Assignment-2

Abhinay Maurya

2021101132

Task-1 :Linear Regression

The LinearRegression() in sklearn module creates a linear prediction model by selecting a set of coefficients

w=(w1,…,wn)

and a bias b to minimise the residual sum of squares between the actual targets in the training dataset, and the targets predicted by linear approximation of the features.

LinearRegression() takes a few parameters which help to decide certain features of the linear model. The LinearRegression().fit() is the function which finds the optimal values of the intercepts where the arguments are the existing input(training dataset). It creates the model to fit the training dataset. The coefficients are initialised by the fit() method to the most optimum values by minimising the Mean Squared Error between the training data and the predicted data. The .fit() function fits any instance of linear regression.

Task-2 : Gradient Descent

Gradient descent is an iterative optimization algorithm that aims to find the optimal values of the coefficients (or parameters) of a model that minimizes the cost or loss function.

In the case of a model with one independent variable and one dependent variable, we can represent the model as: $y = \beta_0 + \beta_1 x$

where y is the dependent variable, x is the independent variable, and β_0 and β_1 are the coefficients to be estimated.

The goal of gradient descent is to minimize the cost function, which is a function of the difference between the predicted values of the model and the actual values of

the dependent variable. One common cost function is the mean squared error (MSE), which is defined as: $MSE = (1/n) \sum(y_i - \hat{y}_i)^2$

where n is the number of observations, $\mathbf{y_i}$ is the actual value of the dependent variable for the i-th observation, and $\hat{\mathbf{y}}_\mathbf{i}$ is the predicted value of the dependent variable for the i-th observation.

To find the optimal values of the coefficients, gradient descent iteratively updates the values of the coefficients by taking steps in the direction of the steepest descent of the cost function. The step size is determined by the learning rate, which is a hyperparameter that controls the size of the steps.

The algorithm starts with an initial guess of the coefficients, denoted as β_0 and β_1. Then, it updates the coefficients at each iteration as follows:

$\beta_0 = \beta_0 - \alpha(1/n) \sum(\hat{y}_i - y_i)$

$\beta_1 = \beta_1 - \alpha(1/n) \sum(\hat{y}_i - y_i)x_i$

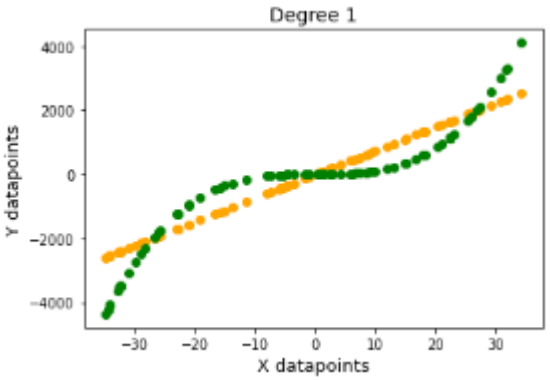
where α is the learning rate, $\hat{y}_i$ is the predicted value of the dependent variable for the i-th observation, and x_i is the value of the independent variable for the i-th observation.

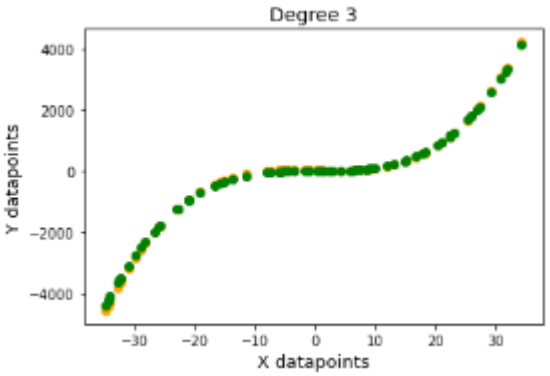
The update rule for the coefficients can be derived by taking the partial derivative of the cost function with respect to each coefficient and setting it to zero. This results in a system of equations that can be solved iteratively using the above update rule.

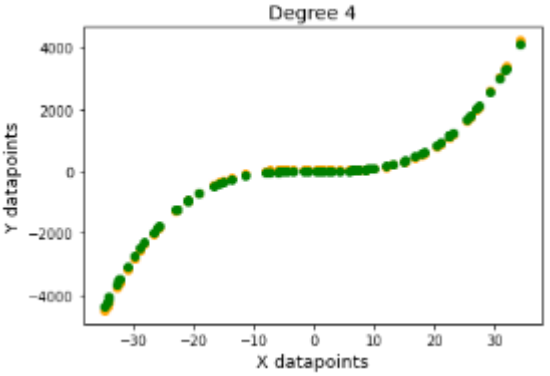
The algorithm continues to update the coefficients until the cost function converges to a minimum or until a maximum number of iterations is reached. At the end of the process, the optimal values of the coefficients are the ones that minimize the cost function and provide the best fit for the data.

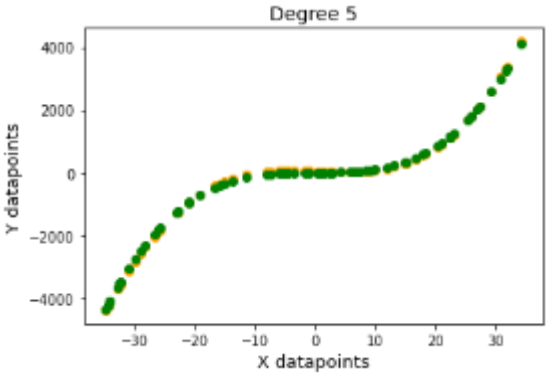
Task-3 : Calculating Bias and Variance

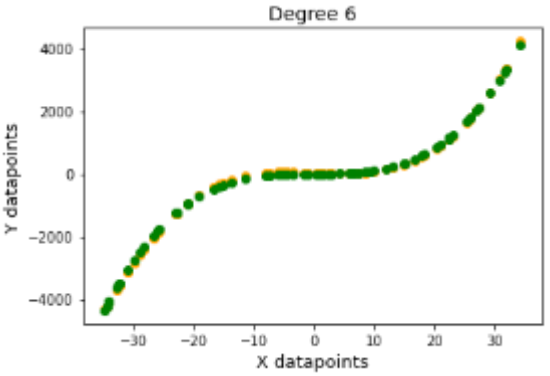
The following graphs depict the predicted values against the original testing values. As can be seen from the following graphs, the worst overlap for predicted and original values is for polynomials of degrees 1 and 2 while it is the best for polynomials with degrees 3-5. So our prediction is that the function most likely represents a polynomial of degree 3 or 4 or 5.

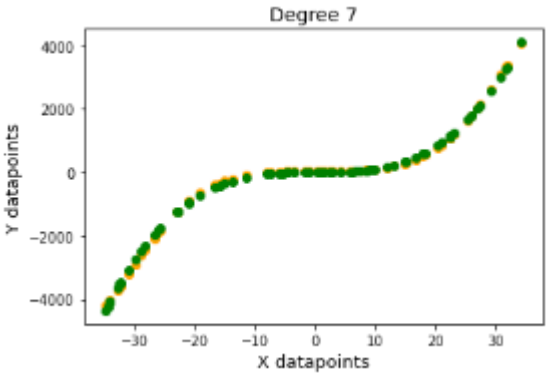

Degree 1
Y datapoints
X datapoints

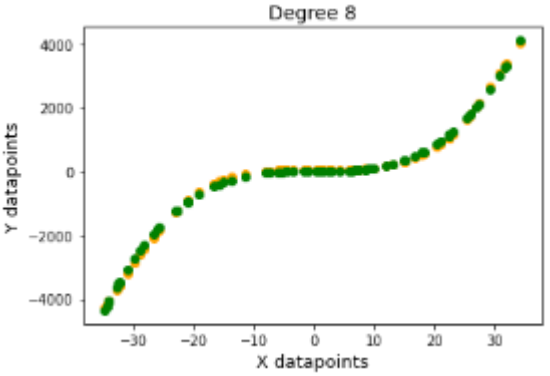

Degree 3
Y datapoints
X datapoints

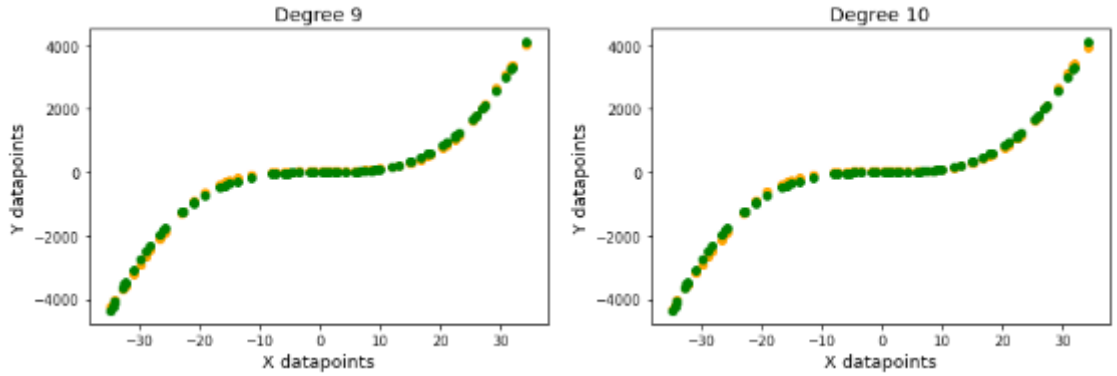

Degree 4
Y datapoints
X datapoints


Degree 5
Y datapoints
X datapoints


Degree 6
Y datapoints
X datapoints


Degree 7
Y datapoints
X datapoints


Degree 8
Y datapoints
X datapoints



If we observe the bias and variance for polynomials from degree 1 to 15 we notice that the bias is high for degrees 1 (which implies underfitting) and then it abruptly falls for degree=3 and then it remains nearly constant for a while until it starts to slowly increase for higher degrees.

As for variance, it keeps on increasing from degree 1 (where we have an underfitting model) to 15 (where we have an overfitting model). We need to find the right tradeoff between variance and bias to avoid both underfitting and overfitting. From the following table we can infer that the right tradeoff will be for degrees 2 to 10.

	Bias	Variance
1	0.26940	0.00868
2	0.08626	0.00122
3	0.03327	0.00034
4	0.02428	0.00037
5	0.02388	0.00046
6	0.02396	0.00058
7	0.02483	0.00092
8	0.02489	0.00176
9	0.03042	0.00828
10	0.02866	0.00650
11	0.03659	0.03269
12	0.07092	0.88449
13	0.04225	0.05432
14	0.15643	8.04335
15	0.08913	2.66953

Task-4 : Calculating Irreducible Error

Irreducible error is the error that cannot be reduced by creating good models. It is a measure of the amount of noise in the data.

$E[(f(x) - \hat{f}(x))^2] = \text{Bias}^2 + \sigma^2 + \text{Variance}$

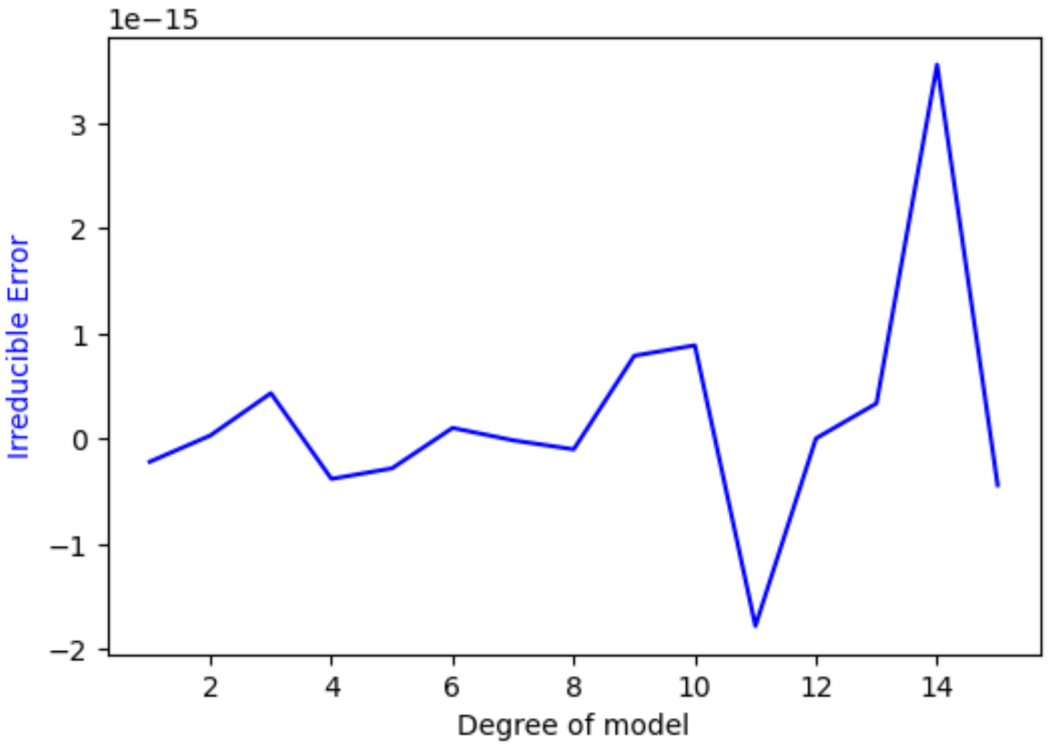
σ2 = E[(f (x) − f ˆ (x))2] − (Bias2 + Variance)

where f (x) represents the true value,

f ˆ (x) represents the predicted value,

[E[(f (x) − f ˆ (x))2] is the mean squared error (MSE) and

σ2represents the irreducible error.



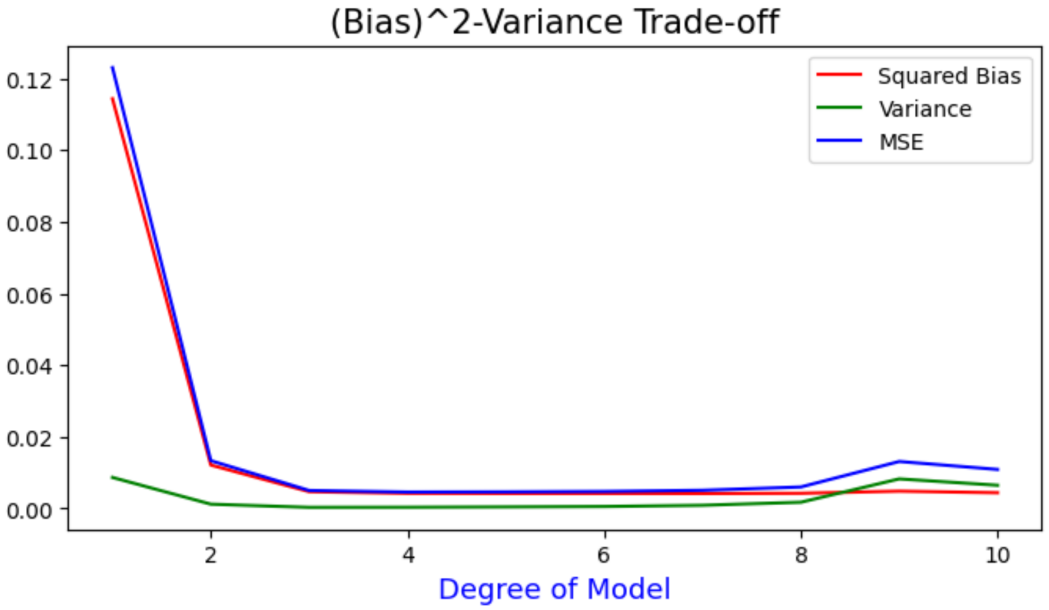
For almost all cases from degree 1 to 10,12,13, the irreducible error remains close to 0. Ideally irreducible error should be 0 but some small amount of error arises due to the presence of noise in virtually every data.

The irreducible error is a type of error that cannot be reduced by improving the model. It is caused by factors that are not accounted for by the model, such as measurement noise, unmeasured variables, or inherent randomness in the data generating process. The value of the irreducible error should not change as you vary the class function or the model complexity, since it represents a fundamental limitation in the ability of any model to accurately capture the underlying relationship between the input and output variables. The array of irreducible errors provided in the question contains 15 values, one for each degree of a polynomial model. The values range from -1.78329573e-15 to 8.84708973e-16, with some values very close to zero. These small values suggest that the irreducible error is relatively small for this dataset, and that the model is able to capture most of the underlying relationship between the input and output variables.

	Degree	Irreducible Error
1	1.0	-2.220400e-16
2	2.0	2.776000e-17
3	3.0	4.302100e-16
4	4.0	-3.851100e-16
5	5.0	-2.844900e-16
6	6.0	9.975000e-17
7	7.0	-1.821000e-17
8	8.0	-1.040800e-16
9	9.0	7.858300e-16
10	10.0	8.847100e-16
11	11.0	-1.783300e-15
12	12.0	0.000000e+00
13	13.0	3.330700e-16
14	14.0	3.552710e-15
15	15.0	-4.440900e-16

Task-5 : Plotting Bias2 − Variance graph

We observe that the bias drops suddenly from degree=2 to degree=3 then remains about the same and then for higher degrees it starts increasing gradually. As for the variance it gradually increases from degree 1 to 15 and the error drops steeply from degree 2 to 3 and then increases. In short the model moves from underfitting to overfitting with the minimum error at degree=3. Thus we conclude that the data is best fitting for a model with degree=8 as the error is the least and it shows that a good tradeoff exists between bias and variance.



We observed that the model with degree=1 exhibited significant underfitting, with a high bias and low variance as compared to degree 2 to 10. This suggests that the model is too simple to capture the underlying patterns in the data. Additionally, we noted that there was a small gap between the training and validation error, indicating that the data is relatively simple and easy to model.

We observed that the model with degree=2 to 10 achieved its best fit when the bias and variance were both low as compared to others. indicating that the model was able to capture the underlying patterns in the data without overfitting. At this point in the Bias2−Variance plot, the model was relatively simple, with a moderate number of features, and achieved high accuracy on both the training and validation sets

We observed that the model with degree= 12,14,15 exhibited significant overfitting, with a low bias and high variance with respect to other model . This suggests that the model is fitting the noise in the data, rather than the underlying patterns. Additionally, we noted that there was a large gap between the training and validation error, indicating that the data is complex and difficult to model