Name of The Project

Exploring various molecular property through Machine Learning.

Name : Manadip Sutradhar

Project supervisor name : Dr Debashree Gosh.

Institute : Indian Association for the cultivation of science.

Initially,I utilize a Neural Network Machine Learning model to predict various molecular properties.

Data Analysis for the property-1

Neural Network:-

Learning Rate	No of Hidden Layer	No of Input Nodes	No of Hidden Nodes	Activation Functiom	Training Dataset	Testing Data set	Epochs
0.001	1	210	100	relu	90%	10%	1000

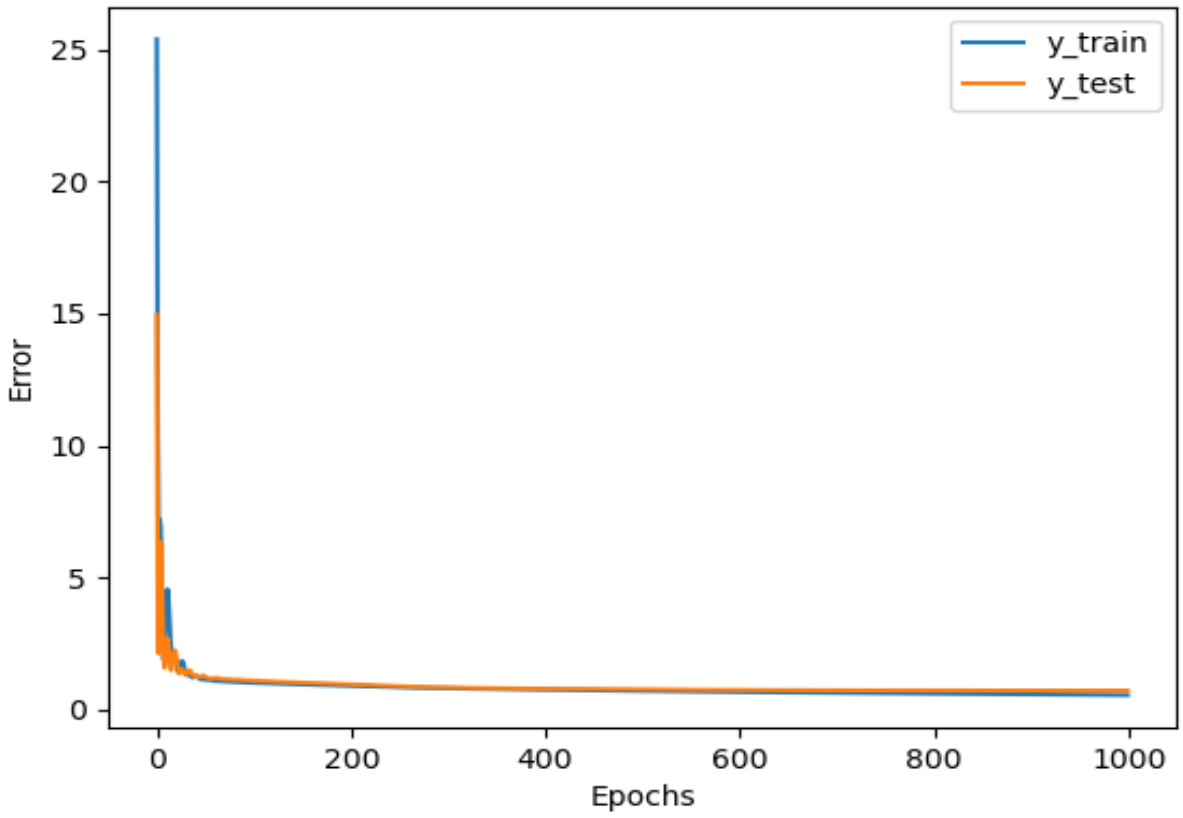
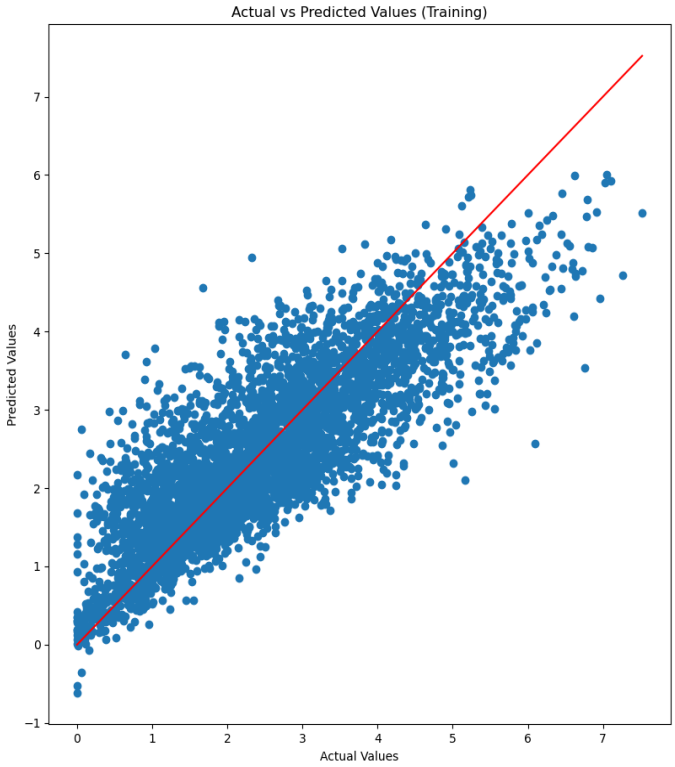
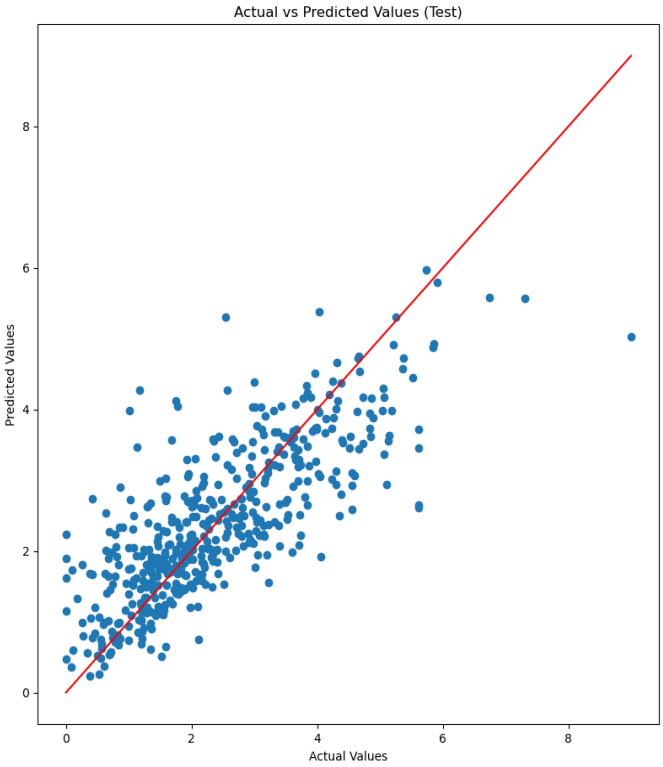


Figure1:-Mean Squared Error (MSE) Curves on the training data for the Feed-Forward Neural Network for 100 hidden Nodes with 1000 epoches in each trial.





Actual vs predicted output for training

Actual vs predicted output for testing

Training loss at different EPOCHS and different Test Size:-

Test Size	1K Epochs	1.5K Epochs	2k Epochs	2.5 k Epochs	3k Epochs
0.1	0.43	0.42	0.35	0.34	0.31
0.2	0.518	0.434	0.367	0.314	0.278
0.3	0.513	0.406	0.333	0.293	0.279

Testing loss at different EPOCHS and different Test Size:-

Test Size	1K Epochs	1.5K Epochs	2k Epochs	2.5 k Epochs	3k Epochs
0.1	0.69	0.68	0.71	0.75	0.78
0.2	0.3753	0.753	0.776	0.802	0.811
0.3	0.766	0.786	0.859	0.906	0.953

Conclusion:-
From Figure 1, we can infer that although the training and testing curves do not exhibit significant overfitting, neither the training nor the testing performance is satisfactory. Consequently, the curves converge little bit but the model is not well-fitted.

Data Analysis for the property-2

Neural Network:-

Learning Rate	No of Hidden Layer	No of Input Nodes	No of Hidden Nodes	Activation Function	Training Dataset	Testing Data set	Epochs
0.001	1	210	100	relu	80%	20%	1000

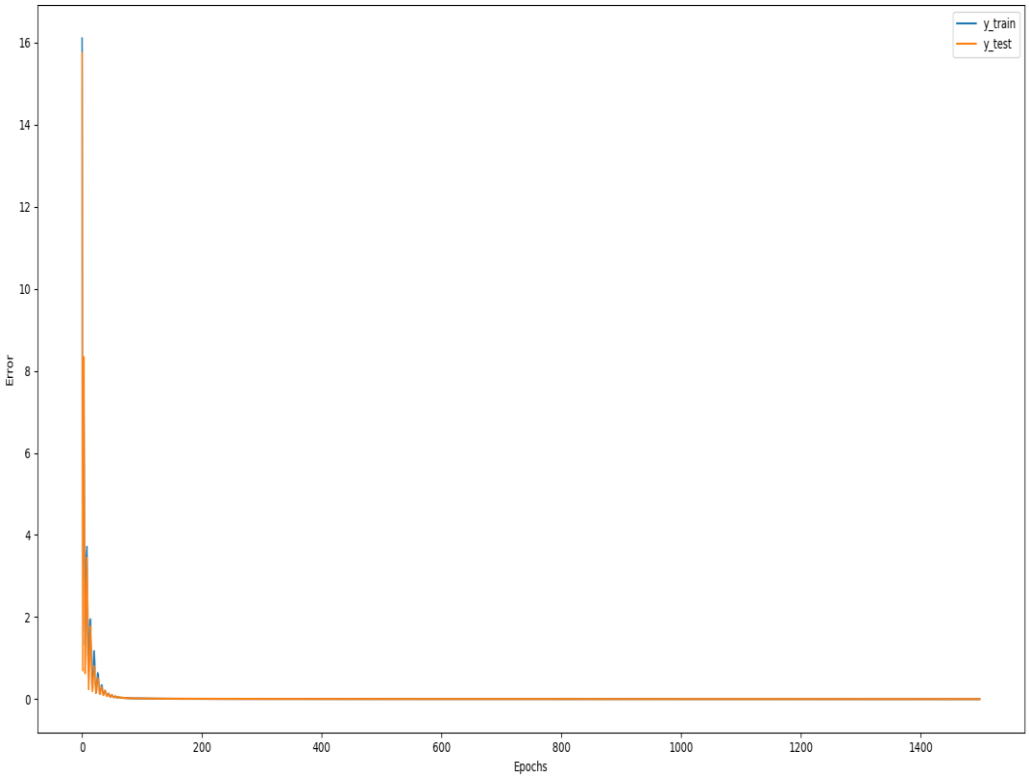
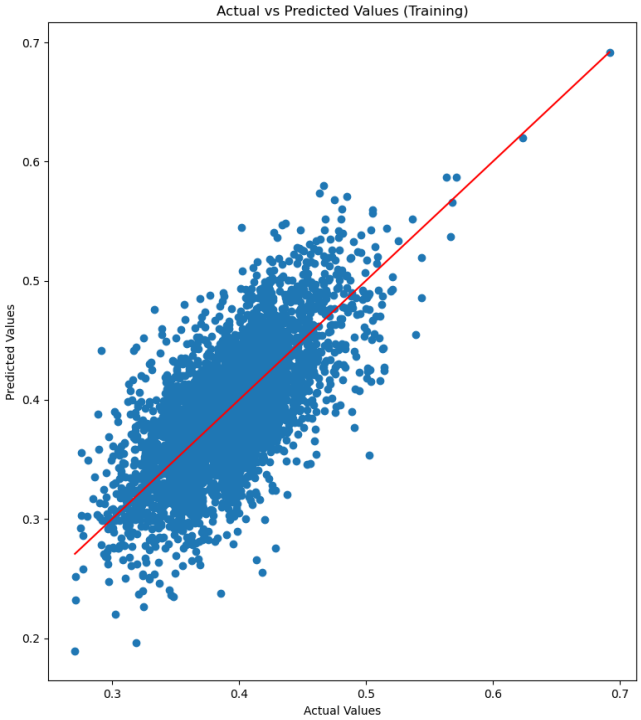
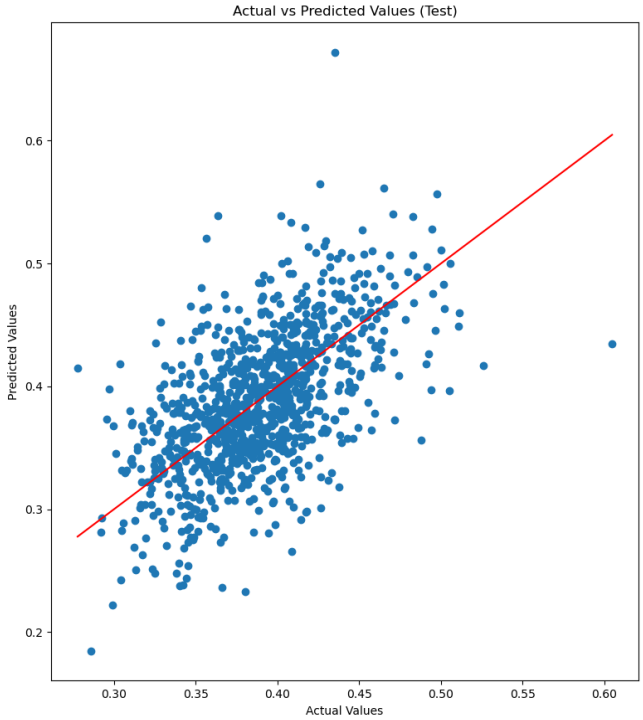


Figure2-Mean Squared Error (MSE) Curves on the training data for the Feed-Forward Neural Network for 100 hidden Nodes with 1000 epoches in each trial.





Actual vs predicted output for training

Actual vs predicted output for testing

Training loss at different EPOCHS and different Test Size :-

Test Size	1K Epochs	1.5K Epochs	2k Epochs	2.5 k Epochs	3k Epochs
0.1	0.001478	0.0009386	0.0006712	0.0005520	0.0004639
0.2	0.001456	0.00092	0.000737	0.000938	0.000454
0.3	0.00131	0.00085	0.0027	0.00059	0.00044

Testing loss at different EPOCHS and different Test Size:-

Test Size	1K Epochs	1.5K Epochs	2k Epochs	2.5 k Epochs	3k Epochs
0.1	0.002174	0.001479	0.001118	0.000890	0.0008169
0.2	0.002223	0.00159	0.00131	0.00137	0.00087
0.3	0.00232	0.00181	0.0035	0.00129	0.00104

Conclusion:-Like the property-1 , property-2 also is neither well trained nor tested.

Data Analysis for the property-3

Neural Network:-

Learning Rate	No of Hidden Layer	No of Input Nodes	No of Hidden Nodes	Activation Functiom	Training Dataset	Testing Data set	Epochs
0.001	1	210	100	relu	80%	20%	1000

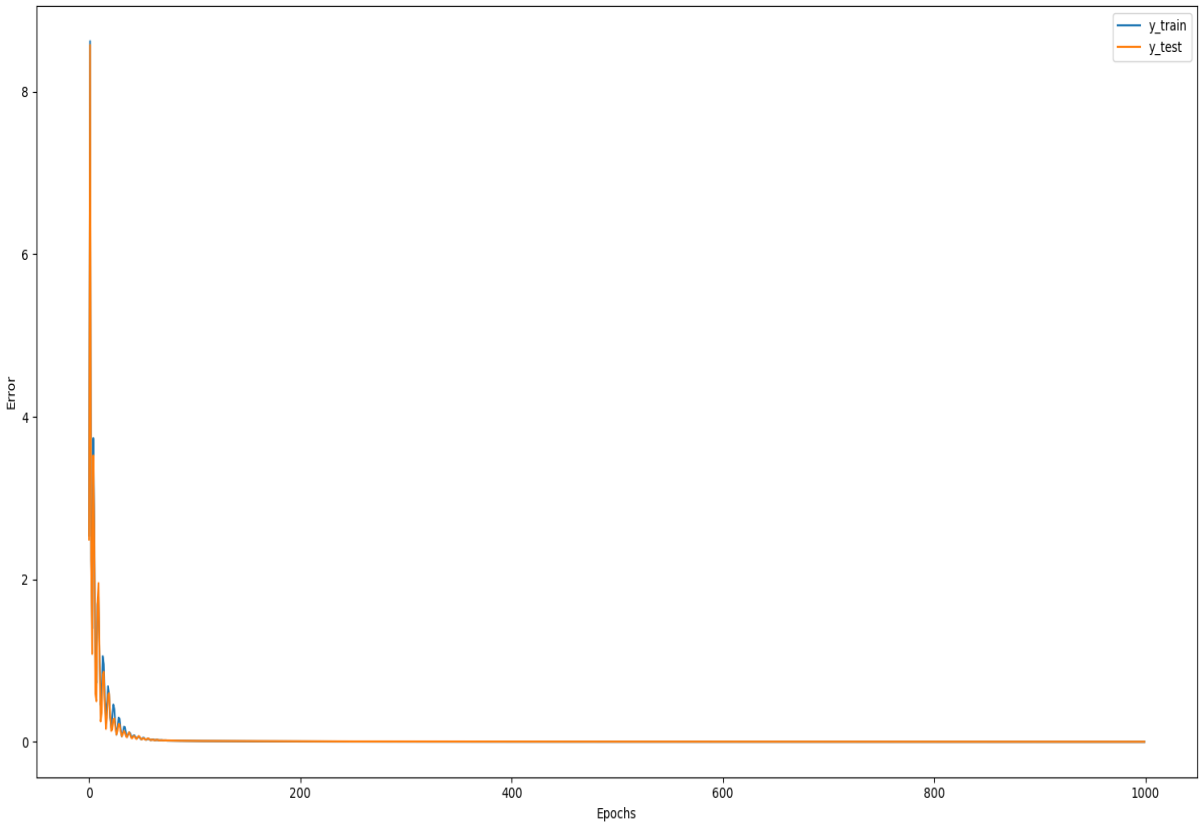
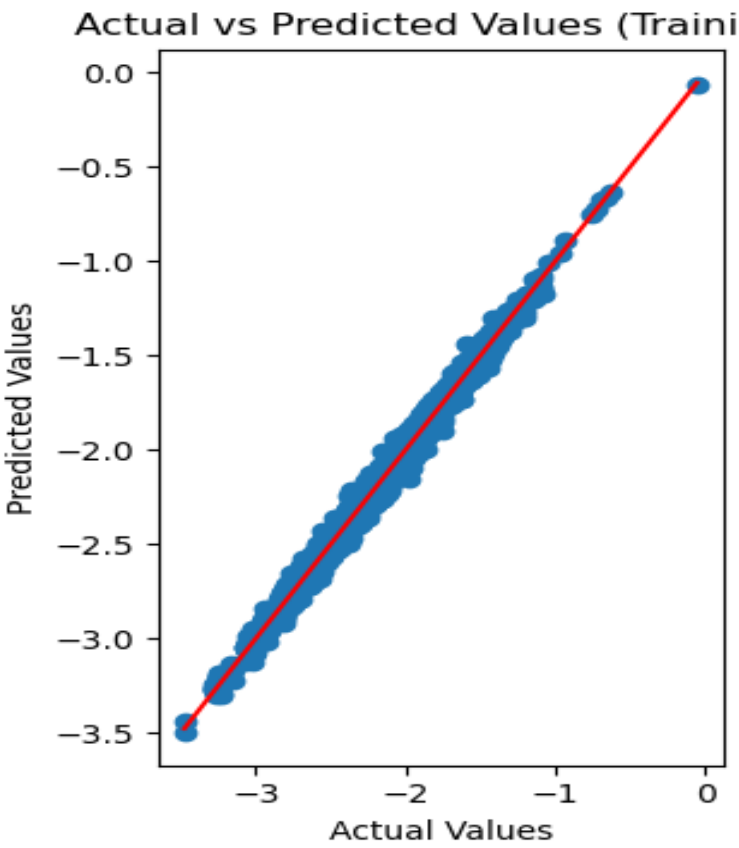
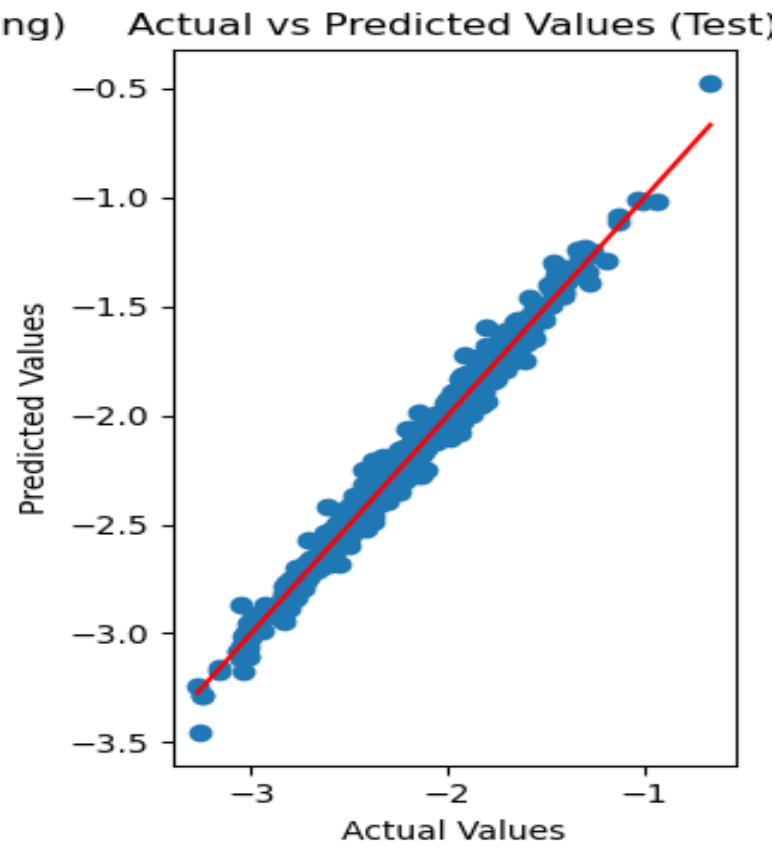


Figure-Mean Squared Error (MSE) Curves on the training data for the Feed-Forward Neural Network for 100 hidden Nodes with 1000 epoches in each trial.





Actual vs predicted output for training

Actual vs predicted output for testing

Training loss at different EPOCHS and different Test Size :-

Test Size	1K Epochs	1.5K Epochs	2k Epochs	2.5 k Epochs	3k Epochs
0.1	0.0011	0.0066	0.0005	0.0003	0.00033
0.2	0.00109	0.0011	0.0005	0.0040	0.0003

Testing loss at different EPOCHS and different Test Size:-

Test Size	1K Epochs	1.5K Epochs	2k Epochs	2.5 k Epochs	3k Epochs
0.1	0.00214	0.0071	0.0012	0.00103	0.0009
0.2	0.00216	0.0058	0.0013	0.0040	0.0010

Conclusion:_

For the property 3 at any ratio of Training and Testing Data Set are well fitted.

Data Analysis fro the property-4

Nural Network:-

Learning Rate	No of Hidden Layer	No of Input Nodes	No of Hidden Nodes	Activation Functiom	Training Dataset	Testing Data set	Epochs
0.001	1	210	100	relu	80%	20%	1000

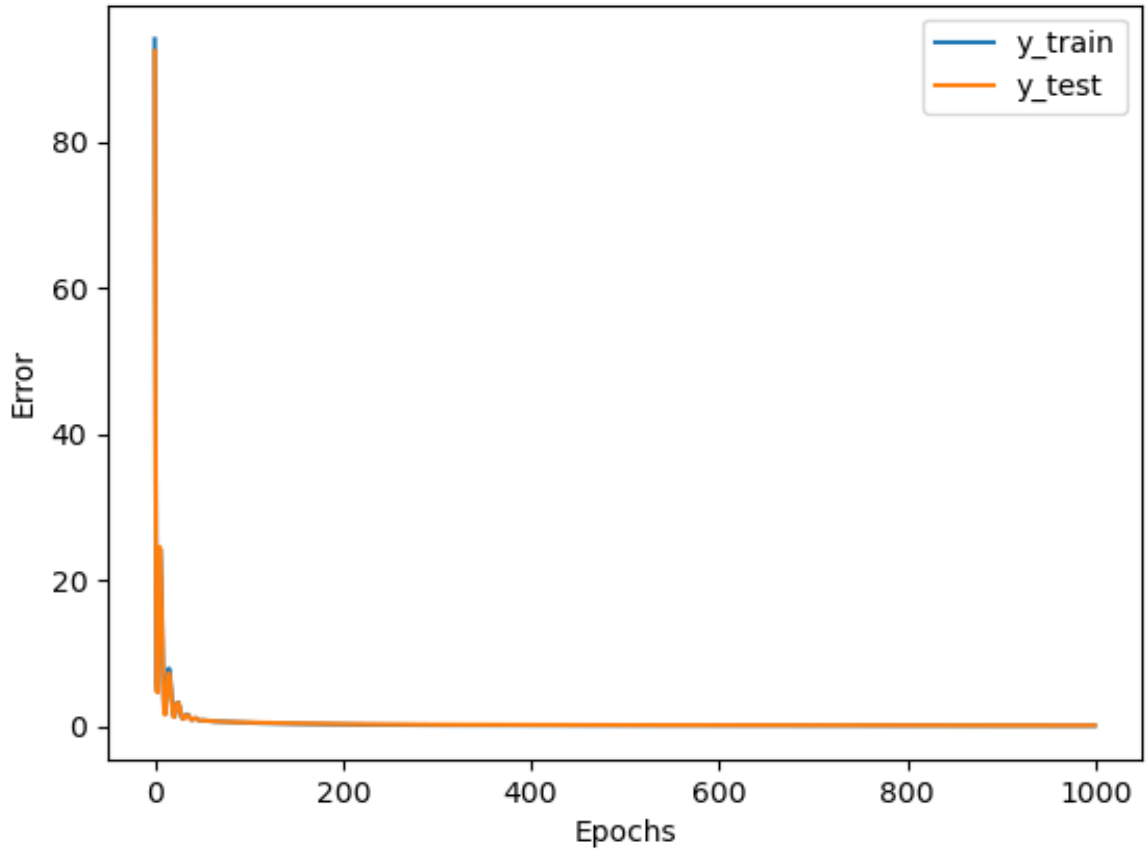
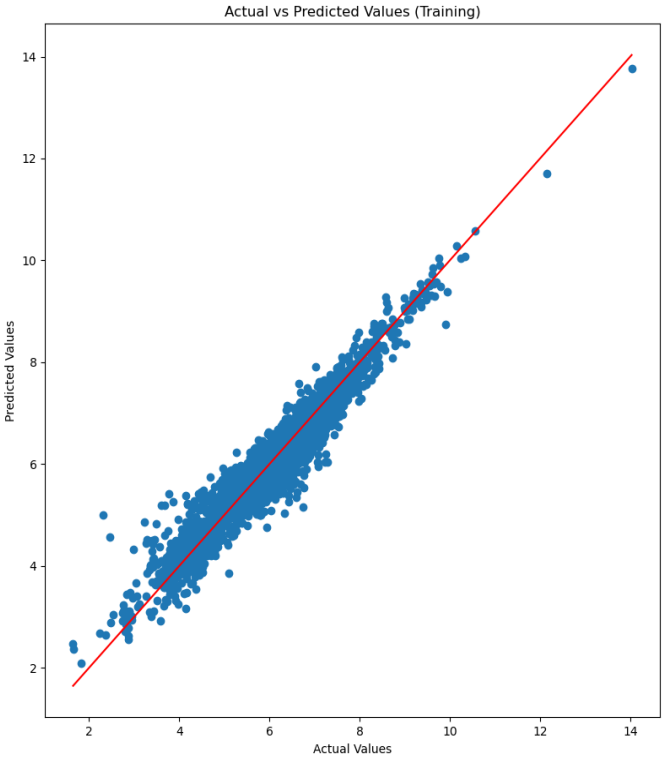
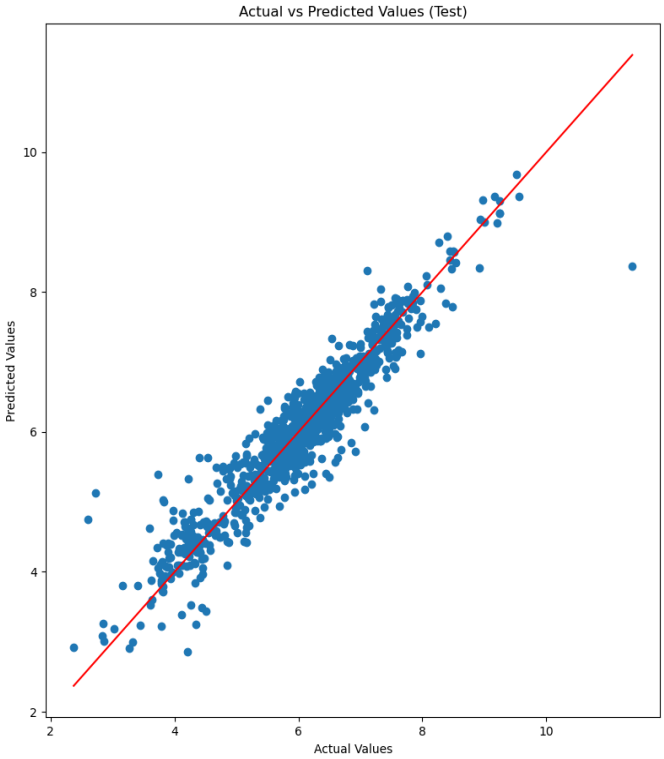


Figure-Mean Squared Error (MSE) Curves on the training data for the Feed-Forward Neural Network for 100 hidden Nodes with 1000 epoches in each trial.





Actual vs predicted output for training

Actual vs predicted output for testing

Training loss at different EPOCHS and different Test Size :-

Test Size	1K Epochs	1.5K Epochs	2k Epochs	2.5 k Epochs	3k Epochs
0.1	0.094	0.072	0.059	0.051	0.043
0.2	0.093	0.068	0.057	0.047	0.070

Testing loss at different EPOCHS and different Test Size:-

Test Size	1K Epochs	1.5K Epochs	2k Epochs	2.5 k Epochs	3k Epochs
0.1	0.153	0.12	0.112	0.105	0.101
0.2	0.131	0.068	0.108	0.101	0.120

Conclusion:-

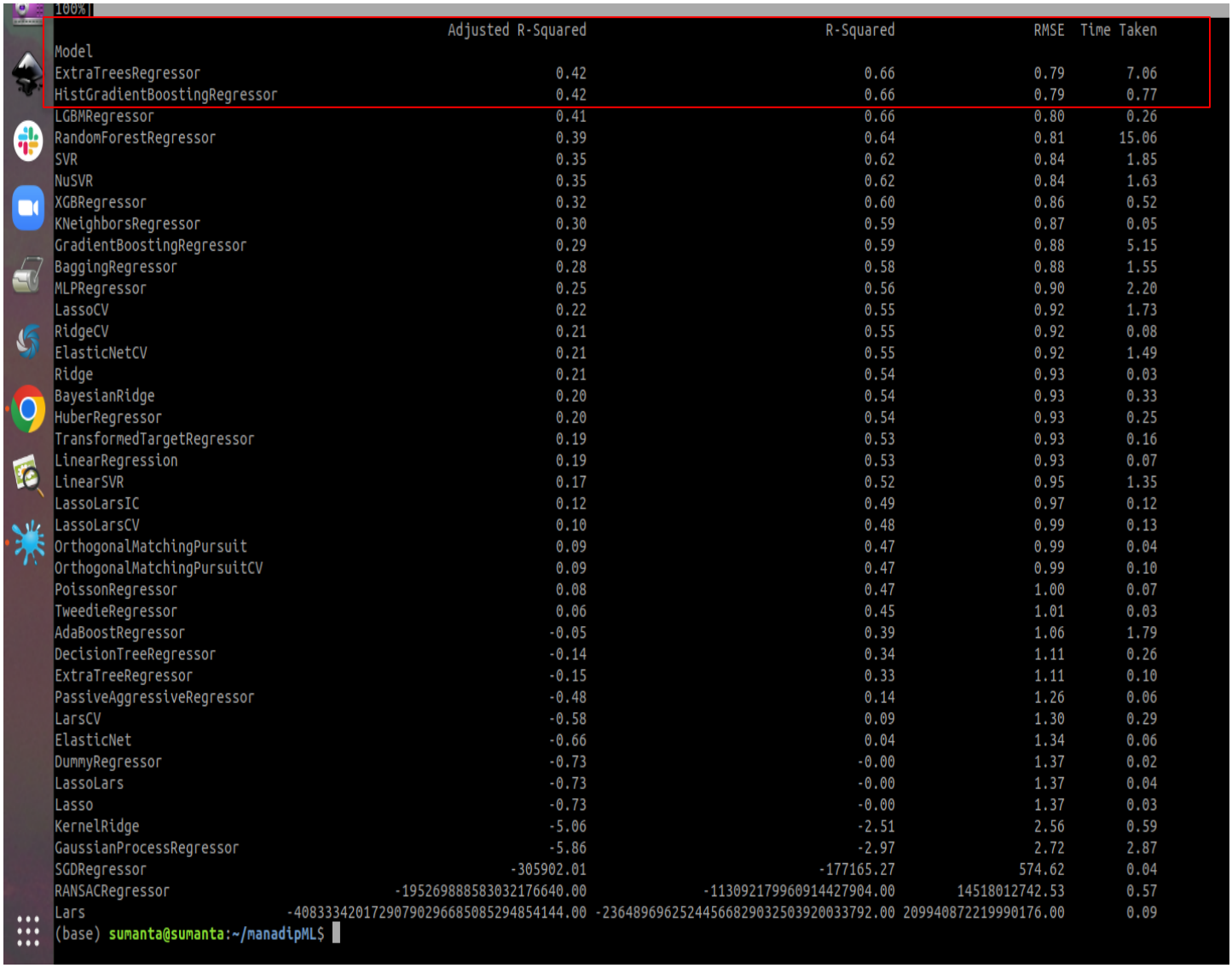
Also for the property4 at any ratio of Training and Testing Data are well fitted.

Conclusion: Based on the above analysis, it appears that properties 1 and 2 are not suitable for the Neural Network algorithm, whereas properties 3 and 4 demonstrate a good fit for neural networks.

So, now I am ging to apply another machine learning Algorithm on this data set :-

LAAZY-PREDICT:-

Lazy Predict provides a comprehensive array of 41 regression machine learning algorithms, systematically evaluating their RMSE performance to aid in model selection.



100%

Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
ExtraTreesRegressor	0.42	0.66	0.79	7.06
HistGradientBoostingRegressor	0.42	0.66	0.79	0.77
LGBMRegressor	0.41	0.66	0.80	0.26
RandomForestRegressor	0.39	0.64	0.81	15.06
SVR	0.35	0.62	0.84	1.85
NuSVR	0.35	0.62	0.84	1.63
XGBRegressor	0.32	0.60	0.86	0.52
KNeighborsRegressor	0.30	0.59	0.87	0.05
GradientBoostingRegressor	0.29	0.59	0.88	5.15
BaggingRegressor	0.28	0.58	0.88	1.55
MLPRegressor	0.25	0.56	0.90	2.20
LassoCV	0.22	0.55	0.92	1.73
RidgeCV	0.21	0.55	0.92	0.08
ElasticNetCV	0.21	0.55	0.92	1.49
Ridge	0.21	0.54	0.93	0.03
BayesianRidge	0.20	0.54	0.93	0.33
HuberRegressor	0.20	0.54	0.93	0.25
TransformedTargetRegressor	0.19	0.53	0.93	0.16
LinearRegression	0.19	0.53	0.93	0.07
LinearSVR	0.17	0.52	0.95	1.35
LassoLarsIC	0.12	0.49	0.97	0.12
LassoLarsCV	0.10	0.48	0.99	0.13
OrthogonalMatchingPursuit	0.09	0.47	0.99	0.04
OrthogonalMatchingPursuitCV	0.09	0.47	0.99	0.10
PoissonRegressor	0.08	0.47	1.00	0.07
TweedieRegressor	0.06	0.45	1.01	0.03
AdaBoostRegressor	-0.05	0.39	1.06	1.79
DecisionTreeRegressor	-0.14	0.34	1.11	0.26
ExtraTreeRegressor	-0.15	0.33	1.11	0.10
PassiveAggressiveRegressor	-0.48	0.14	1.26	0.06
LarsCV	-0.58	0.09	1.30	0.29
ElasticNet	-0.66	0.04	1.34	0.06
DummyRegressor	-0.73	-0.00	1.37	0.02
LassoLars	-0.73	-0.00	1.37	0.04
Lasso	-0.73	-0.00	1.37	0.03
KernelRidge	-5.06	-2.51	2.56	0.59
GaussianProcessRegressor	-5.86	-2.97	2.72	2.87
SGDRegressor	-305902.01	-177165.27	574.62	0.04
RANSACRegressor	-1952698885830321766 40.00	-1130921799609144279 04.00	14518012742.53	0.57
Lars	-4083334201729079029668508529485414 4.00	-2364896962524456682903250392003379 2.00	209940872219990176.00	0.09

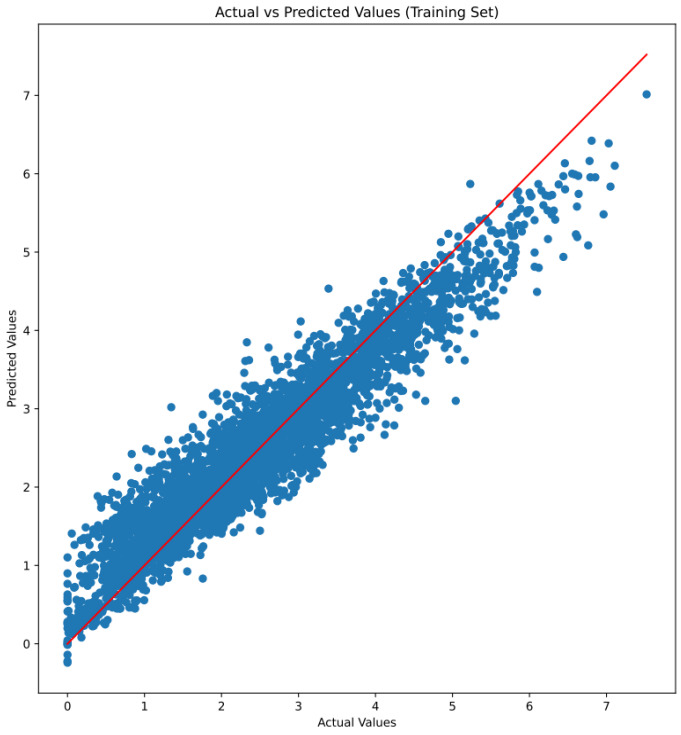
(base) sumanta@sumanta:~/manadipML$

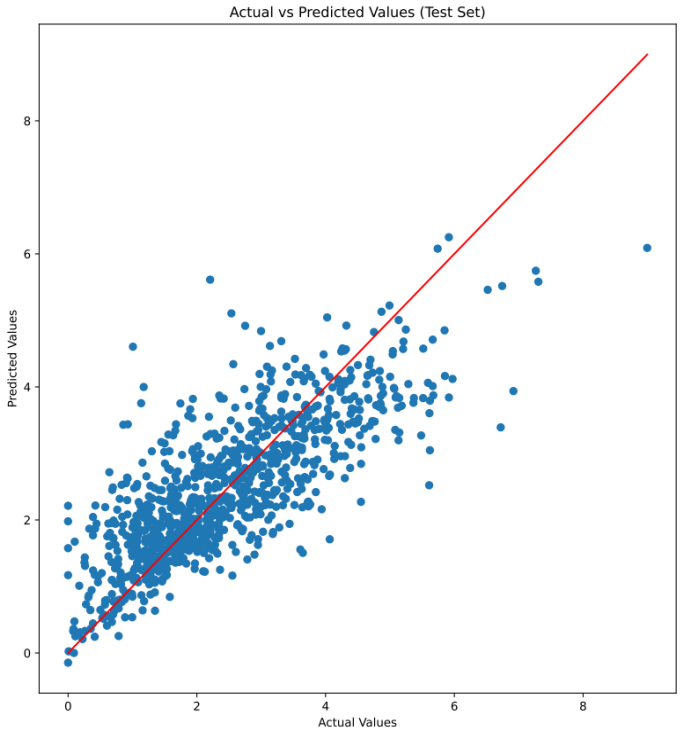
Model	RMSE	Time Taken
ExtraTreesRegressor	0.79	7.06
HistGradinetBoostingRegressor	0.79	0.77

Since the HistGradientBoosting Regressor exhibits lower RMSE and requires less time than other models, I have decided to apply the HistGradientBoosting Model to my dataset across various properties (1-4).

Data Analysis For the property -1

HistGRadientBoosting Regressor





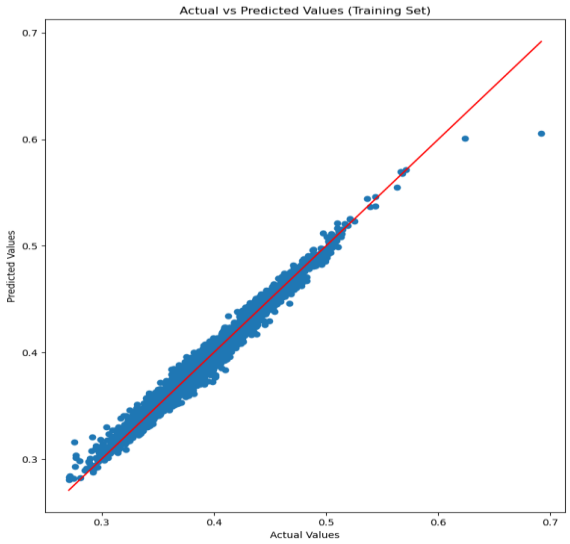
Actual vs predicted output for training
(4000 Data in Training)

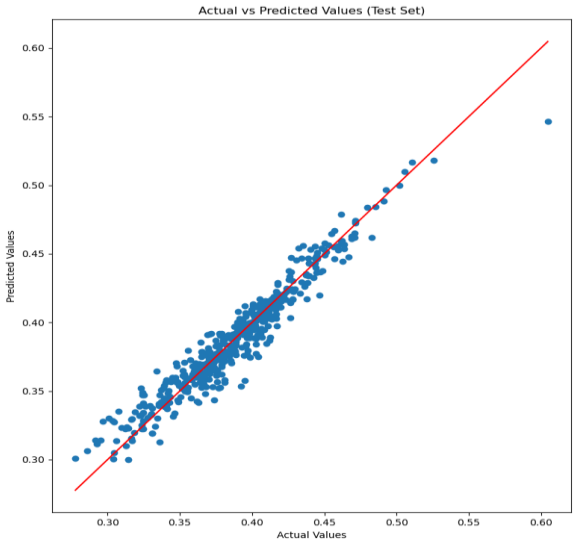
Actual vs predicted output for testing
(1000 Data in Testing)

So, the Gradient Boosting algorithm shows somewhat better performance compared to the Neural Network algorithm. Its Training and Testing results indicate a stronger fit than those of the Neural Network.

For the property -2

HistGRadientBoosting Regressor



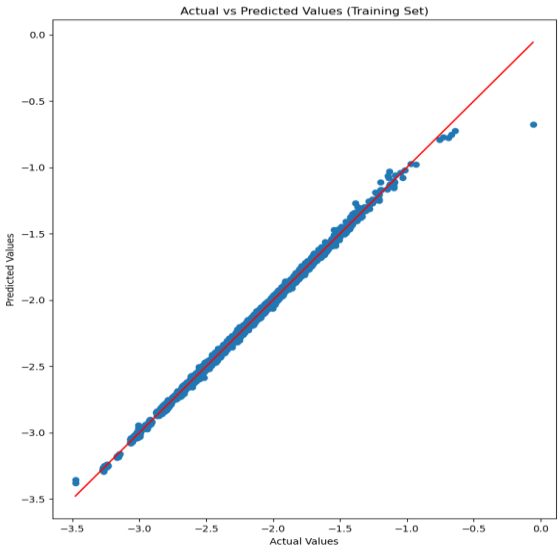


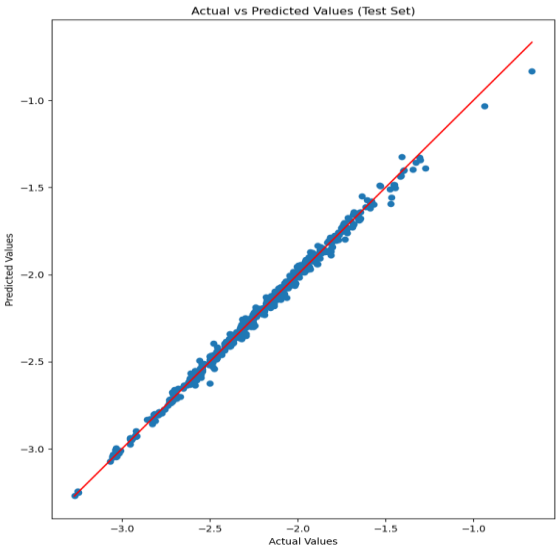
..Actual vs predicted output for training
(4000 Data in Training)

Actual vs predicted output for testing
(1000 Data in Testing)

For the property -3

HistGRadientBoosting Regressor



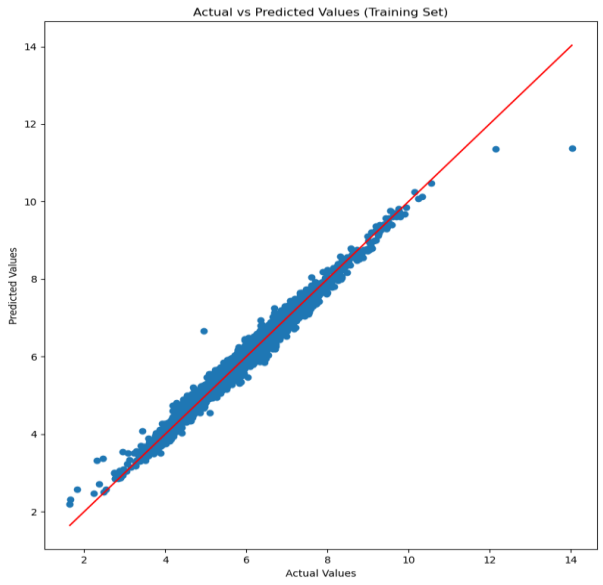


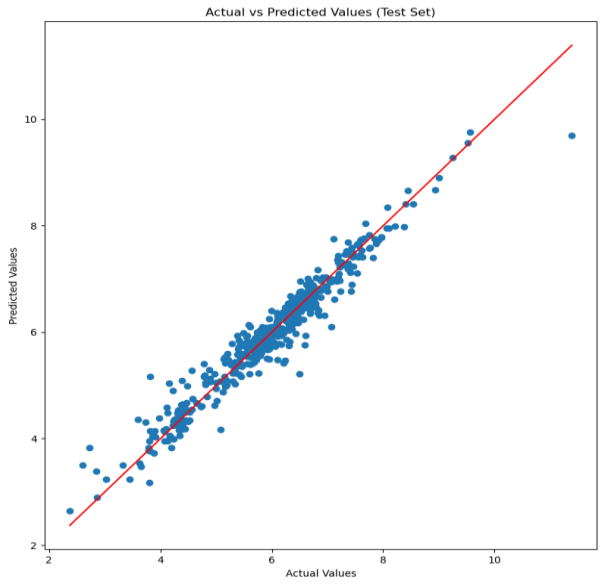
Actual vs predicted output for training
(4000Data in Training)

Actual vs predicted output for testing.
(1000 Data in Testing)

For the property -4

HistGRadientBoosting Regressor

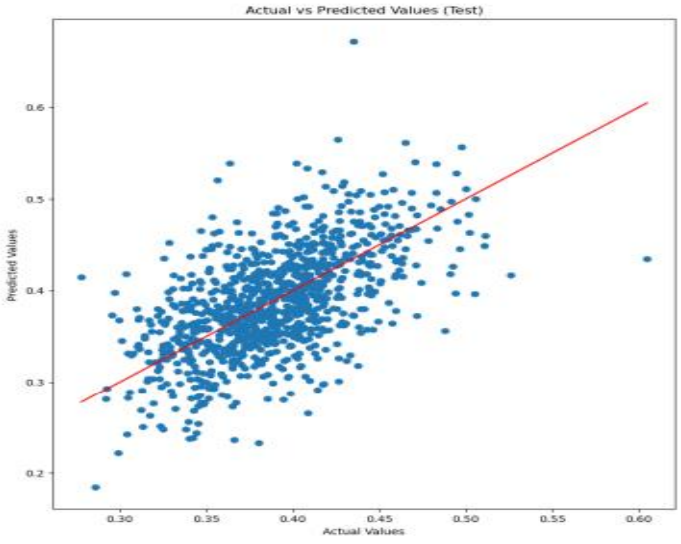


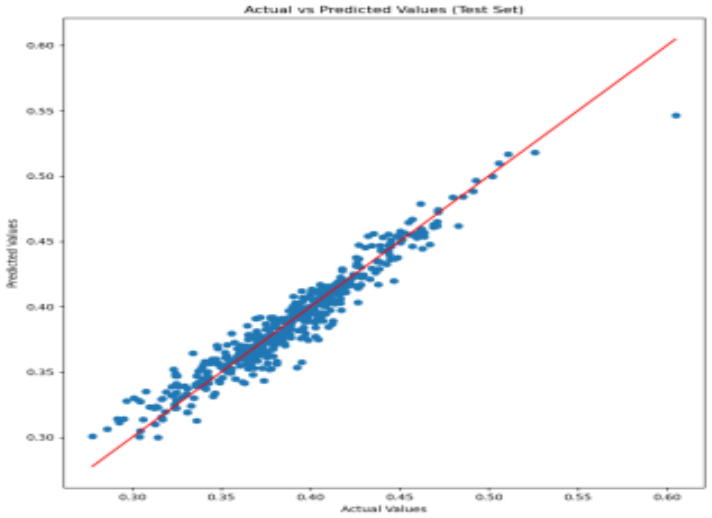


.Actual vs predicted output for training
(4000 Data in Training)

Actual vs predicted output for testing
(1000 Data in Testing)

Therefore, we find that HistGradientBoosting Regressor Model is best fit model for the property-2 than Neural Network.





Neural Network

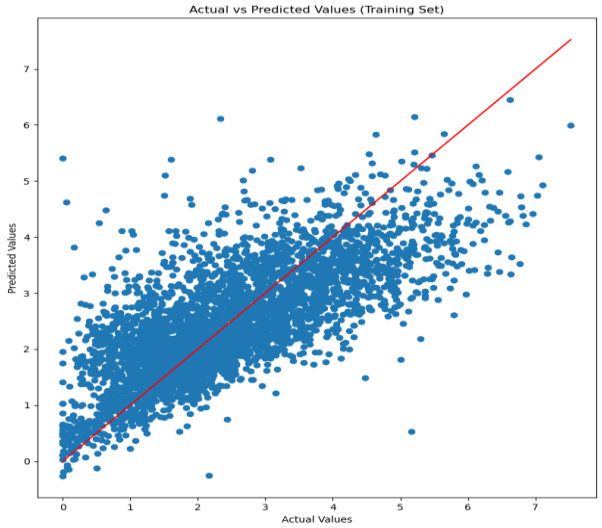
HistGradientBoosting Regressor

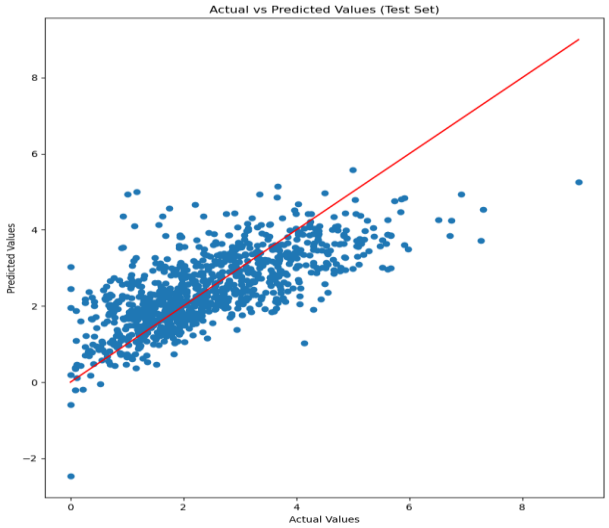
But the property-1 is not well fitted for both Neural Network and HistGradientBoosting Regressor.

Lets see How Multiple Linear Regression work on our Dataset:-

Property-1

Multiple Linear Regression -(Training: Testing) = 80:20



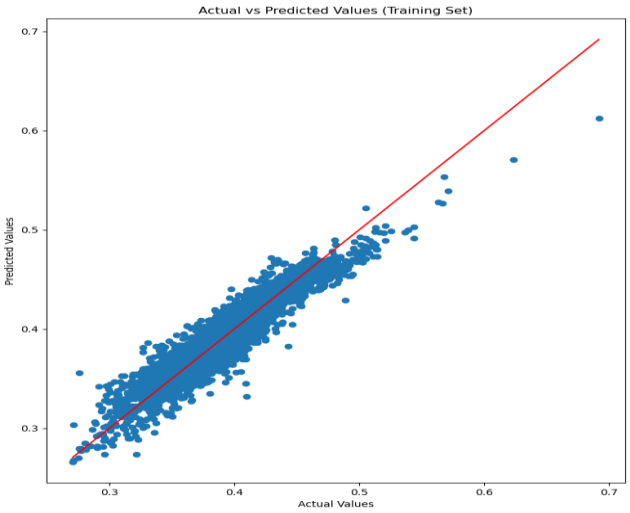


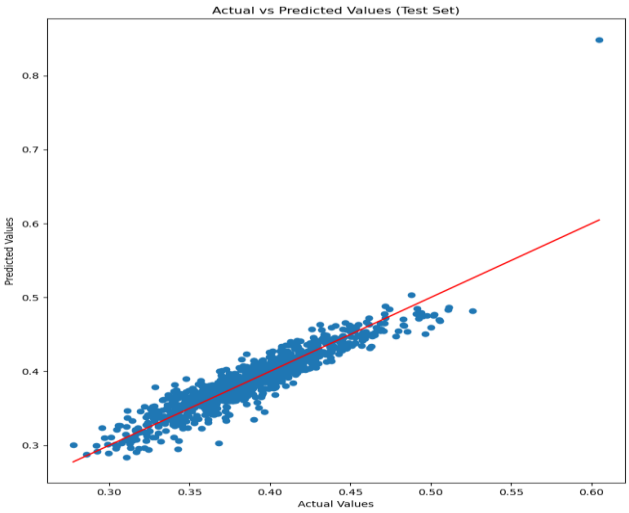
Actual vs predicted output for Training (4000)

Actual vs predicted output for Testing (1000)

Property-2

Multiple Linear Regression -(Training :Testing)=80:20



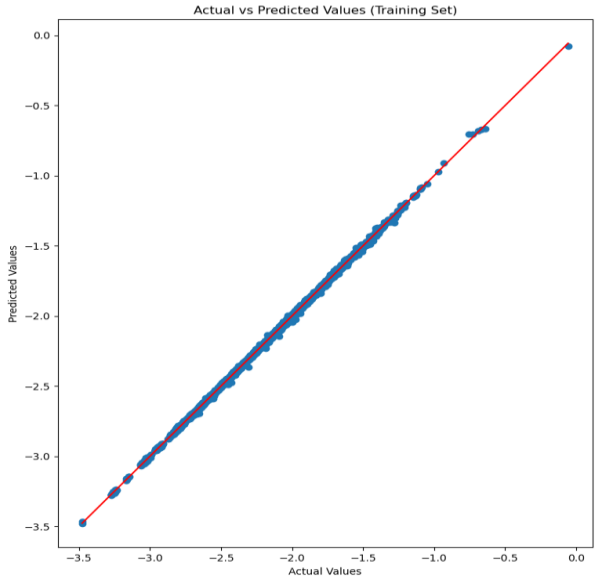


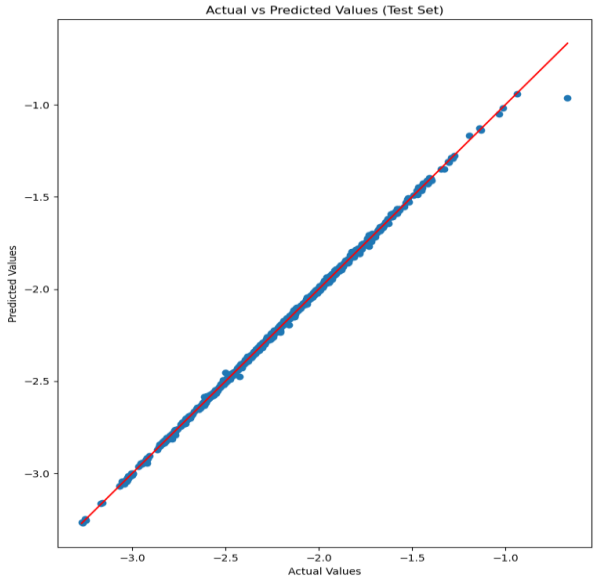
Actual vs predicted output for Training (4000)

Actual vs predicted output for Testing (1000)

Property-3

Multiple Linear Regression -(Training: Testing) = 80:20



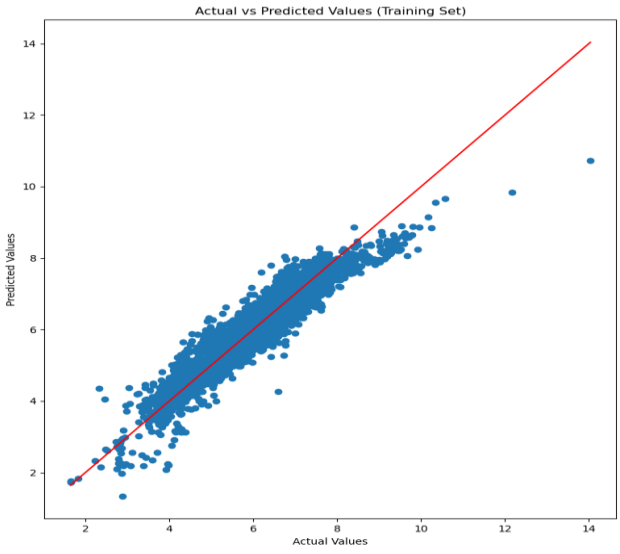


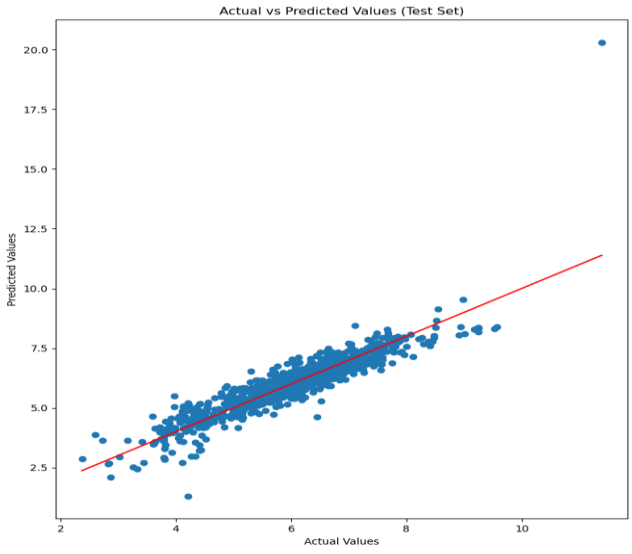
Actual vs predicted output for Training (4000)

Actual vs predicted output for Testing (1000)

Property-4

Multiple Linear Regression -(Training: Testing) = 80:20





Actual vs predicted output for Training(4000)

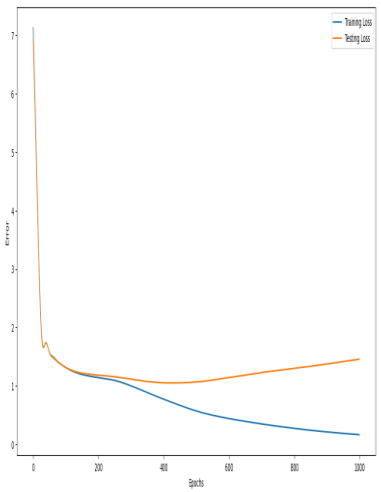
Actual vs predicted output for Testing(1000)

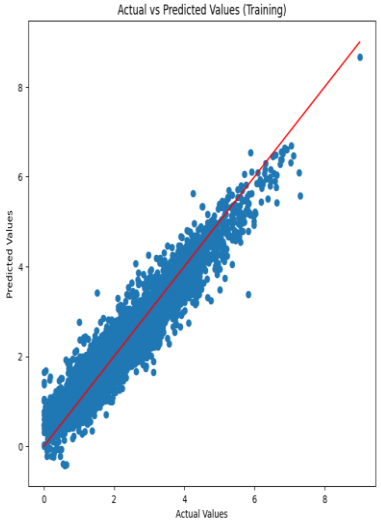
problem is not resolved.
So lets do the Neural Network model by
1)Changing The Molecular Descriptor.(Morgan Descriptor)

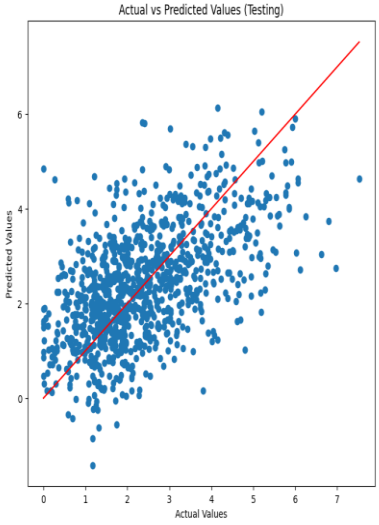
Nural Network

Learning Rate	No of Hidden Layer	No of Input Nodes	No of Hidden Nodes	Activation Functiom	Training Dataset	Testing Data set	Epochs
0.001	1	512	100	relu	80%	20%	1000

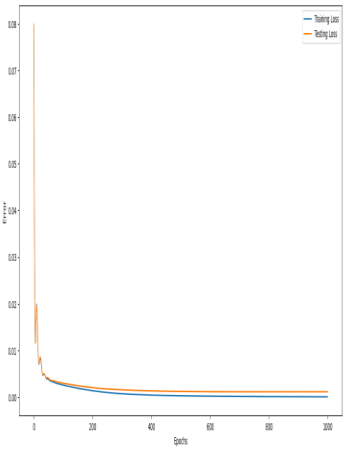
Property -1

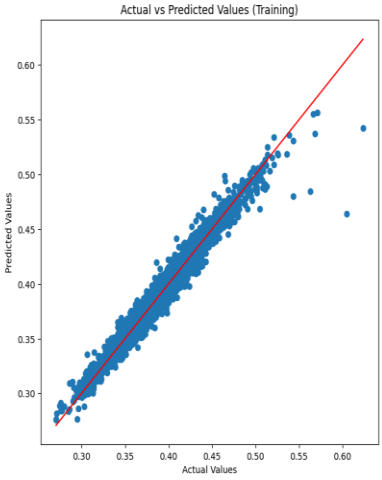


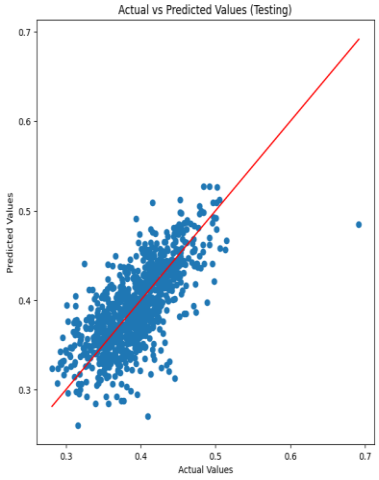




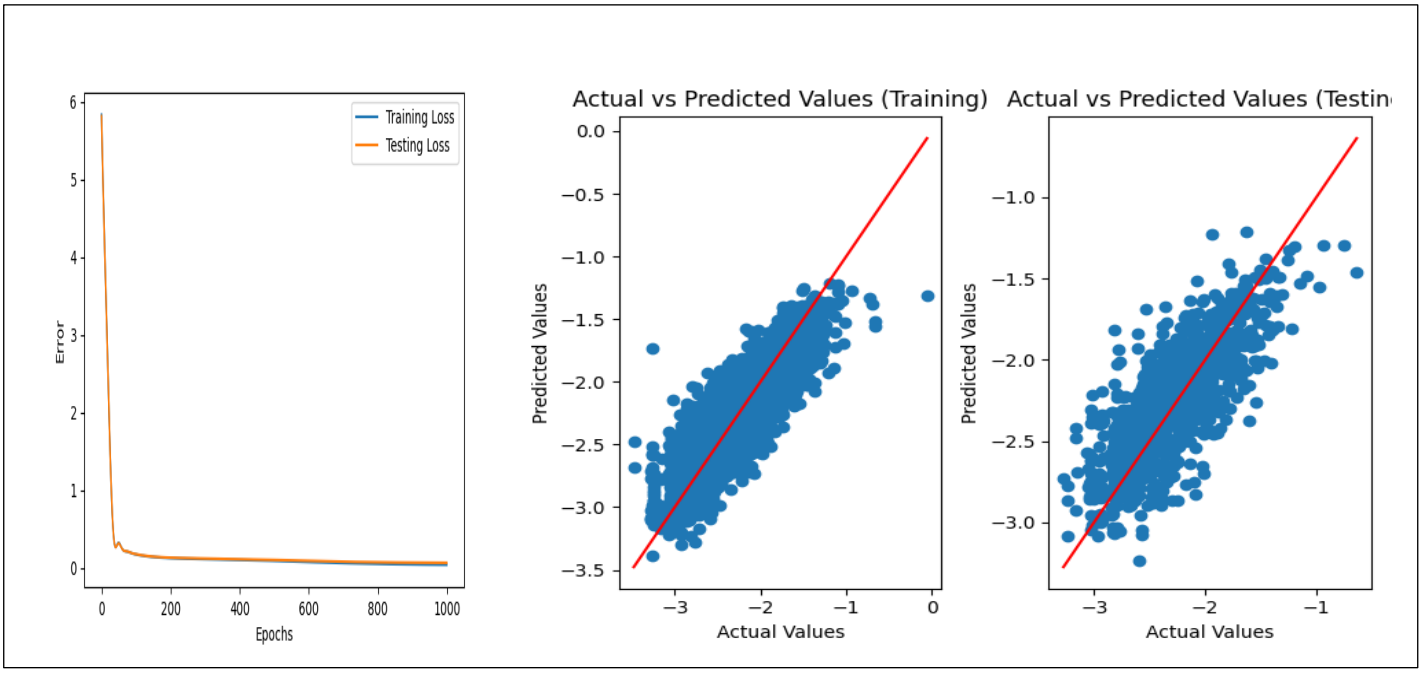
Property-2



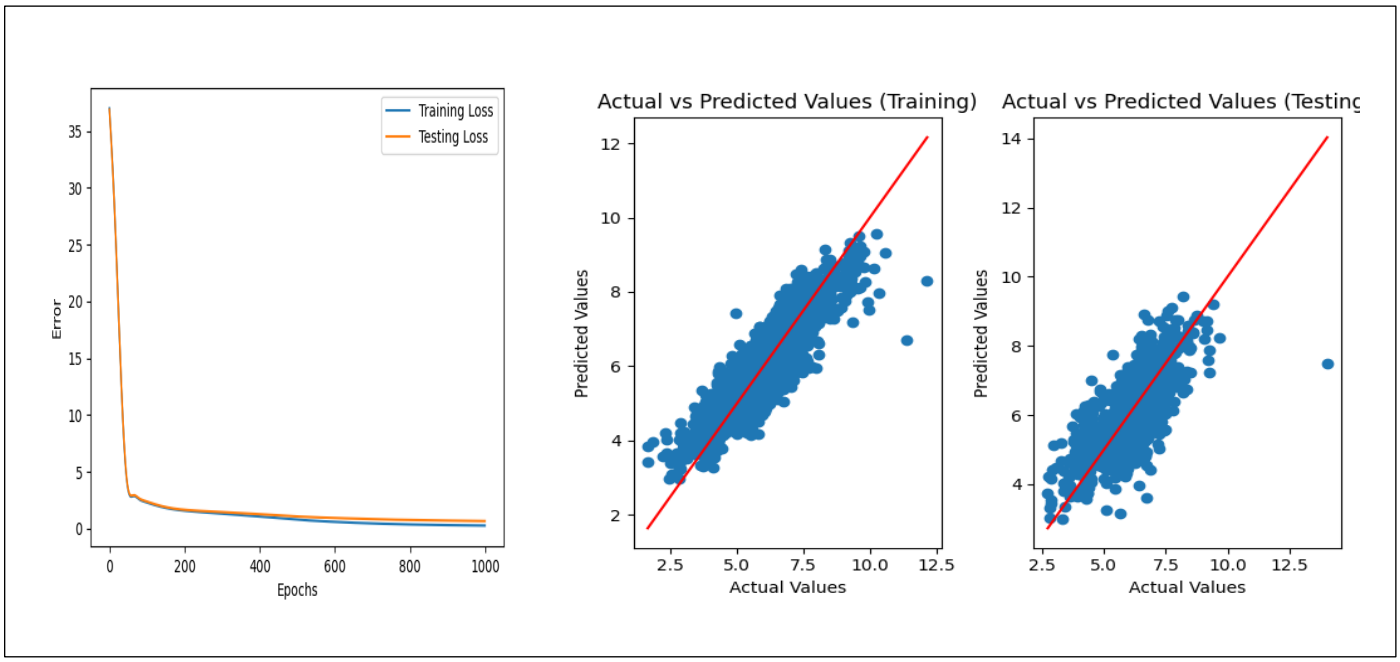




Property-3



Property-4

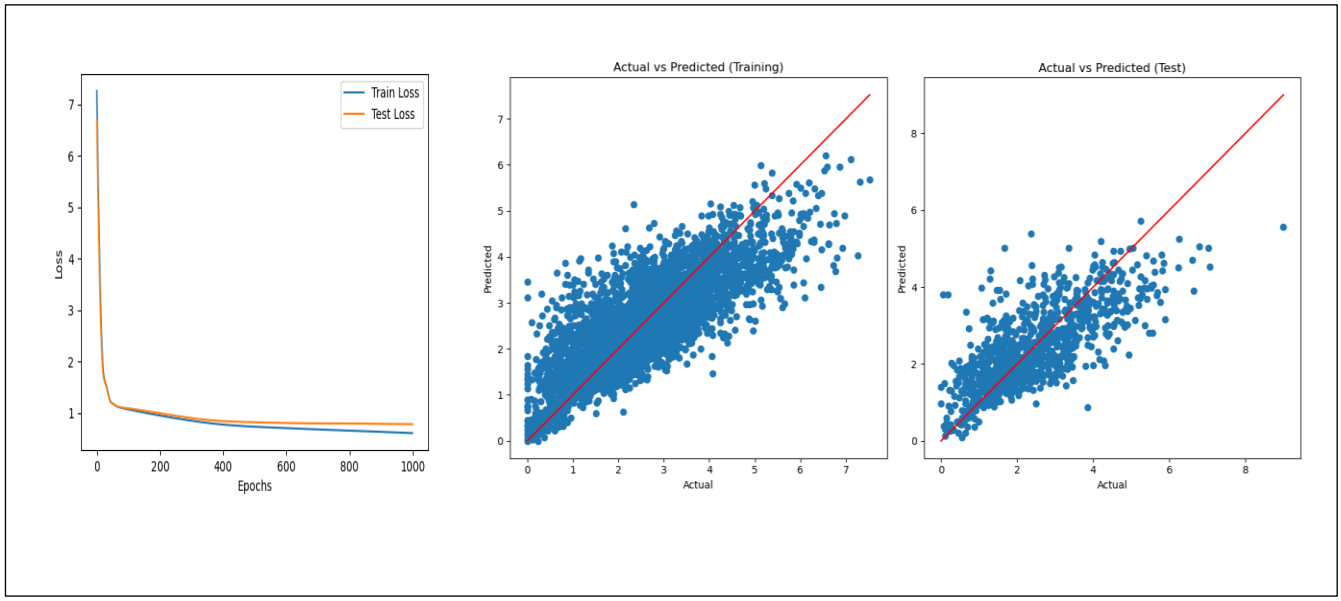


1)Changing The Descriptor.(EState Fingerprint)

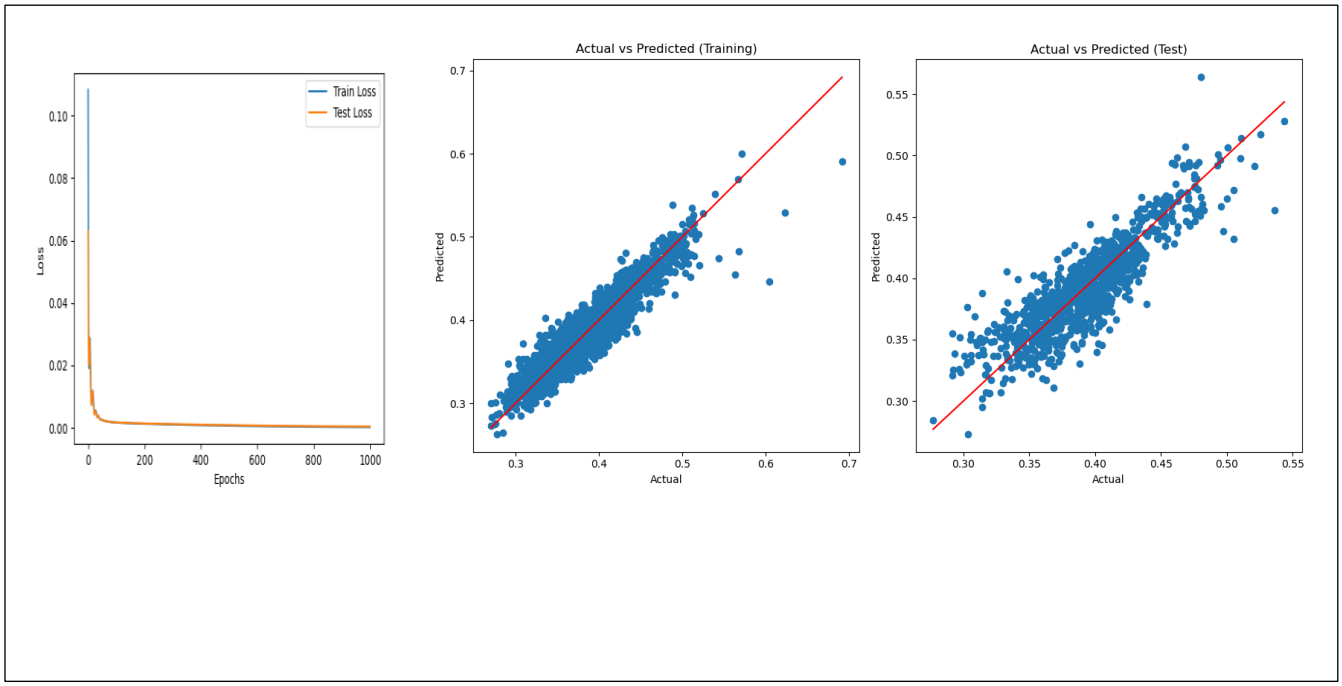
Neural Network

Learning Rate	No of Hidden Layer	No of Input Nodes	No of Hidden Nodes	Activation Functiom	Training Dataset	Testing Data set	Epochs
0.001	1	80	100	relu	80%	20%	1000

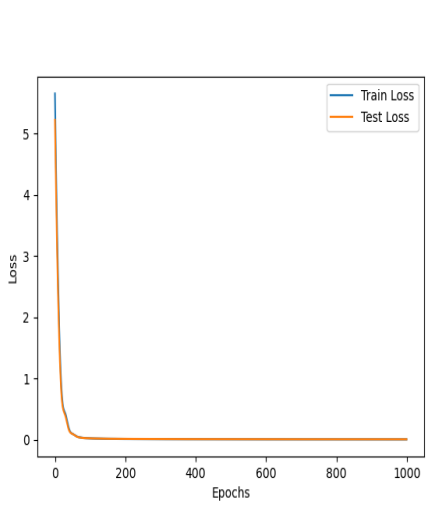
Property-1

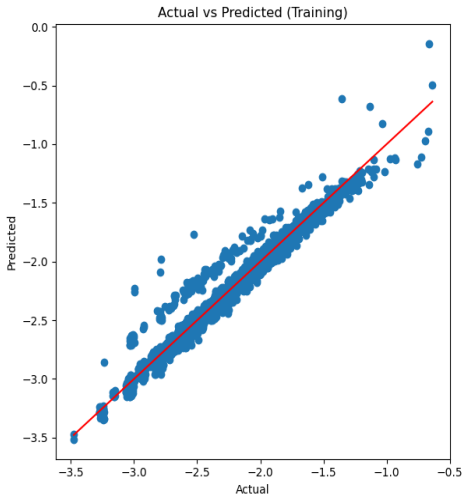


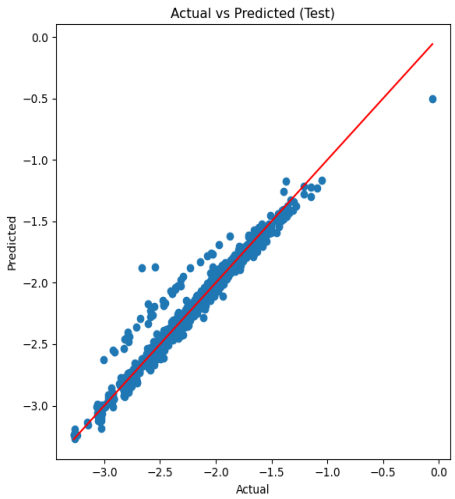
Property-2



Proprety-3







Property-4