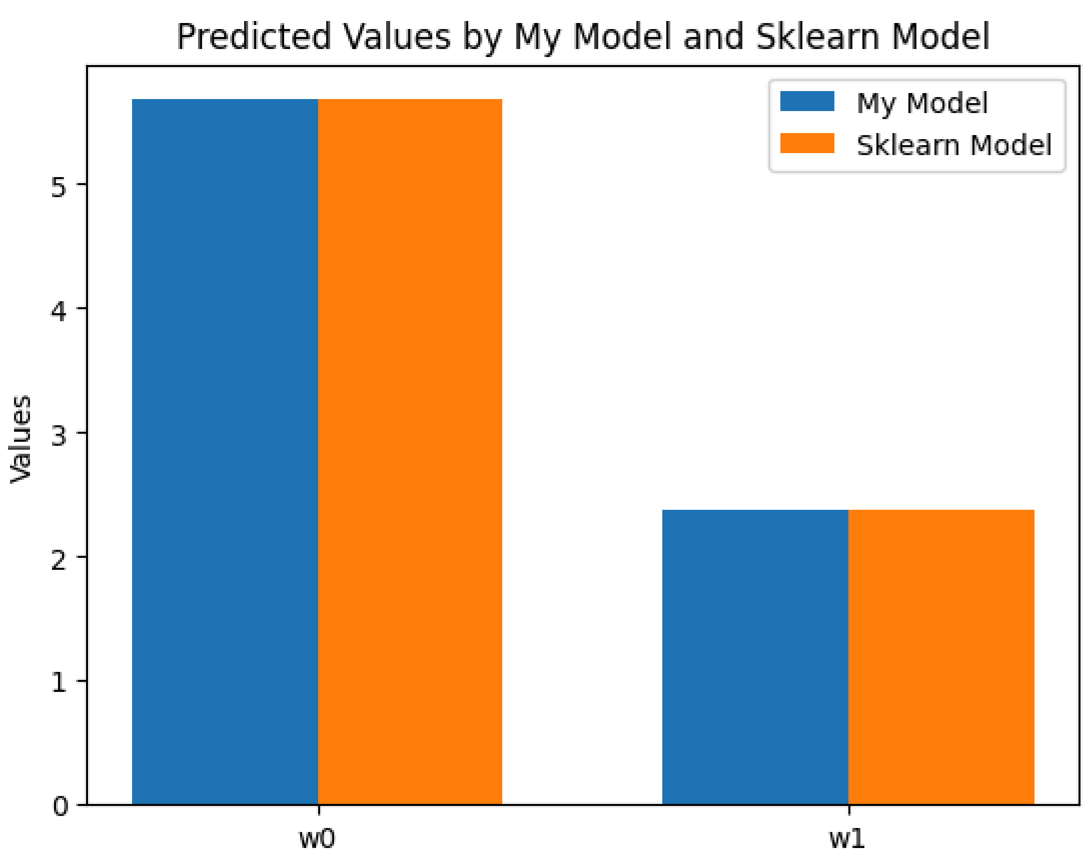
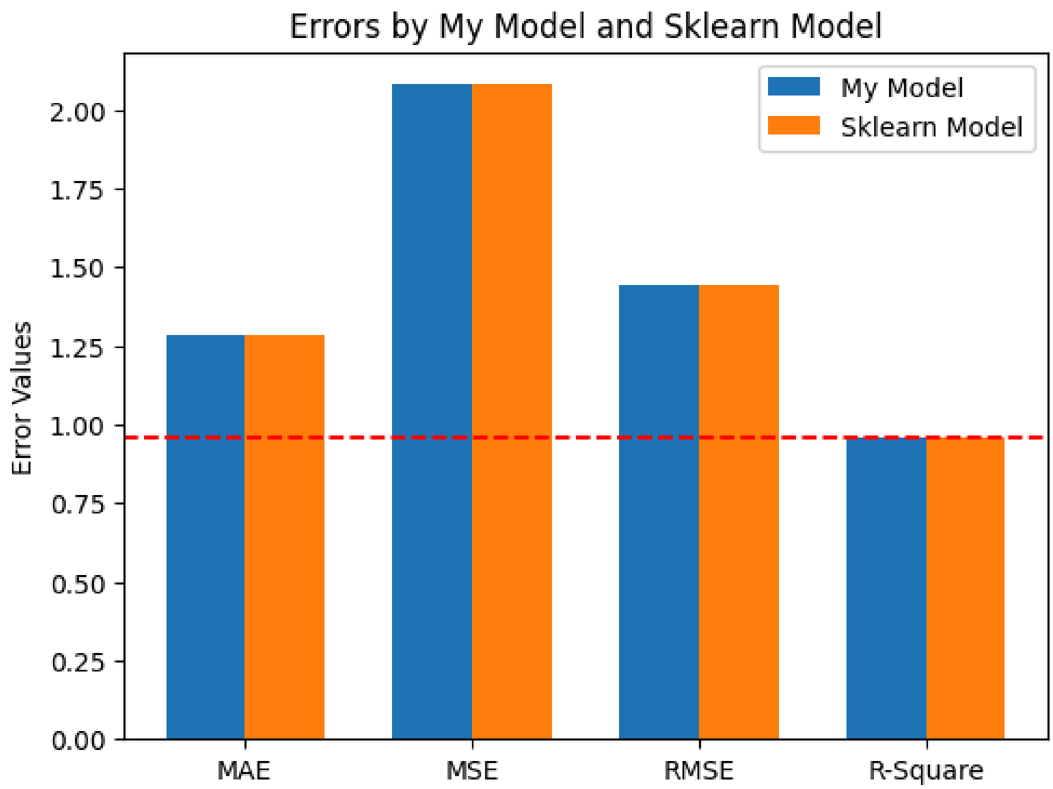
Report

Dataset 1:

Predicted_values_by_my_model = [5.680787126761226, 2.384060066057183] *#[w0, w1]*
Predicted_values_by_Sklearn_model = [5.68078713, 2.38406007]

Error_by_my_model = [1.2805559784291467, 2.0785254017773265, 1.4417091945941547, 0.9579571905586358] *#[mae, mse,rmse,r_square]*
Error_by_Sklearn_model = [1.280555978429147, 2.078525401777328, 1.4417091945941551, 0.9579571905586357]



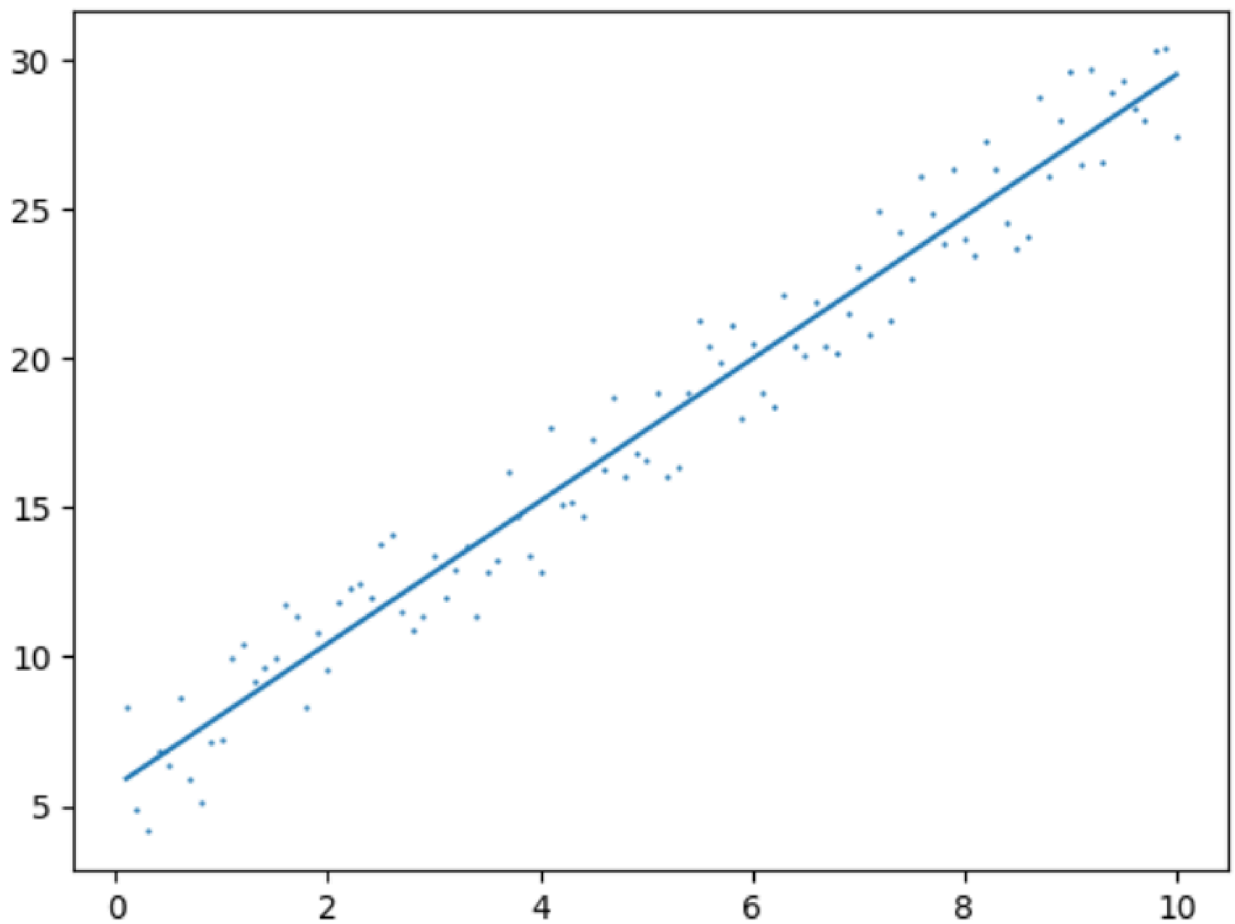


Conclusion:

In this dataset, I have applied Simple Linear Regression and got the same result as predicted by Scikit-learn Library. I have also got almost same MAE,MSE,RMSE and R_square. R_square (0.96) is close to 1. So we can say that my model is working fine with Simple Linear Regression.

Best Fit Hyperplane;

$$y = 5.681 + 2.384x$$

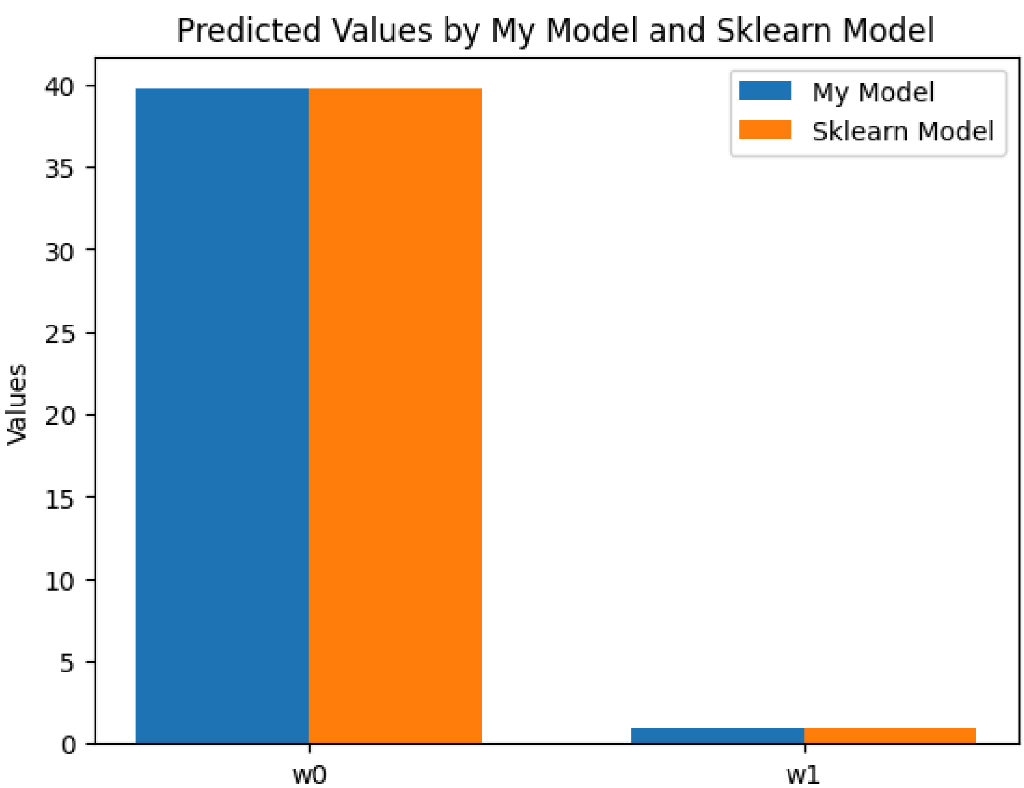


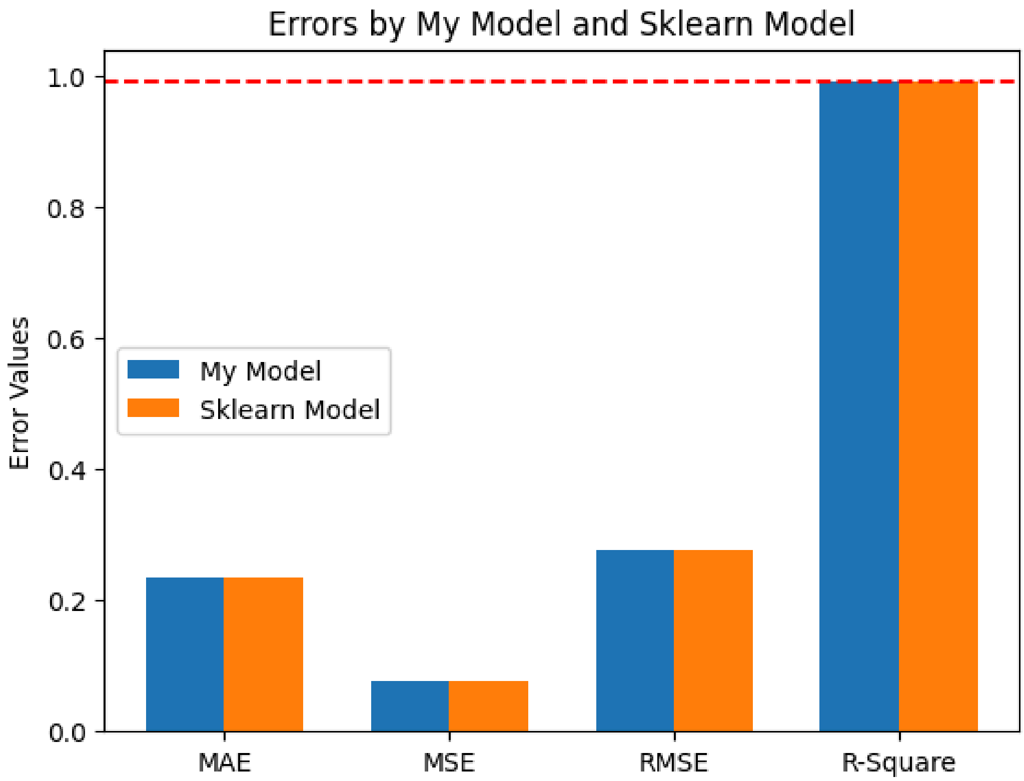
Dataset 2:

Predicted_values_by_my_model = [39.7306395177676,0.9729974518460589] *#alpha, Beta*
Predicted_values_by_Sklearn_model = [39.73063952, 0.97299745]

Error_by_my_model = [0.2349883528902577,
0.07643342704351971,
0.27646596000867757,
0.9904038522690993]

Error_by_Sklearn_model = [0.23498835289025738,
0.07643342704351966,
0.27646596000867746,
0.9904038522690993]



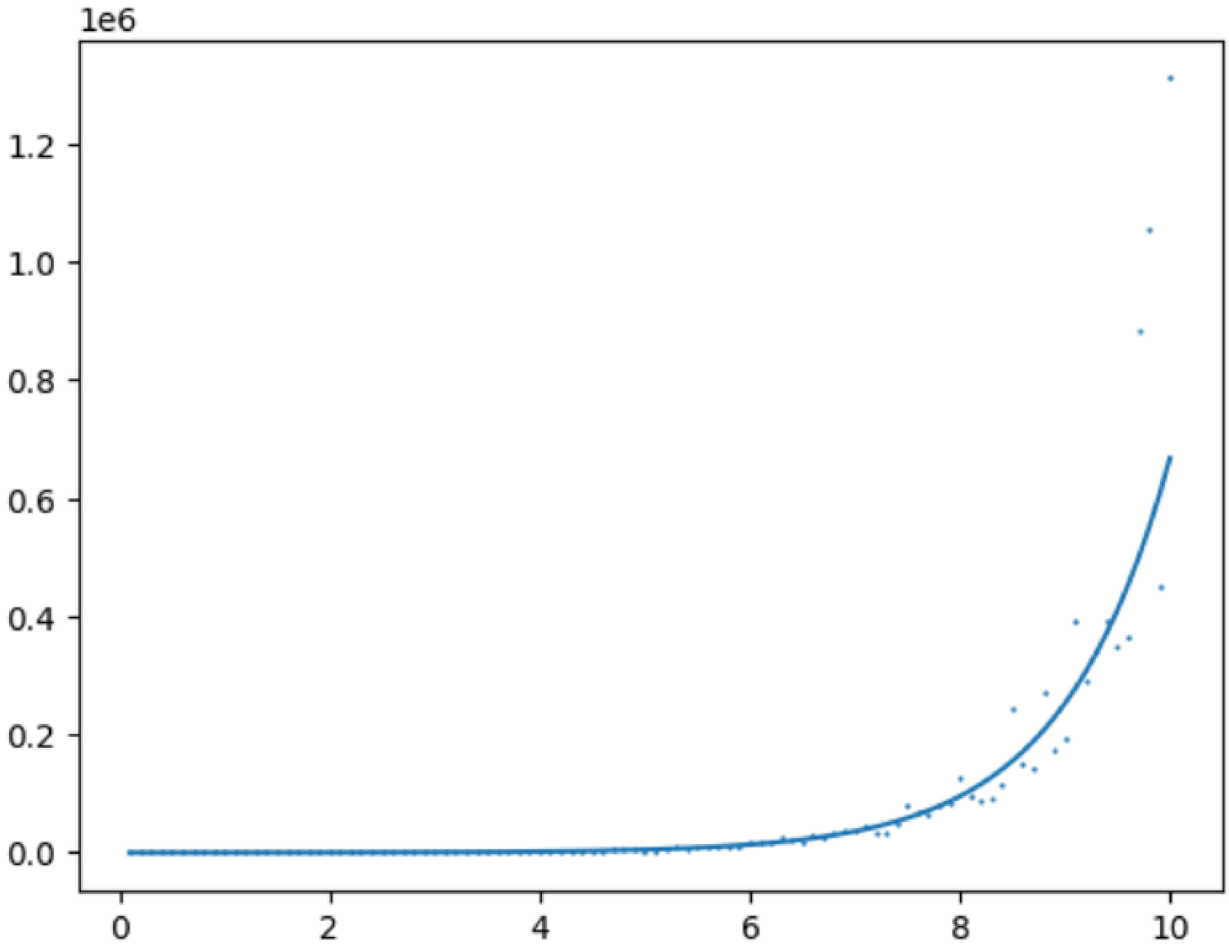


Conclusion:

In this dataset, I have applied Simple Linear Regression after **Non linear transformation** and got the same result as predicted by Scikit-learn Library. I have also got almost same MAE, MSE, RMSE and R_square. R_square (0.99) is close to 1. So we can say that my model is working fine with Simple Linear Regression.

Best Fit Hyperplane;

$$y = \log(39.73) + 0.97x$$

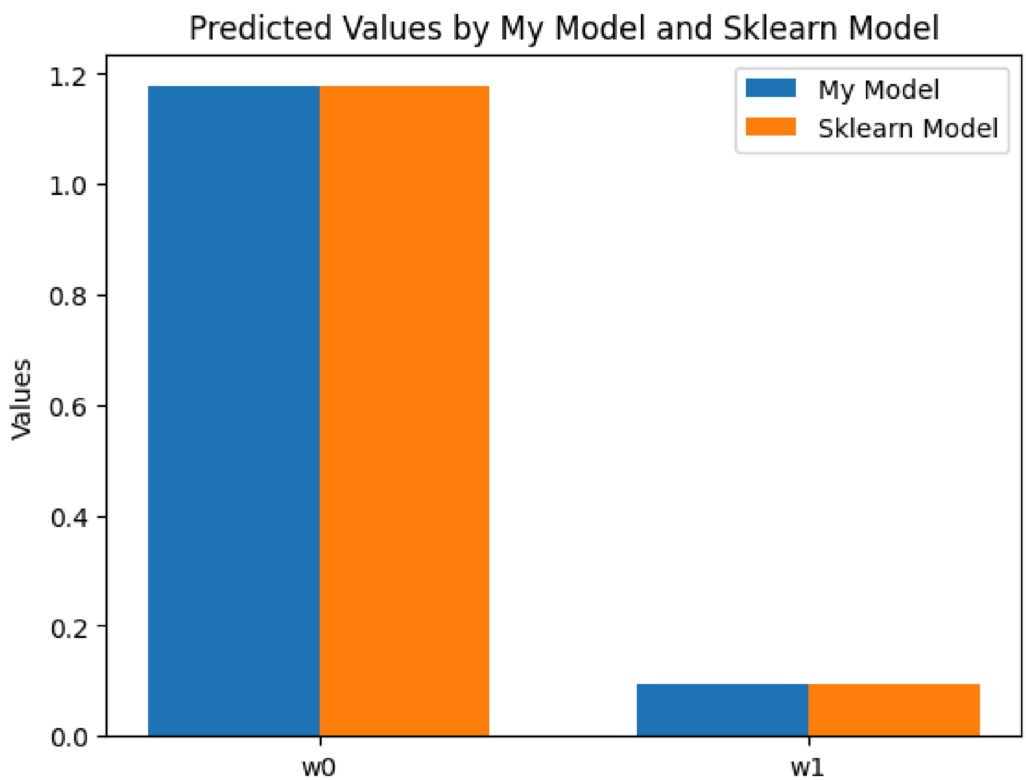


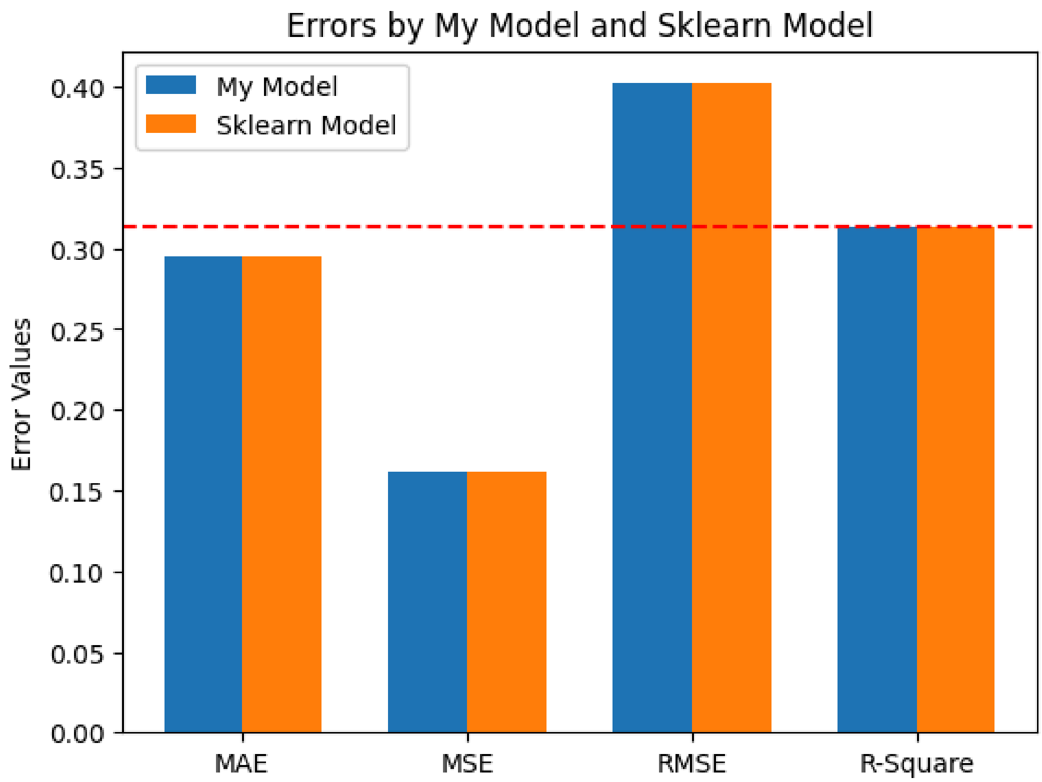
Dataset 3:

Predicted_values_by_my_model =
[1.1770620783119932,0.09419021414817955] *#w0, w1*
Predicted_values_by_Sklearn_model = [1.17706208, 0.09419021]

Error_by_my_model = [0.29467793301310385,
0.16173044143088552,
0.4021572347116057,
0.3136973226728079]

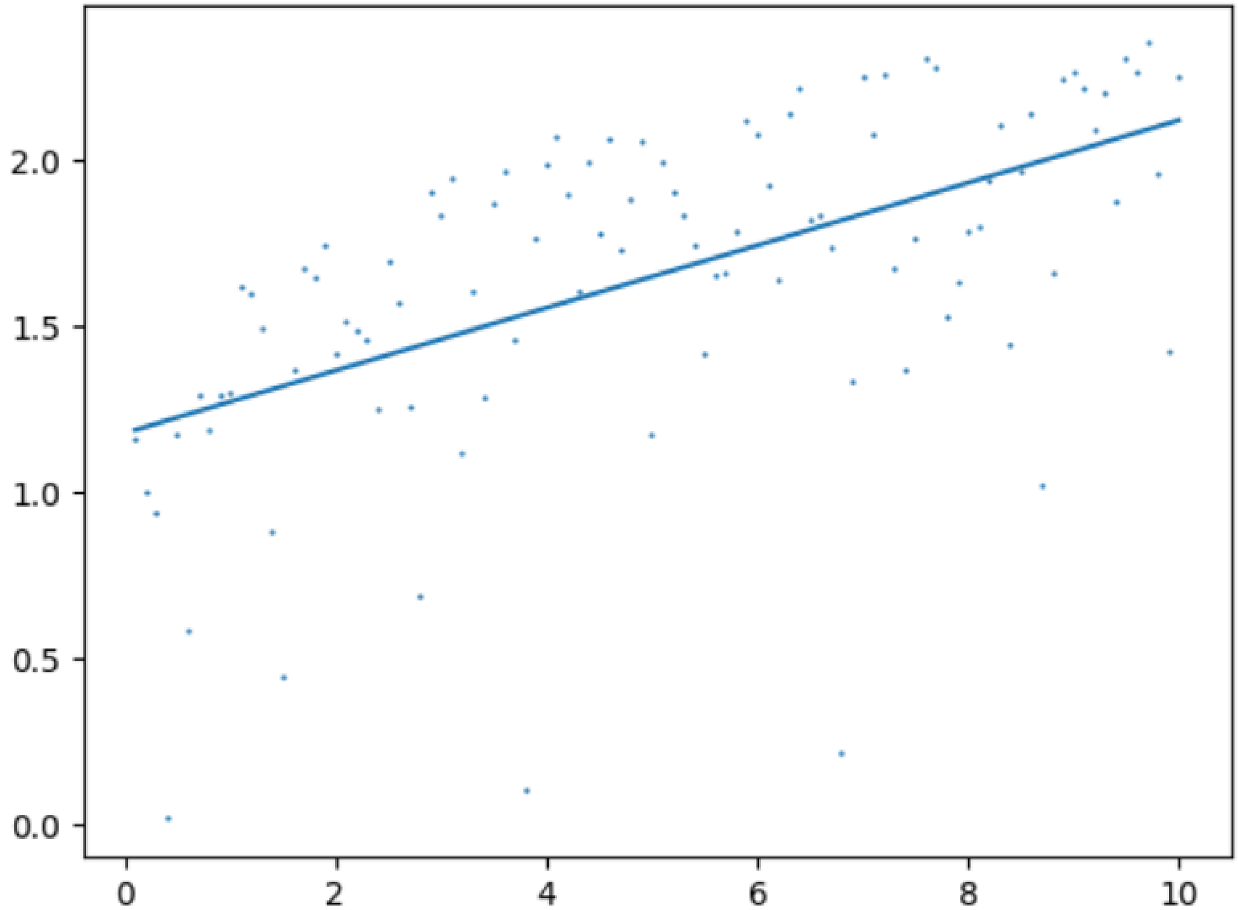
Error_by_Sklearn_model = [0.29467793301310374,
0.16173044143088552,
0.4021572347116057,
0.3136973226728079]





After applying Linear Regression model These are the approximate errors.

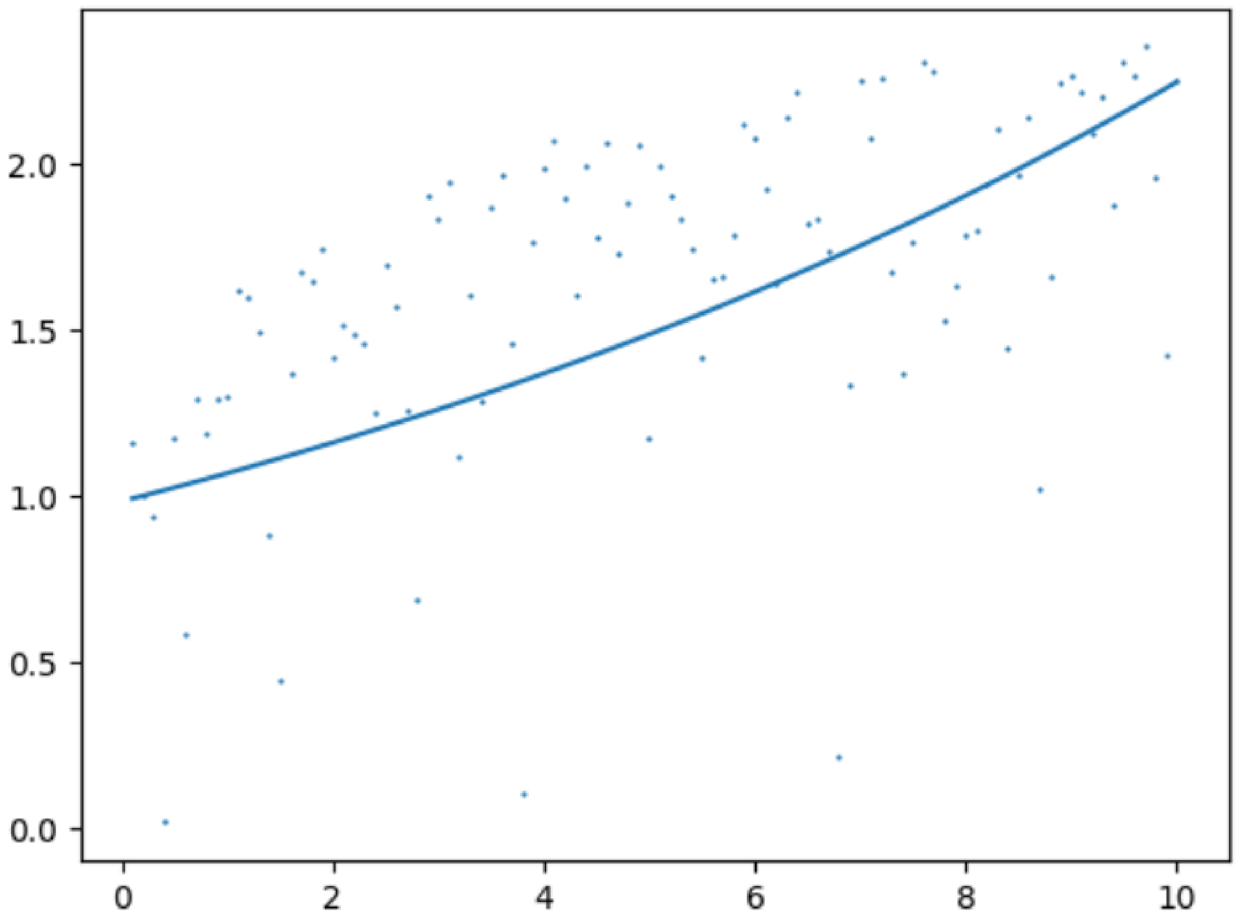
```
MAE: 0.2946779330131038
MSE: 0.1617304414308855
RMSE: 0.402157234711605
r_square: 0.3136973226728079
```



In this dataset, I have applied Simple Linear Regression and got the same result as predicted by Scikit-learn Library. I have also got almost same MAE, MSE, RMSE and it is low also.
But **R_square is 0.31** which is close to 0. So we can say that linear regression is not fit good with this dataset.

Now, Applying exponential function on same dataset.

I got following best fit line;



After applying exponential function, errors are;

```
MAE: 1.2632395065781865
MSE: 1.7323987153763363
RMSE: 1.3162061826994798
r_square: -6.351429118982884
```

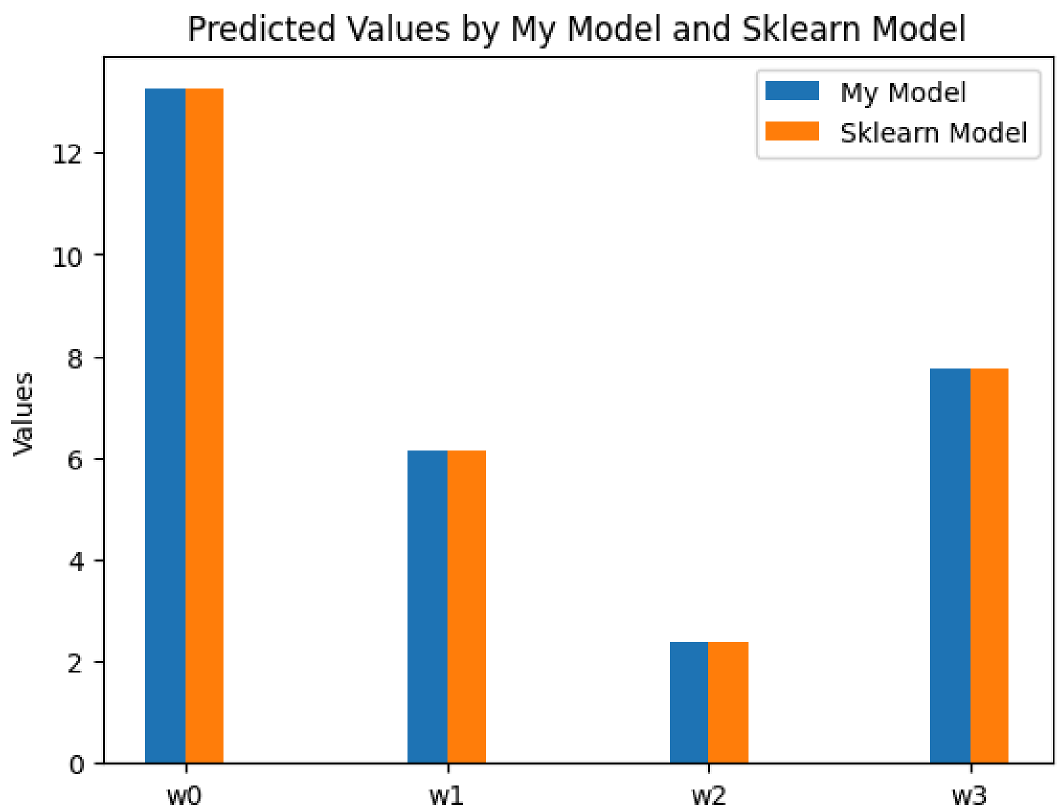
Final Conclusion:

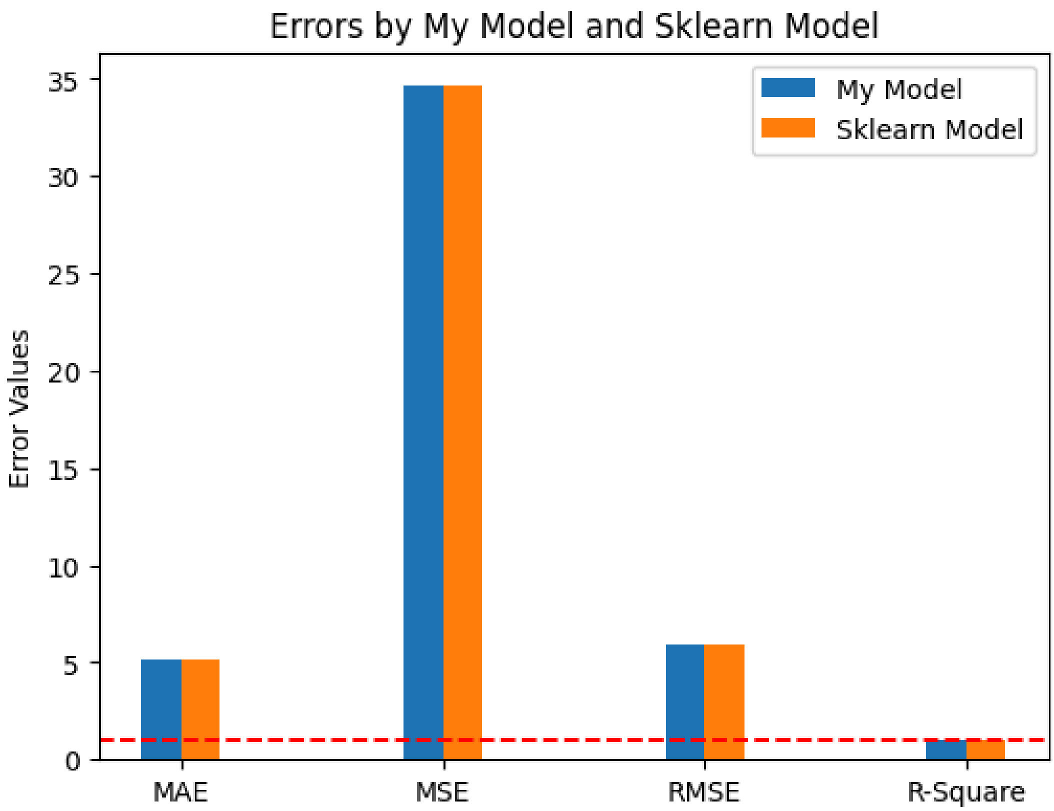
In this dataset, I have applied Simple Linear Regression then I got **R_square = 0.31** which is very low then I do **Non linear transformation** and applied exponential function on it. In this case I got **Negative R_square** which is worse. So we can say that non of the model is working fine with dataset.

Dataset 4:

Predicted_values_by_my_model = [13.239477941118366, 6.132437615313393, 2.392265549258809, 7.746810380660236]
#w0, w1, w2, w3
Predicted_values_by_Sklearn_model = [13.239477824445359, 6.13243763, 2.39226554, 7.74681038]

Error_by_my_model = [5.155505639902197, 34.62048082924356, 5.883917133104745, 0.9841749058943147]
Error_by_Sklearn_model = [5.15550562646378, 34.62048082924356, 5.883917133104745, 0.9841749058943147]





Conclusion:

In this dataset, I have applied multiple Simple Linear Regression and got the same result as predicted by Scikit-learn Library. I have also got almost same MAE, MSE, RMSE and it is low also. R_square is 0.98 which is close to 1. So we can say that multiple linear regression is fit good with this dataset.

Best Fit Hyperplane: 13.24 + 6.13*x1 + 2.39*x2 +7.75*x3

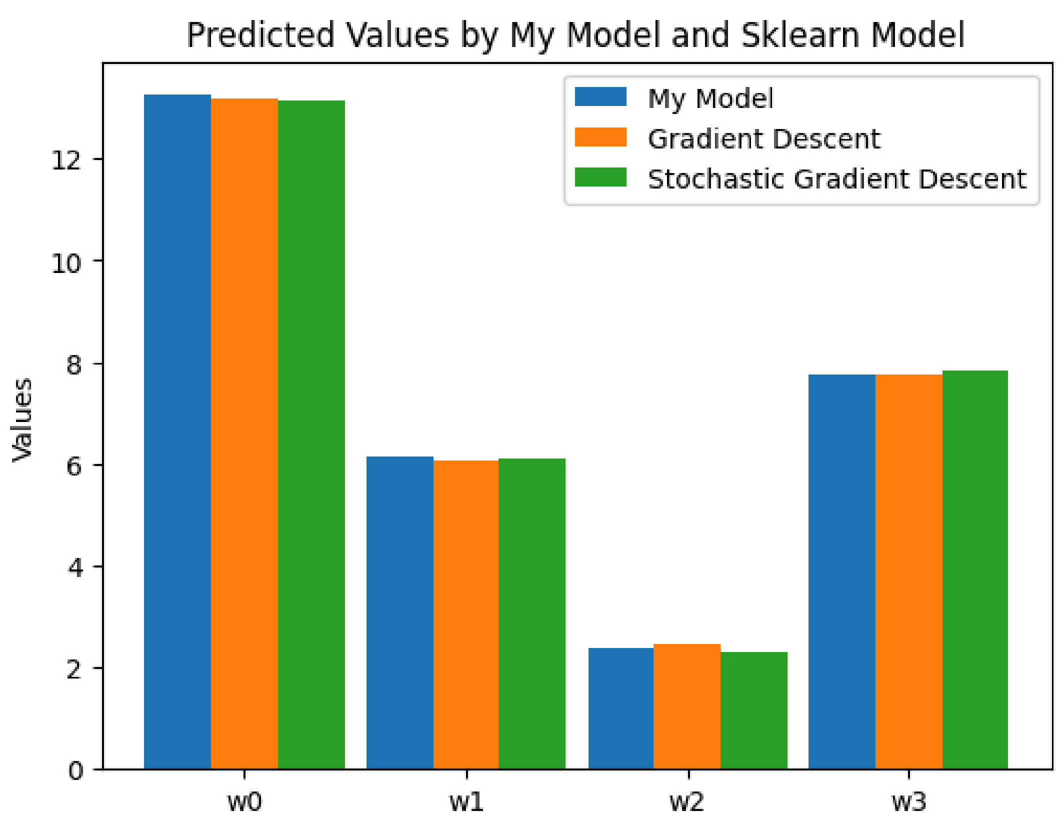
Gradient Descent v/s Stochastic Gradient Descent on Data4.csv

Prediction of w0, w1, w2, w3 by My_model, GD and SGD;

Predicted_values_by_my_model **=** [13.239477941118366, 6.132437615313393, 2.392265549258809, 7.746810380660236] *#[w0, w1, w2, w3]*

Predicted_values_by_GD **=** [13.18578203958556, 6.060680905255514, 2.445134785871812, 7.766764500756425]

Predicted_values_by_SGD **=** [13.154571488555078, 6.091250059460483, 2.3134524409841464, 7.827527898786717]

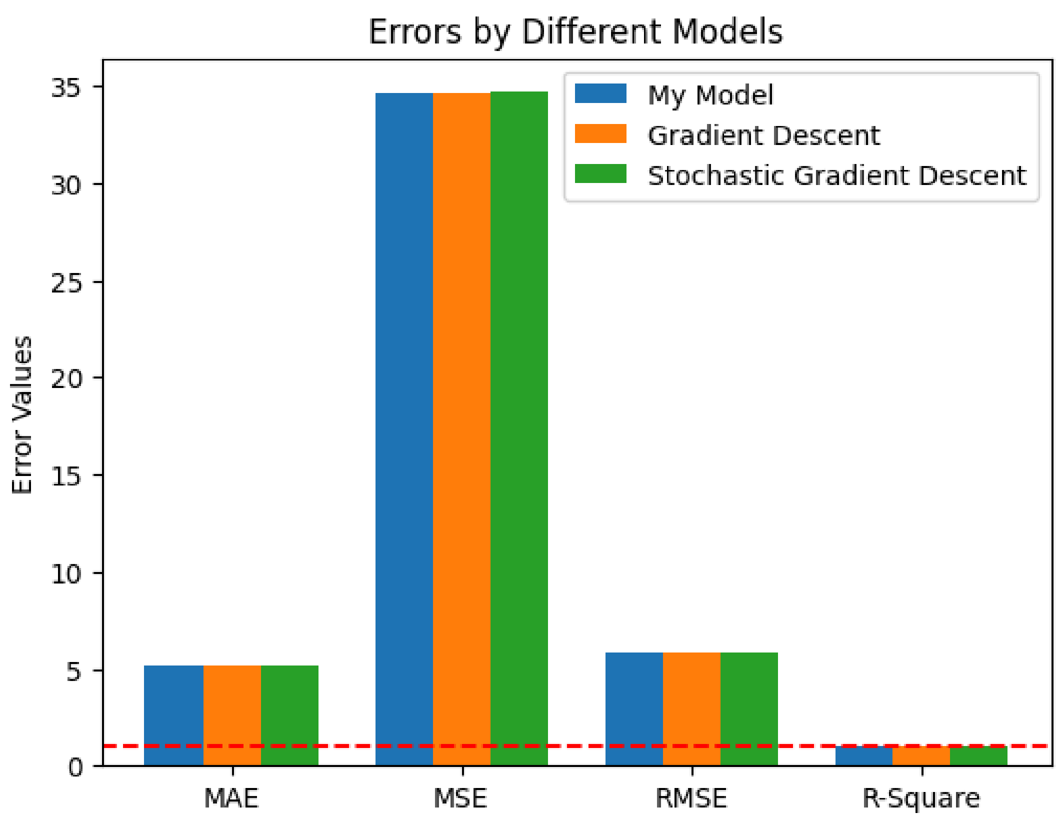


Error among My_model, GD and SGD;

Error_by_my_model **=** [5.155505639902197, 34.62048082924356, 5.883917133104745, 0.9841749058943147]

Error_by_GD **=** [5.155022267892361, 34.62081600683792, 5.883945615557465, 0.9841749058943147]

Error_by_SGD **=** [5.119894250543379, 34.68046898055256, 5.889012564136076, 0.9841749058943147]



Conclusion:

I have predicted w0, w1, w2 and w3 using both GD and SGD. I found that GD is giving better result than SGD and result of SGD is updated every time when I run the algorithm(due to randomness). I have also got almost same MAE, MSE, RMSE and it is low also. R_square is 0.98 which is close to 1. So we can say that GD and SGD is predicting approximately same.

Nagmani Kumar