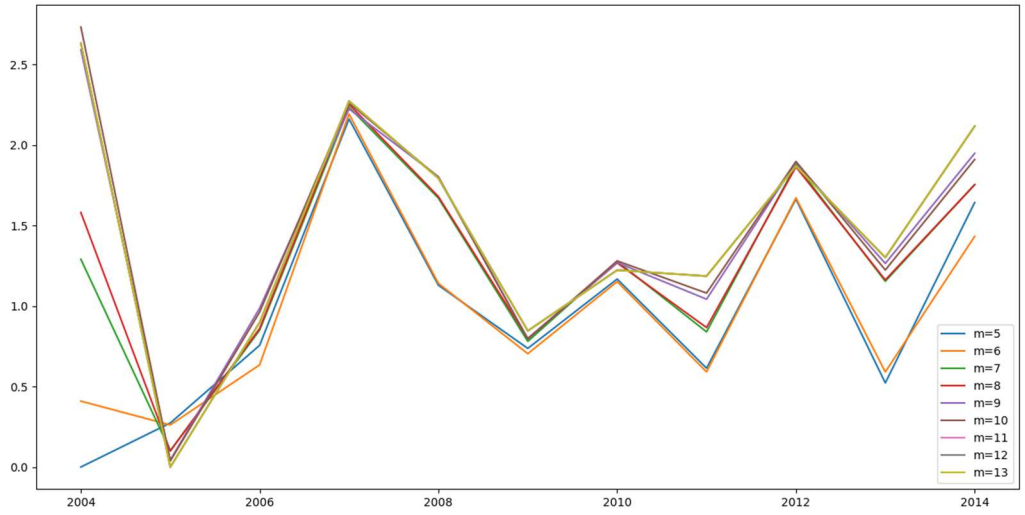
The noise distribution appears like some quadratic distribution.

3. Part-2:

We have a time series data set. First for finding the feature vector, I plotted the graphs for the two possible feature vectors year and month.
I found that for each year, the variation of data with month is following kind of a similar pattern.

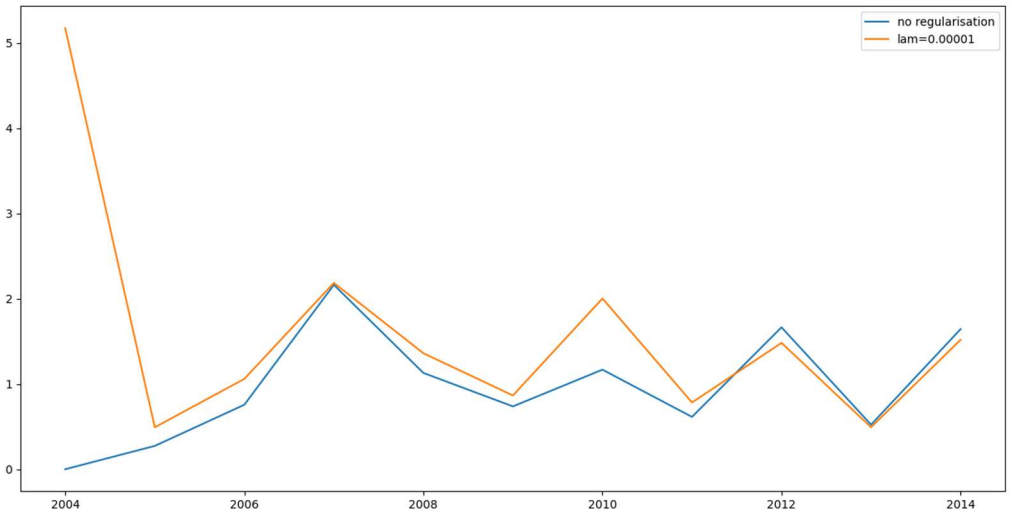


So, I choose month as feature vector. Since, there are more data points for the year 2005. I chose it as the training set. And, the remaining are testing sets.
Now, I chose value of feature vector as (month/12). So, that the values are normalized and do not cause any over flow errors.
Now, I tried to fit this using Polynomial regression.
I chose the evaluation function as MSE (Mean Squared Error).
Now, I observed the value of MSE for each testing set with variation of m. This is present in the below plot. (Y-axis is the MSE for corresponding year and polynomial degree).



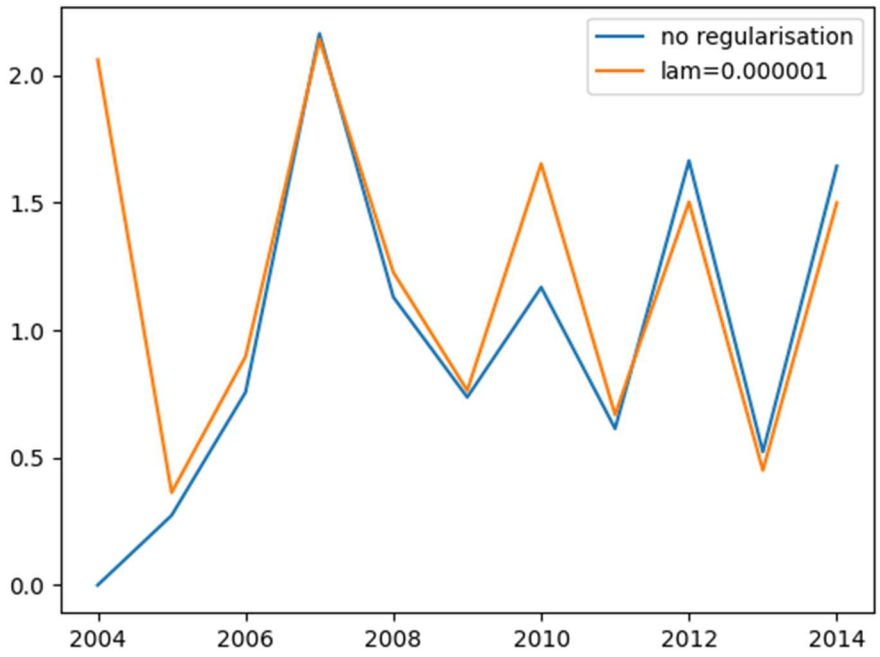
I have omitted the plots of m=1 to 4. They were underfitting and giving large errors on both training and testing data sets.
From this plot we could see that the MSE error is small for m=5 for all the testing data.
So, the best fit is for **m=5.**

Lambda-Tuning:
For lambda=0.00001:



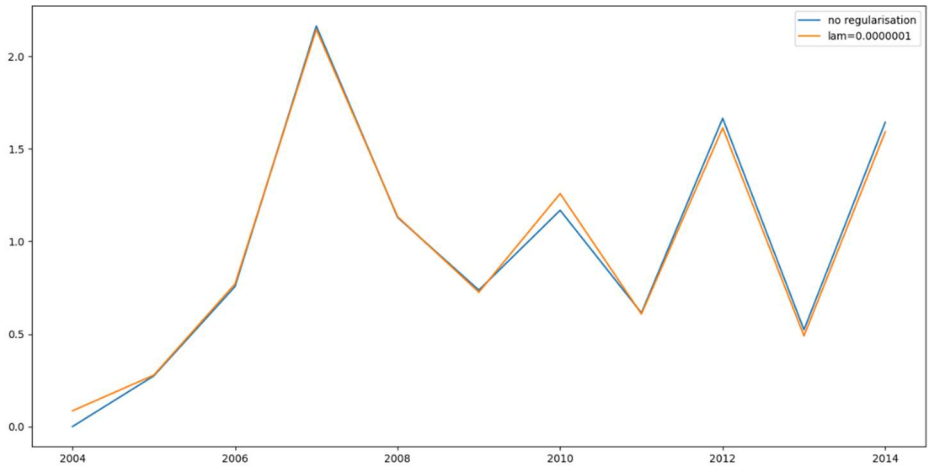
Here, lambda is dominating much.

For lambda=0.000001:



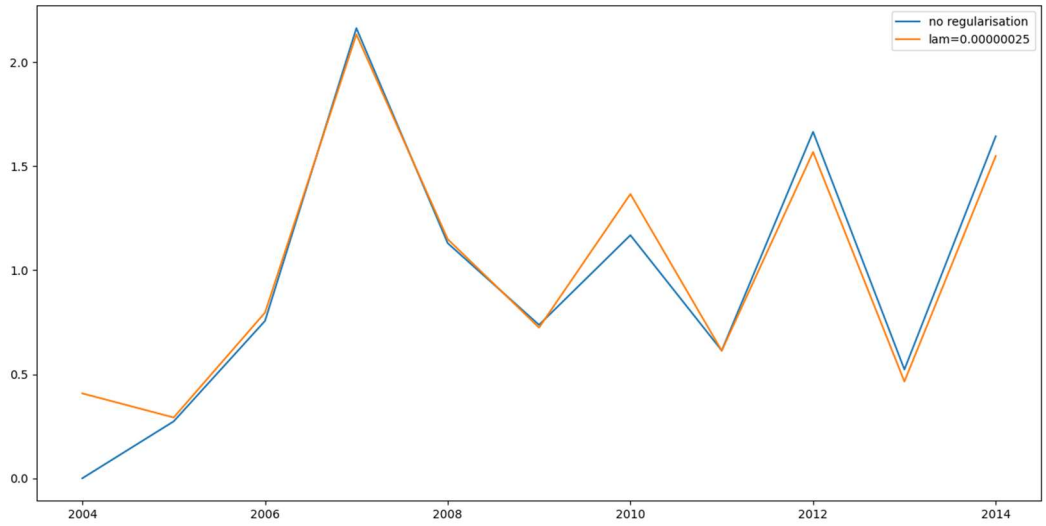
Here, it is better than before but lambda is yet dominating.

For lambda=0.0000001:



Here, the effect of lambda has almost vanished.

For lambda=0.00000025:



Here, all the errors are good except those errors for 2004,2010. Which are varying too much.

So, I feel we could get a lambda which could have a better MSE than lambda=0.
So, m=5 with lambda=0 are our hyper parameters.

The **w** vector **for m=5 and lambda=0** is
[5.40998636 -202.43357273 1938.42605547 -4895.4425146 4826.73089099 -1676.83564018]
The larger values of w are because of I divided my X by 12 initially.