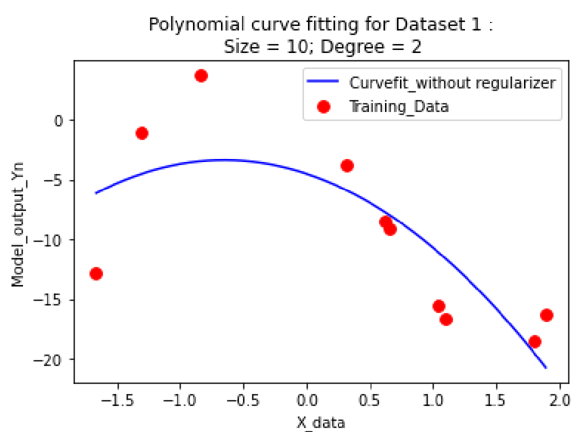
TASK 1:Polynomial curve fitting

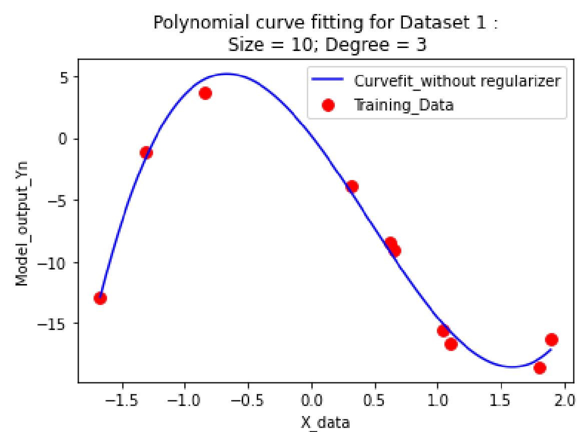
Dataset 1*: 1-dimensional (Univariate) input data*: **function_2.csv**

Approximated functions obtained using training datasets of different sizes (10 and 200) for different model complexities (degrees 2, 3, 6 and 9) and different values of λ.

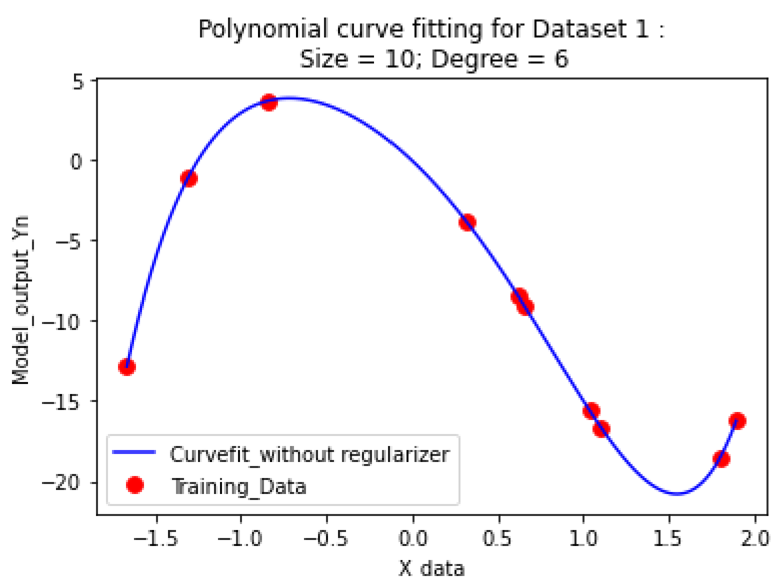
Without regularization: 10 training points



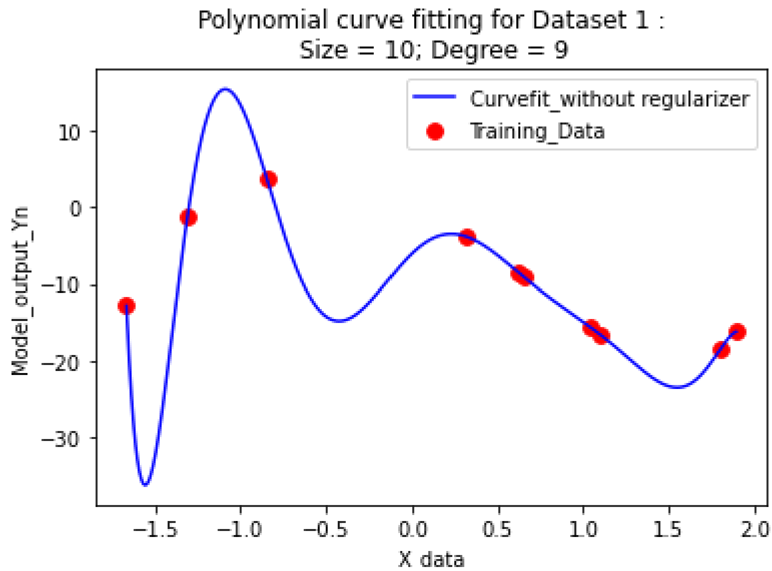
Plot 1.1



Plot 1.2



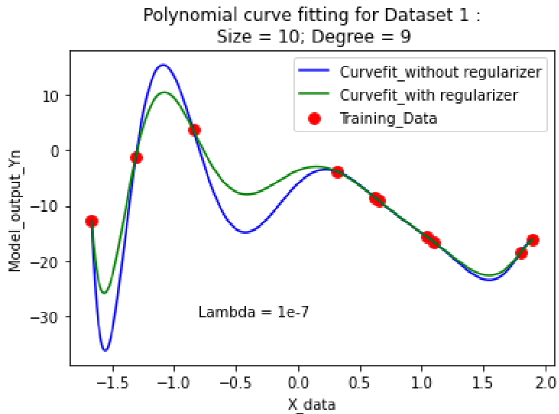
Plot 1.3

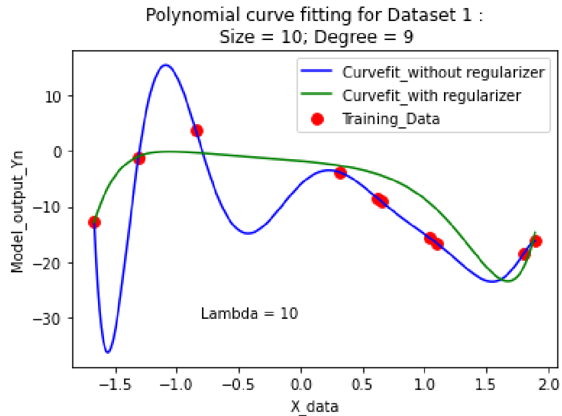


Plot 1.4

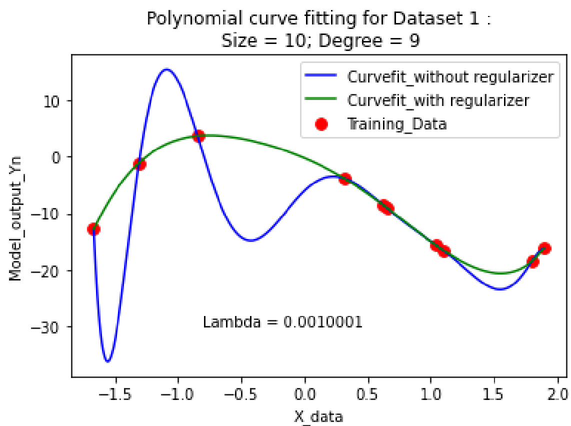
Inferences: (Size-10, no regularization)

- Here, we can see that a low complexity(degree) of 2 is not sufficient to fit the training points well. But Degree 3 performs significantly better but still doesn't pass through a few points.
- And degree 6 fits the curve amazingly well and passes through all the training points!
- A high complexity of 9, with just 10 training points, implies that our curve does pass through all the training points as evidenced by the graph, but there might be overfitting here, as we have not provided sufficient training points to train the 9th-degree model.





Plot 1.16 **Plot 1.17**



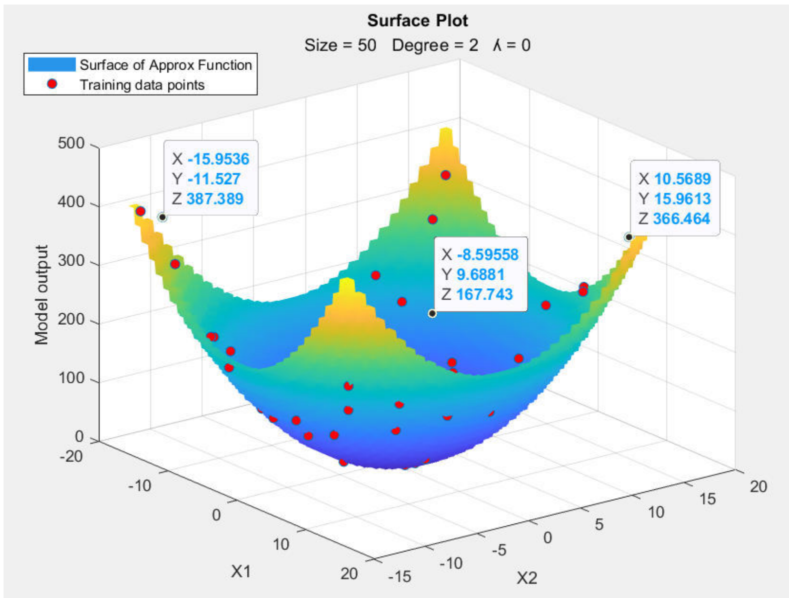
Plot 1.18

This case makes us appreciate the importance of regularization perfectly! When we have a few training data points and try to fit a complex model using them, we usually tend to overfit the training data! Regularization penalizes the fluctuations/roughness of the fitted curve. In this case, we can see that the regularized curve is much smoother than each of the non-regularized curves!

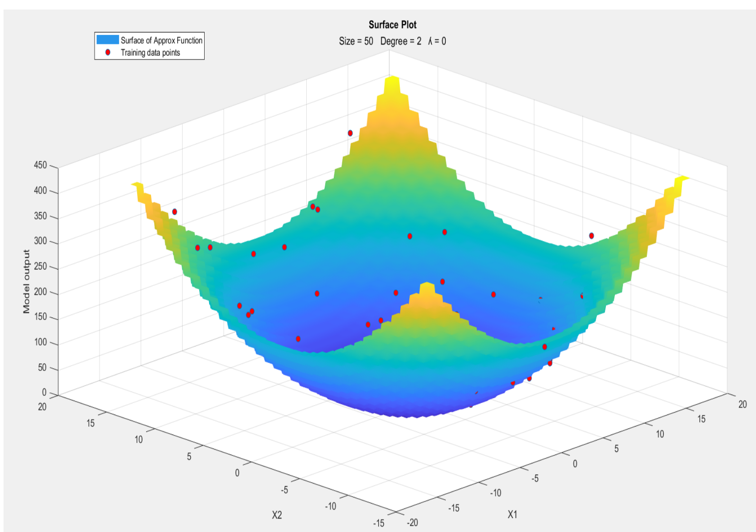
And the value of the regularization parameter determines how good we fit the curve. When we get the Lambda values just right as in **Plot 1.18.** We get a very neat curve, that perfectly fits all the training points, while also remaining smooth, generalizing very well! While, a very low lambda value, means that the curve is still overfitting the training data, as there is not enough importance given to the regularization. And in the case of extremely high lambda values, we underfit the training points and it doesn't even pass through all the training points!

TASK 2: Linear regression using polynomial basis functions

Note: Here, for 3D plots, we had to export the data onto an excel sheet from python variable explorer, then imported it and plotted it using Matlab as the Python surface plots were not good enough.



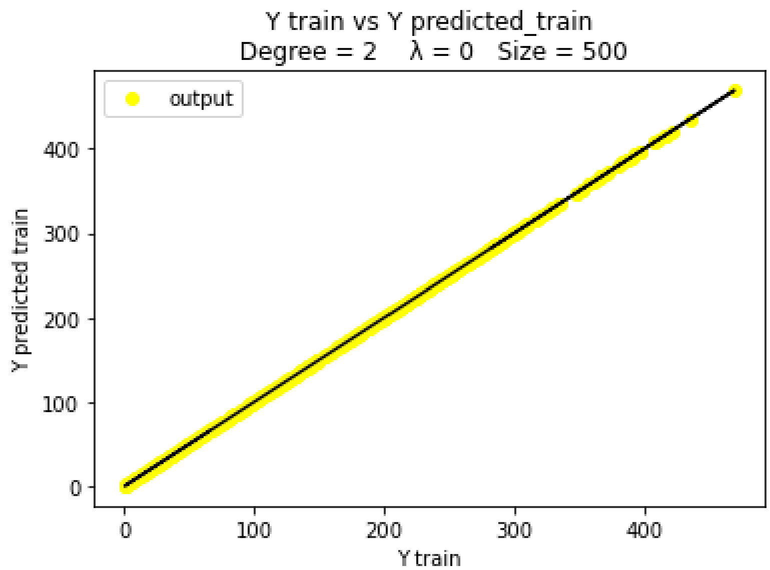
Plot 2.1a



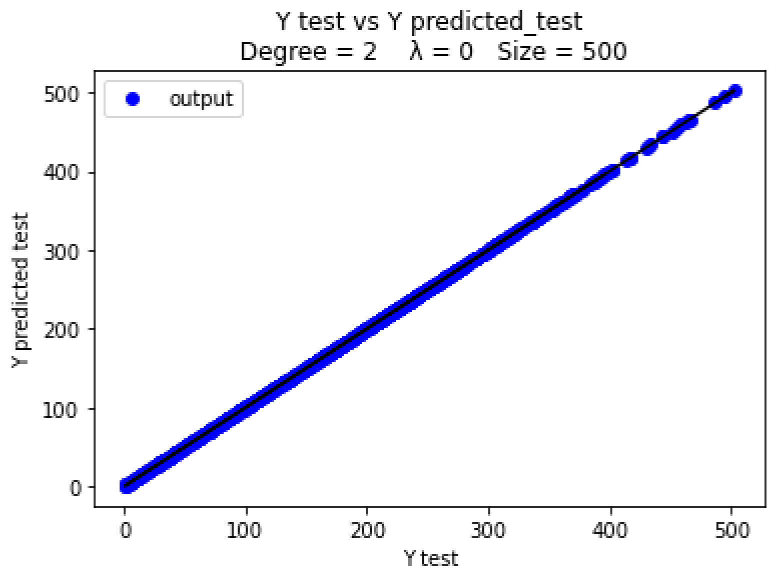
Plot2.1b

Non-regularized model of **degree 2** and **size 50** itself seems to fit the training data points fairly well!

Best Overall Model {Size-500, Degree-2, λ=0}



Plot 2.37



Plot 2.38

The model performs extremely well for both training and testing data points!

model does not generalize well to all unseen points. To overcome this, we had to shuffle the entire data frame and then run the whole process! Albeit this, the first row of the above table was still kept in the table, as it was the best performing model amongst all the unshuffled models!

We now plot the best performing models on both unshuffled and shuffled data sets!

Best Unshuffled Model:

Plot 3.3 a,b