Business Report

SMDM Project Business Report DSBA

Capstone project – Supply Chain Management

Sanjay Srinivasan

PGP-DSBA Online

JULY' 21 Batch

Date: 12-06-2022

INDEX

List Of Figures

Model building and interpretation.

a) Build various models (You can choose to build models for either or all of descriptive, predictive or prescriptive purposes).

A machine learning model is **built by learning and generalizing from training data, and make predictions for the business problem.**

In regression analysis, model building **is the process of developing a probabilistic model that best describes the relationship between the dependent and independent variables**.

Steps involved in model building:

1) Problem Statement
2) Data Collection.
3) Data Cleaning.
4) Exploratory Data Analysis.
5) Model Development
6) Train the Model.
7) Test the Models.
8) Applying Models
9) Inferences, Recommendations and business insights based on the model

The four main analytical models organisations can deploy are:

1) Descriptive
2) Diagnostic
3) Predictive
4) Prescriptive

Descriptive **- *It generally uses historical data*** from a single internal source to pinpoint when an event occurred. ***Descriptive analytics are often displayed on dashboards and in reports.***

Diagnostic **-** A diagnostic model is a framework for **identifying, analysing and interpreting data in a given context to identify possible needs.** An effective diagnostic model **allows identifying reliable data to help clients better understand their company's strengths, deficiencies, and opportunities for improvement, to later articulate a targeted intervention and measurement strategy.**

Predictive - Predictive modelling **is a mathematical process used to predict future events or outcomes by analyzing patterns in a given set of input data.**

Prescriptive **-** Prescriptive analytics **utilizes similar modelling structures to predict outcomes and then utilizes a combination of machine learning, business rules, artificial intelligence, and algorithms to simulate various approaches to these numerous outcomes.**

Model building is performed with the following Model:

- Linear Regression
- Lasso Regression (L1 Regularization Model)
- Ridge Regression (L2 Regularization Model)
- Support Vector Regression Model
- Huber Regression
- Random Forest Regressor model
- Artificial Neural Network Regressor Model
- Ada Boost Regressor Model

To find Parameters for Regression model:

RMSE (Root Mean Squared Error) - The root-mean-square error (RMSE) is a **frequently used measure of the differences between values (sample or population values) predicted by a model or an estimator and the values observed.** The best model will have the value in the **range 0.2 - 0.5**

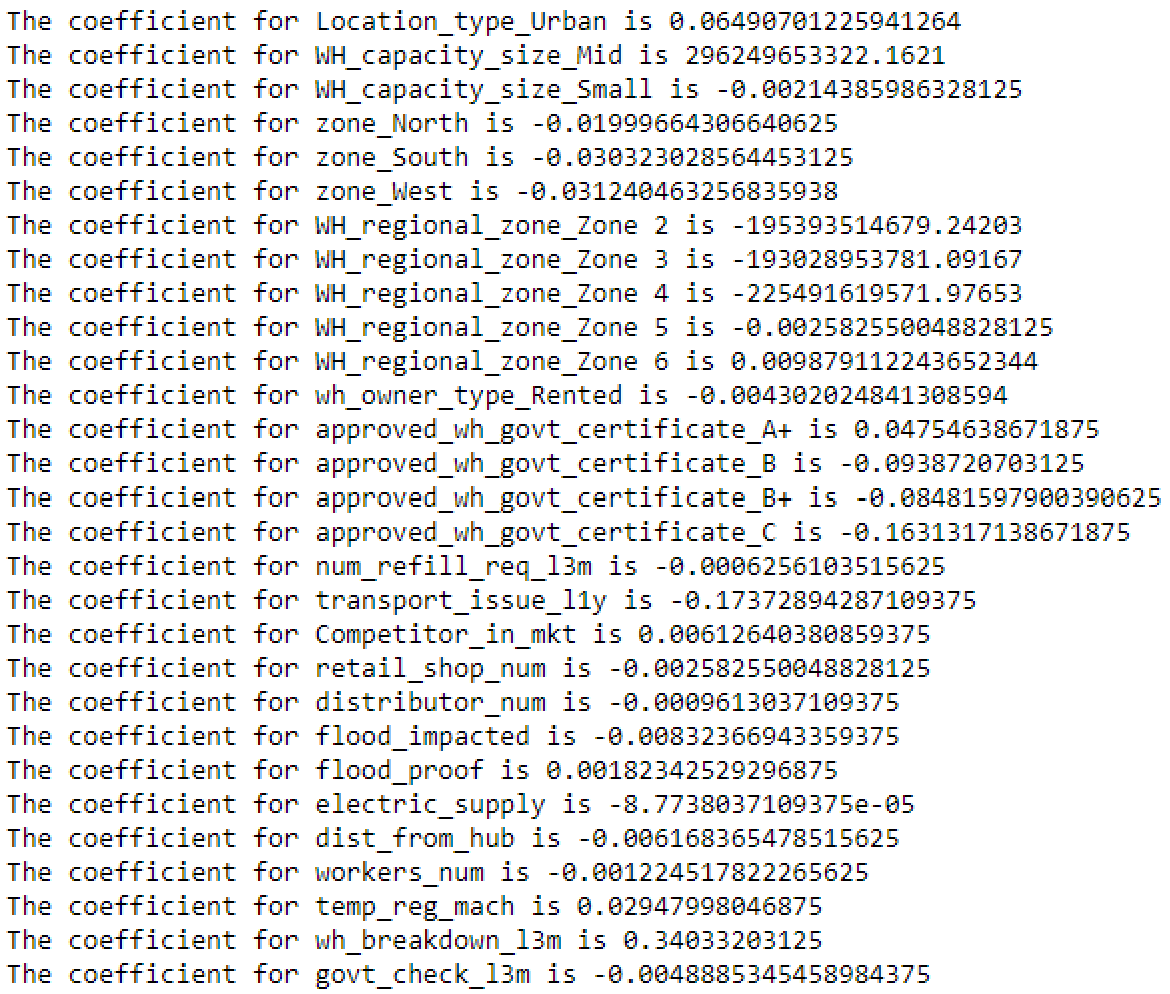
MSE (Mean Squared Error) - In Statistics, Mean Square Error (MSE) is defined as **Mean or Average of the square of the difference between actual and estimated values.** The best model will have the value lesser value (0 is the best value for the model).

MAE (Mean Absolute Error) - The **MAE measures the average magnitude of the errors in a set of forecasts, without considering their direction.**

R-Squared value **-** R-squared is a **statistical measure that represents the goodness of fit of a regression model.** The ideal value for r-square is 1. The closer the value of r-square to 1, the better is the model fitted. The best model will have the value higher value(1 is the best value for the model).

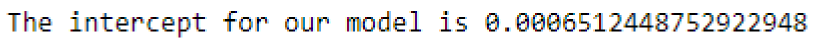
Linear Regression Model:

Linear regression analysis is used to **predict the value of a variable based on the value of another variable**. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.



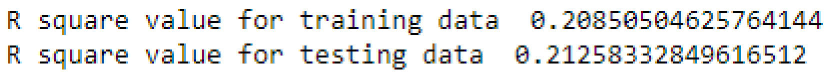
```
The coefficient for Location_type_Urban is 0.06490701225941264
The coefficient for WH_capacity_size_Mid is 296249653322.1621
The coefficient for WH_capacity_size_Small is -0.00214385986328125
The coefficient for zone_North is -0.01999664306640625
The coefficient for zone_South is -0.030323028564453125
The coefficient for zone_West is -0.031240463256835938
The coefficient for WH_regional_zone_Zone 2 is -195393514679.24203
The coefficient for WH_regional_zone_Zone 3 is -193028953781.09167
The coefficient for WH_regional_zone_Zone 4 is -225491619571.97653
The coefficient for WH_regional_zone_Zone 5 is -0.002582550048828125
The coefficient for WH_regional_zone_Zone 6 is 0.009879112243652344
The coefficient for wh_owner_type_Rented is -0.004302024841308594
The coefficient for approved_wh_govt_certificate_A+ is 0.04754638671875
The coefficient for approved_wh_govt_certificate_B is -0.0938720703125
The coefficient for approved_wh_govt_certificate_B+ is -0.08481597900390625
The coefficient for approved_wh_govt_certificate_C is -0.1631317138671875
The coefficient for num_refill_req_l3m is -0.0006256103515625
The coefficient for transport_issue_l1y is -0.17372894287109375
The coefficient for Competitor_in_mkt is 0.00612640380859375
The coefficient for retail_shop_num is -0.002582550048828125
The coefficient for distributor_num is -0.0009613037109375
The coefficient for flood_impacted is -0.00832366943359375
The coefficient for flood_proof is 0.0018234252929688
The coefficient for electric_supply is -8.7738037109375e-05
The coefficient for dist_from_hub is -0.006168365478515625
The coefficient for workers_num is -0.001224517822265625
The coefficient for temp_reg_mach is 0.02947998046875
The coefficient for wh_breakdown_l3m is 0.34033203125
The coefficient for govt_check_l3m is -0.004888534545898437
```

Fig 1.1 Coefficient for Linear Regression model



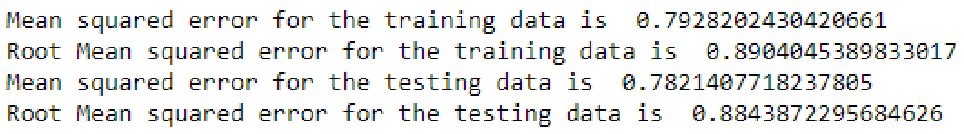
```
The intercept for our model is 0.0006512448752922948
```

Fig 1.2 Intercept for Linear Regression model



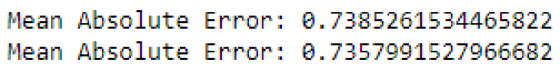
```
R square value for training data  0.20850504625764144
R square value for testing data  0.21258332849616512
```

Fig 1.3 R-Square for Linear Regression model



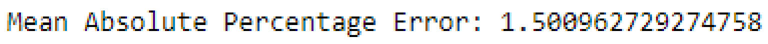
```
Mean squared error for the training data is  0.7928202430420661
Root Mean squared error for the training data is  0.8904045389833017
Mean squared error for the testing data is  0.7821407718237805
Root Mean squared error for the testing data is  0.8843872295684626
```

Fig 1.4 Mean Squared Error and Root Mean Squared Error for Linear Regression model



```
Mean Absolute Error: 0.7385261534465822
Mean Absolute Error: 0.7357991527966682
```

Fig 1.5 Mean Absolute Error for Linear Regression model



```
Mean Absolute Percentage Error: 1.500962729274758
```

Fig 1.6 Mean Absolute Percentage Error for Linear Regression model

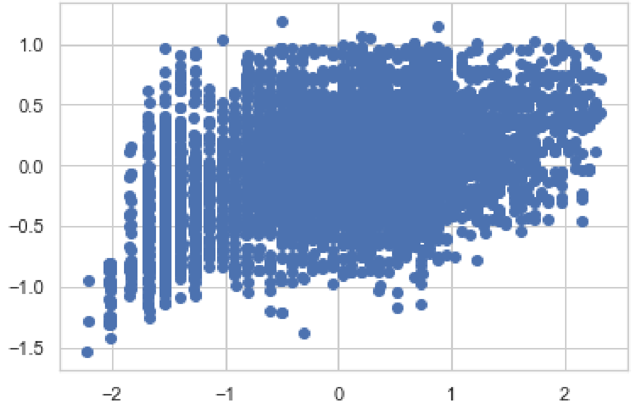
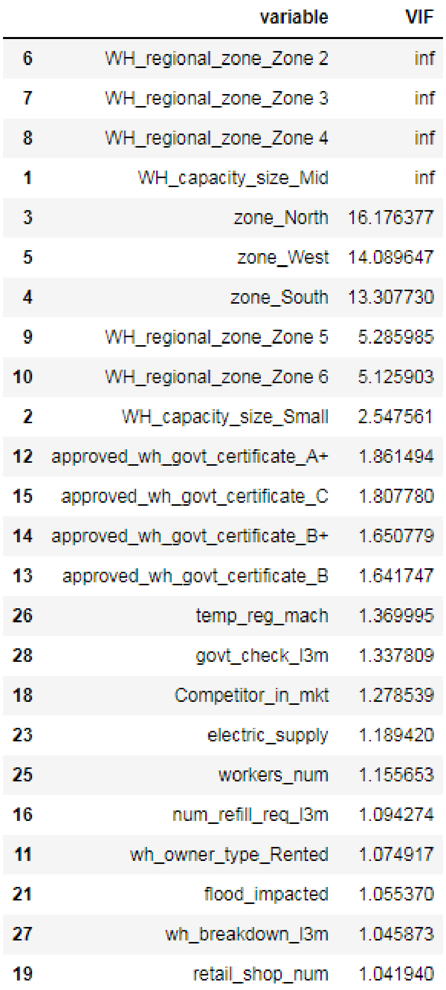


Fig 1.7 Scatter plot for Linear Regression model



	variable	VIF
6	WH_regional_zone_Zone 2	inf
7	WH_regional_zone_Zone 3	inf
8	WH_regional_zone_Zone 4	inf
1	WH_capacity_size_Mid	inf
3	zone_North	16.176377
5	zone_West	14.089647
4	zone_South	13.307730
9	WH_regional_zone_Zone 5	5.285985
10	WH_regional_zone_Zone 6	5.125903
2	WH_capacity_size_Small	2.547561
12	approved_wh_govt_certificate_A+	1.861494
15	approved_wh_govt_certificate_C	1.807780
14	approved_wh_govt_certificate_B+	1.650779
13	approved_wh_govt_certificate_B	1.641747
26	temp_reg_mach	1.369995
28	govt_check_l3m	1.337809
18	Competitor_in_mkt	1.278539
23	electric_supply	1.189420
25	workers_num	1.155653
16	num_refill_req_l3m	1.094274
11	wh_owner_type_Rented	1.074917
21	flood_impacted	1.055370
27	wh_breakdown_l3m	1.045873
19	retail_shop_num	1.041940

Fig 1.8 Variance Inflation Factor for Linear Regression model

<u>Assumptions of Linear Regression:</u>

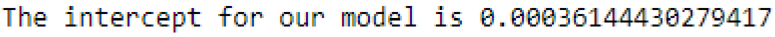
- Linear relationship.
- Multivariate normality.
- No or little multicollinearity.
- No auto-correlation.
- Homoscedasticity.

Lasso Regression Model:

In statistics and machine learning, lasso (least absolute shrinkage and selection operator; also Lasso or LASSO) is a **regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model.**

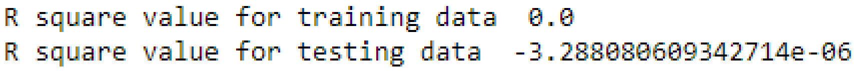
Assumptions of Lasso Regression:

- Linearity - linear regression needs the relationship between the predictor and target variables to be linear
- Independence
- No Heteroskedasticity
- No Multicolinearity.



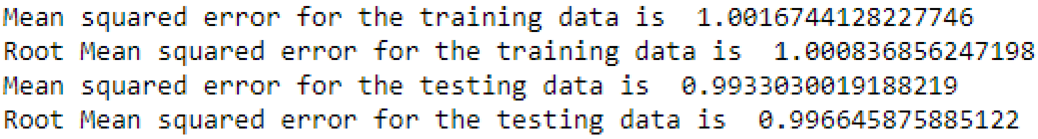
```
The intercept for our model is 0.00036144430279417
```

Fig 1.9 Intercept for Lasso Regression model



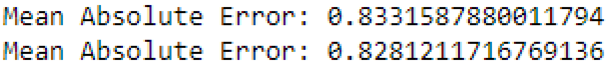
```
R square value for training data  0.0
R square value for testing data  -3.288080609342714e-06
```

Fig 1.10 R-Square for Lasso Regression model



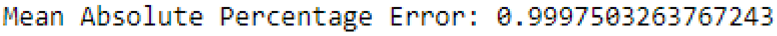
```
Mean squared error for the training data is  1.0016744128227746
Root Mean squared error for the training data is  1.000836856247198
Mean squared error for the testing data is  0.9933030019188219
Root Mean squared error for the testing data is  0.996645875885122
```

Fig 1.11 Mean Squared Error and Root Mean Squared Error for Lasso Regression model



```
Mean Absolute Error: 0.8331587880011794
Mean Absolute Error: 0.8281211716769136
```

Fig 1.12 Mean Absolute Error for Lasso Regression model



```
Mean Absolute Percentage Error: 0.9997503263767243
```

Fig 1.13 Mean Absolute Percentage Error for Lasso Regression model

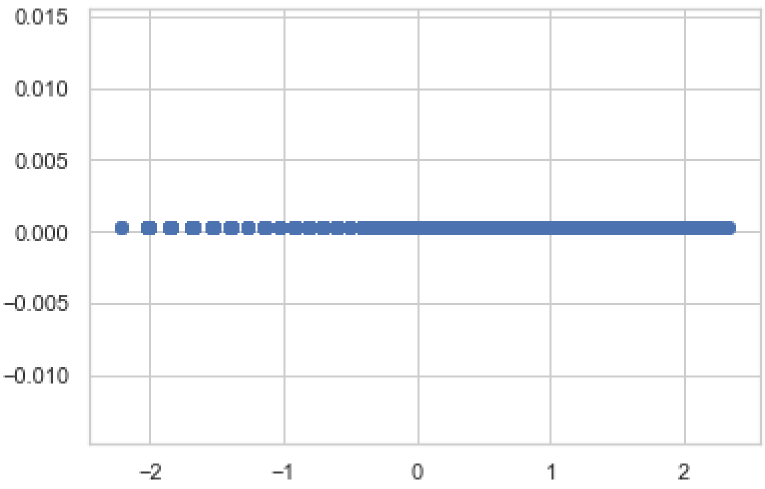


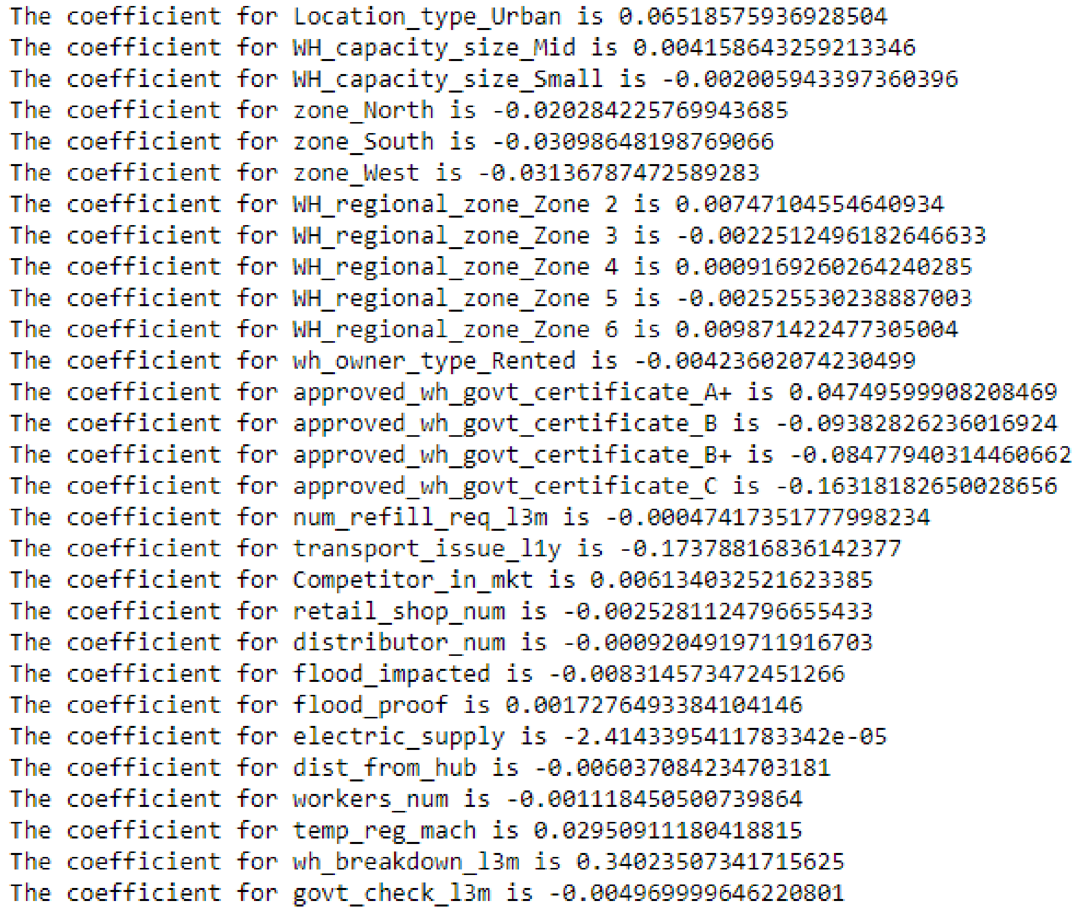
Fig 1.14 Scatter plot for Lasso Regression model

Ridge Regression:

Ridge regression is a **method of estimating the coefficients of multiple-regression models in scenarios where linearly independent variables are highly correlated**.

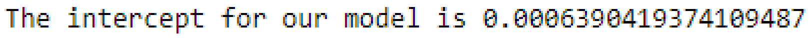
Assumptions of Ridge Regression:

- Linearity
- Constant Variance
- Independence



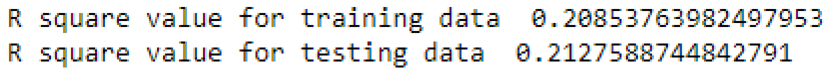
```
The coefficient for Location_type_Urban is 0.06518575936928504
The coefficient for WH_capacity_size_Mid is 0.004158643259213346
The coefficient for WH_capacity_size_Small is -0.002005943397360396
The coefficient for zone_North is -0.020284225769943685
The coefficient for zone_South is -0.03098648198769066
The coefficient for zone_West is -0.03136787472589283
The coefficient for WH_regional_zone_Zone 2 is 0.00747104554640934
The coefficient for WH_regional_zone_Zone 3 is -0.0022512496182646633
The coefficient for WH_regional_zone_Zone 4 is 0.0009169260264240285
The coefficient for WH_regional_zone_Zone 5 is -0.002525530238887003
The coefficient for WH_regional_zone_Zone 6 is 0.009871422477305004
The coefficient for wh_owner_type_Rented is -0.00423602074230499
The coefficient for approved_wh_govt_certificate_A+ is 0.04749599908208469
The coefficient for approved_wh_govt_certificate_B is -0.09382826236016924
The coefficient for approved_wh_govt_certificate_B+ is -0.08477940314460662
The coefficient for approved_wh_govt_certificate_C is -0.16318182650028656
The coefficient for num_refill_req_l3m is -0.00047417351777998234
The coefficient for transport_issue_l1y is -0.1737881683614237
The coefficient for Competitor_in_mkt is 0.006134032521623385
The coefficient for retail_shop_num is -0.0025281124796655433
The coefficient for distributor_num is -0.0009204919711916703
The coefficient for flood_impacted is -0.008314573472451266
The coefficient for flood_proof is 0.0017276493384104146
The coefficient for electric_supply is -2.414339541178342e-05
The coefficient for dist_from_hub is -0.006037084234703181
The coefficient for workers_num is -0.001118450500739864
The coefficient for temp_reg_mach is 0.02950911180418815
The coefficient for wh_breakdown_l3m is 0.34023507341715625
The coefficient for govt_check_l3m is -0.004969999646220801
```

Fig 1.15 Coefficient of Ridge Regression model



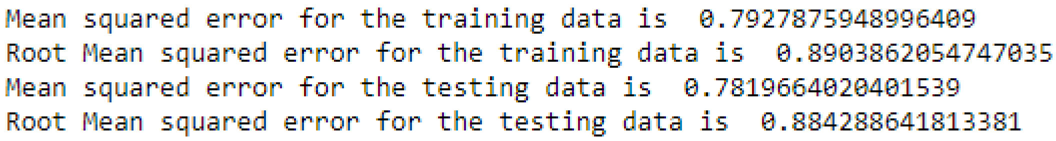
```
The intercept for our model is 0.0006390419374109487
```

Fig 1.16 Intercept of Ridge Regression model



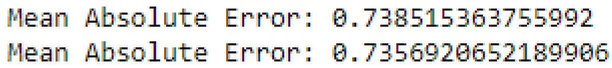
```
R square value for training data  0.20853763982497953
R square value for testing data  0.2127588744842791
```

Fig 1.17 R-Square of Ridge Regression model



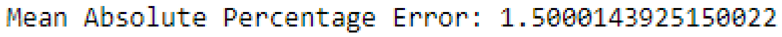
```
Mean squared error for the training data is  0.7927875948996409
Root Mean squared error for the training data is  0.8903862054747035
Mean squared error for the testing data is  0.781966402401539
Root Mean squared error for the testing data is  0.884288641813381
```

Fig 1.18 Mean Squared Error and Root Mean Squared Error for Ridge Regression model



```
Mean Absolute Error: 0.738515363755992
Mean Absolute Error: 0.7356920652189906
```

Fig 1.19 Mean Absolute Error for Ridge Regression model



```
Mean Absolute Percentage Error: 1.5000143925150022
```

Fig 1.20 Mean Absolute Percentage Error for Ridge Regression model

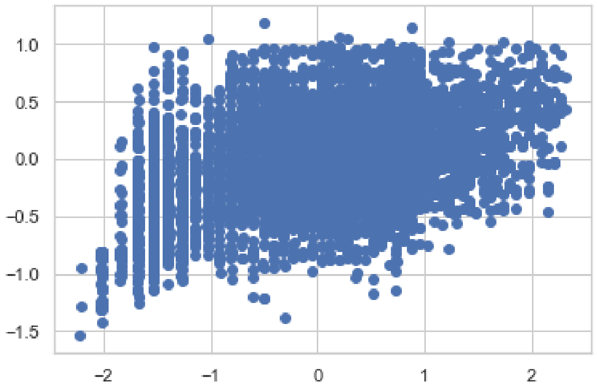
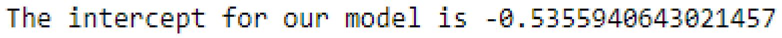


Fig 1.21 Scatterplot for Ridge Regression model

Support Vector Regression Model:

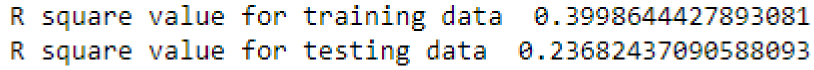
Support Vector Regression is a **supervised learning algorithm that is used to predict discrete values.** Support Vector Regression uses the same principle as the SVMs. The basic idea behind SVR is to find the best fit line. In SVR, the best fit line is the hyper plane that has the maximum number of points.

SVR **gives us the flexibility to define how much error is acceptable in our model and will find an appropriate line (or hyper plane in higher dimensions) to fit the data**.



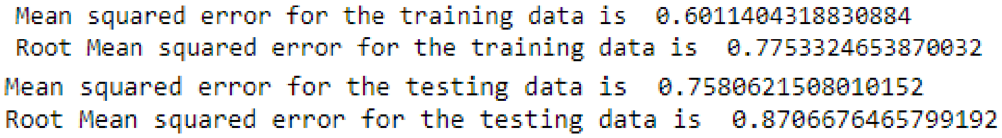
```
The intercept for our model is -0.5355940643021457
```

Fig 1.22 Intercept for Support Vector Regression model



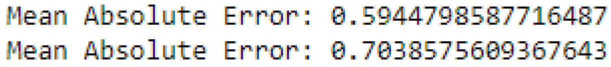
```
R square value for training data  0.3998644427893081
R square value for testing data  0.23682437090588093
```

Fig 1.23 R-Square for Support Vector Regression model



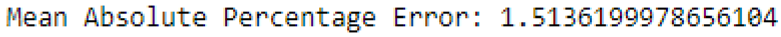
```
Mean squared error for the training data is  0.6011404318830884
Root Mean squared error for the training data is  0.7753324653870032
Mean squared error for the testing data is  0.7580621508010152
Root Mean squared error for the testing data is  0.8706676465799192
```

Fig 1.24 Mean Squared Error and Root Mean Squared Error for Support Vector Regression model



```
Mean Absolute Error: 0.5944798587716487
Mean Absolute Error: 0.7038575609367643
```

Fig 1.25 Mean Absolute Error for Support Vector Regression model



```
Mean Absolute Percentage Error: 1.5136199978656104
```

Fig 1.26 Mean Absolute Percentage Error for Support Vector Regression model

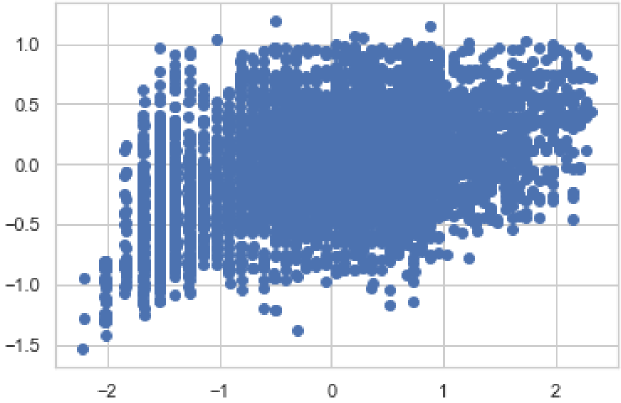


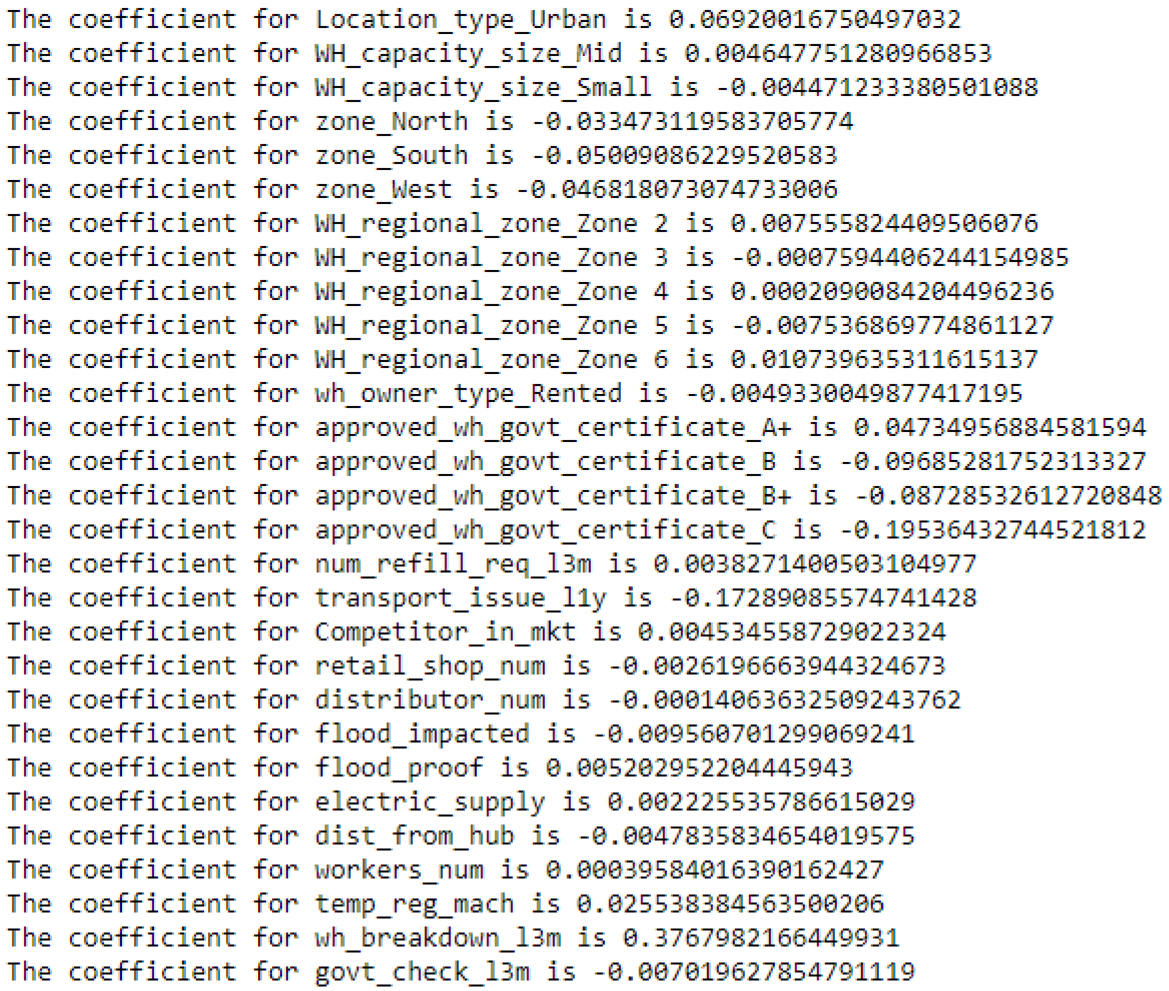
Fig 1.27 Scatterplot for Support Vector Regression model

Huber Regressor Model:

The Huber Regressor optimizes the squared loss for the samples where |(y - X'w) / sigma| < epsilon and the absolute loss for the samples where |(y - X'w) / sigma| > epsilon , where w and sigma are parameters to be optimized.

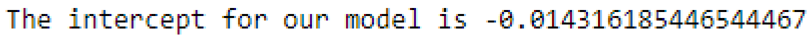
Huber Loss formula:

$$L_\delta(a) = \begin{cases} \frac{1}{2}a^2 & \text{for } |a| \le \delta, \\ \delta(|a| - \frac{1}{2}\delta), & \text{otherwise.} \end{cases}$$



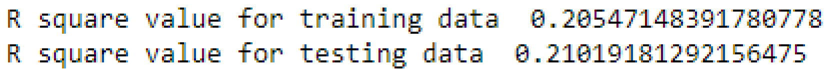
```
The coefficient for Location_type_Urban is 0.06920016750497032
The coefficient for WH_capacity_size_Mid is 0.004647751280966853
The coefficient for WH_capacity_size_Small is -0.004471233380501088
The coefficient for zone_North is -0.033473119583705774
The coefficient for zone_South is -0.05009086229520583
The coefficient for zone_West is -0.046818073074733006
The coefficient for WH_regional_zone_Zone 2 is 0.007555824409506076
The coefficient for WH_regional_zone_Zone 3 is -0.0007594406244154985
The coefficient for WH_regional_zone_Zone 4 is 0.0002090084204496236
The coefficient for WH_regional_zone_Zone 5 is -0.007536869774861127
The coefficient for WH_regional_zone_Zone 6 is 0.010739635311615137
The coefficient for wh_owner_type_Rented is -0.0049330049877417195
The coefficient for approved_wh_govt_certificate_A+ is 0.04734956884581594
The coefficient for approved_wh_govt_certificate_B is -0.09685281752313327
The coefficient for approved_wh_govt_certificate_B+ is -0.08728532612720848
The coefficient for approved_wh_govt_certificate_C is -0.19536432744521812
The coefficient for num_refill_req_l3m is 0.0038271400503104977
The coefficient for transport_issue_l1y is -0.17289085574741428
The coefficient for Competitor_in_mkt is 0.004534558729022324
The coefficient for retail_shop_num is -0.0026196663944324673
The coefficient for distributor_num is -0.00014063632509243762
The coefficient for flood_impacted is -0.009560701299069241
The coefficient for flood_proof is 0.005202952204445943
The coefficient for electric_supply is 0.002225535786615029
The coefficient for dist_from_hub is -0.0047835834654019575
The coefficient for workers_num is 0.00039584016390162427
The coefficient for temp_reg_mach is 0.025538384563500206
The coefficient for wh_breakdown_l3m is 0.3767982166449931
The coefficient for govt_check_l3m is -0.007019627854791119
```

Fig 1.28 Coefficient of Huber Regression model



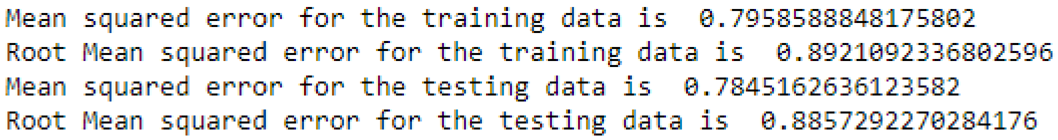
```
The intercept for our model is -0.014316185446544467
```

Fig 1.29 Intercept of Huber Regression model



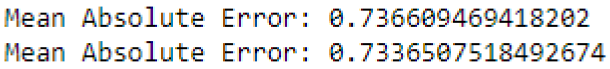
```
R square value for training data  0.20547148391780778
R square value for testing data  0.21019181292156475
```

Fig 1.30 R-Square of Huber Regression model



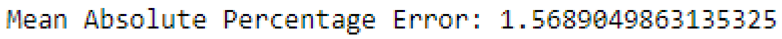
```
Mean squared error for the training data is  0.7958588848175802
Root Mean squared error for the training data is  0.8921092336802596
Mean squared error for the testing data is  0.7845162636123582
Root Mean squared error for the testing data is  0.8857292270284176
```

Fig 1.31 Mean Squared Error and Root Mean Squared Error for Huber Regression model



```
Mean Absolute Error: 0.736609469418202
Mean Absolute Error: 0.7336507518492674
```

Fig 1.32 Mean Absolute Error for Huber Regression model



```
Mean Absolute Percentage Error: 1.5689049863135325
```

Fig 1.33 Mean Absolute Percentage Error for Huber Regression model

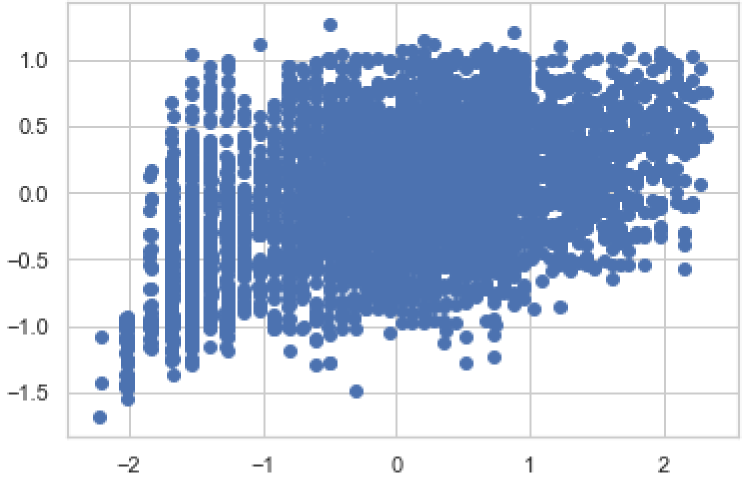
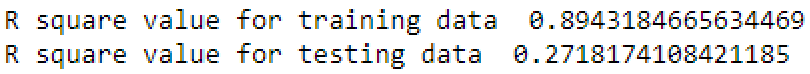


Fig 1.34 Scatterplot for Huber Regression model

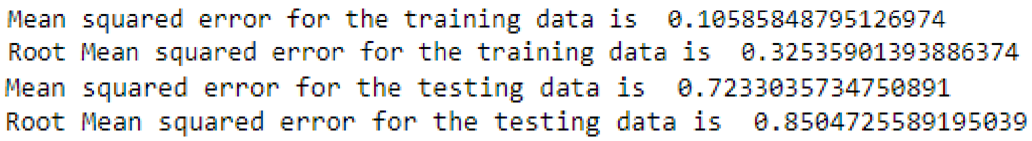
Random Forest Regressor:

A random forest regressor is **a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.**



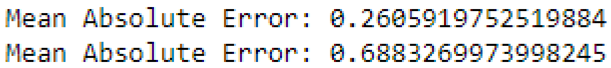
```
R square value for training data  0.8943184665634469
R square value for testing data  0.2718174108421185
```

Fig 1.35 R-Square for Random Forest Regression model



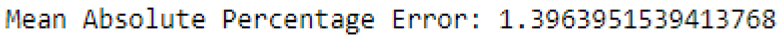
```
Mean squared error for the training data is  0.10585848795126974
Root Mean squared error for the training data is  0.32535901393886374
Mean squared error for the testing data is  0.7233035734750891
Root Mean squared error for the testing data is  0.8504725589195039
```

Fig 1.36 Mean Squared Error and Root Mean Squared Error for Random Forest Regression model



```
Mean Absolute Error: 0.2605919752519884
Mean Absolute Error: 0.6883269973998245
```

Fig 1.37 Mean Absolute Error for Random Forest Regression model



```
Mean Absolute Percentage Error: 1.3963951539413768
```

Fig 1.38 Mean Absolute Percentage Error for Random Forest Regression model

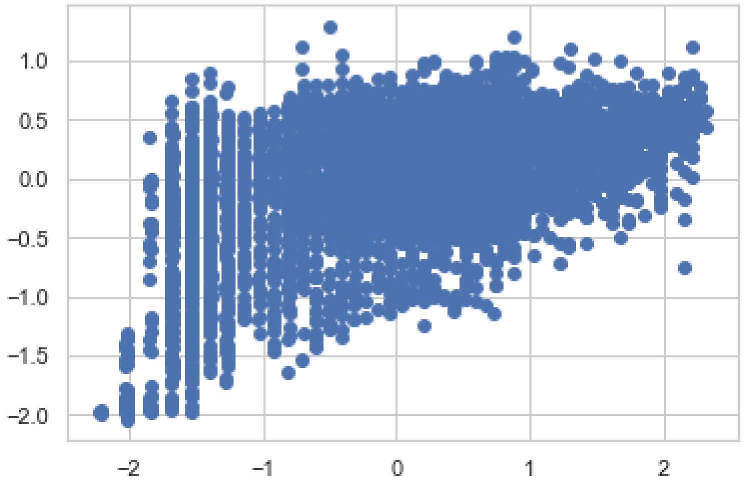
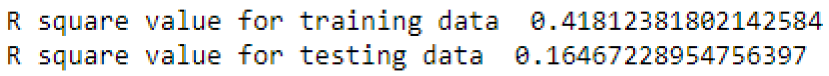


Fig 1.39 Scatterplot for Random Forest Regression model

Artificial Neural Network Regressor:

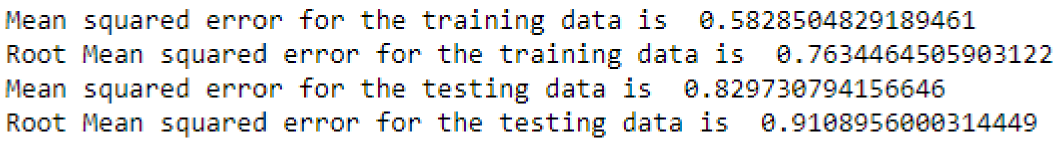
The purpose of using Artificial Neural Networks for Regression over Linear Regression is that **the linear regression can only learn the linear relationship between the features and target and therefore cannot learn the complex non-linear relationship**.

Advantage of ANN regressor over linear regressor is complex non-linear relationship model can be built using ANN regressor model.



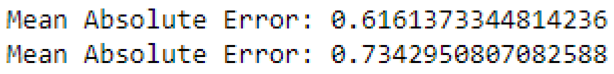
```
R square value for training data  0.41812381802142584
R square value for testing data  0.16467228954756397
```

Fig 1.40 R-Square for ANN Regression model



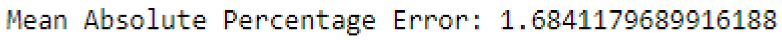
```
Mean squared error for the training data is  0.5828504829189461
Root Mean squared error for the training data is  0.7634464505903122
Mean squared error for the testing data is  0.829730794156646
Root Mean squared error for the testing data is  0.9108956000314449
```

Fig 1.41 Mean Squared Error and Root Mean Squared Error for ANN Regression model



```
Mean Absolute Error: 0.6161373344814236
Mean Absolute Error: 0.7342950807082588
```

Fig 1.42 Mean Absolute Error for ANN Regression model



```
Mean Absolute Percentage Error: 1.6841179689916188
```

Fig 1.43 Mean Absolute Percentage Error for ANN Regression model

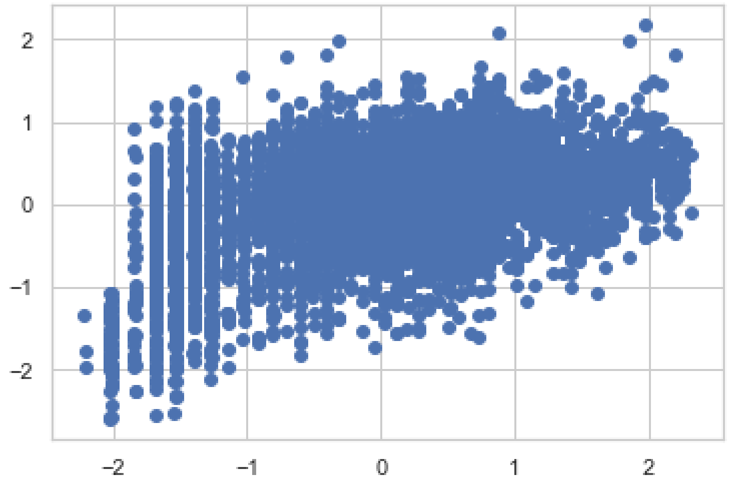
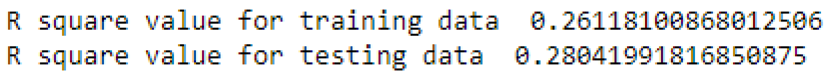


Fig 1.44 Scatterplot for ANN Regression model

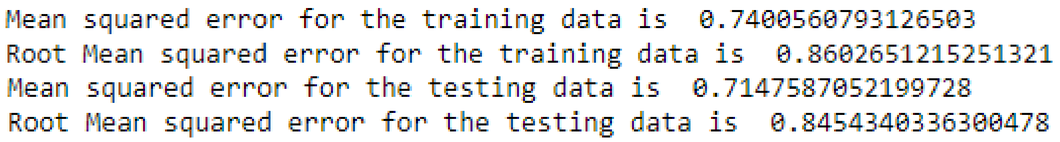
AdaBoost Regressor:

An AdaBoost regressor is a **meta-estimator that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but where the weights of instances are adjusted according to the error of the current prediction.**



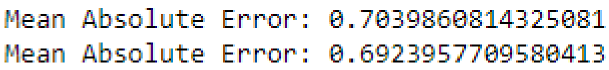
```
R square value for training data  0.26118100868012506
R square value for testing data  0.28041991816850875
```

Fig 1.45 R-Square for Adaboost Regression model



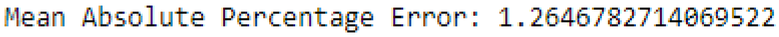
```
Mean squared error for the training data is  0.7400560793126503
Root Mean squared error for the training data is  0.8602651215251321
Mean squared error for the testing data is  0.7147587052199728
Root Mean squared error for the testing data is  0.8454340336300478
```

Fig 1.46 Mean Squared Error and Root Mean Squared Error for Adaboost Regression model



```
Mean Absolute Error: 0.7039860814325081
Mean Absolute Error: 0.6923957709580413
```

Fig 1.47 Mean Absolute Error for Adaboost Regression model



```
Mean Absolute Percentage Error: 1.2646782714069522
```

Fig 1.48 Mean Absolute Percentage Error for Adaboost Regression model

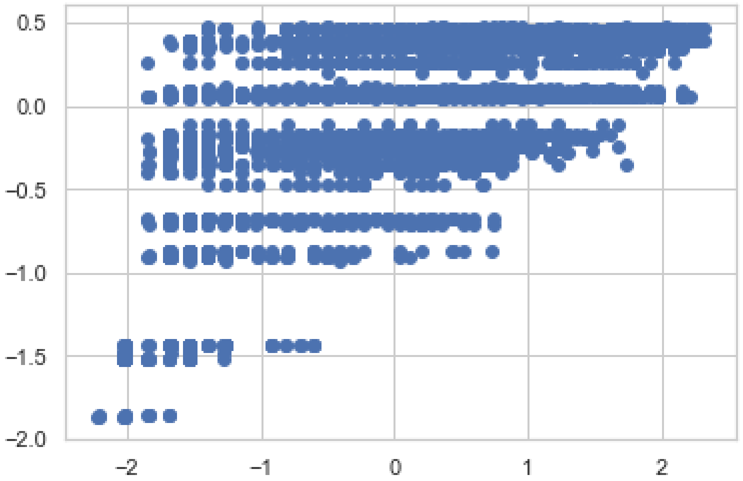
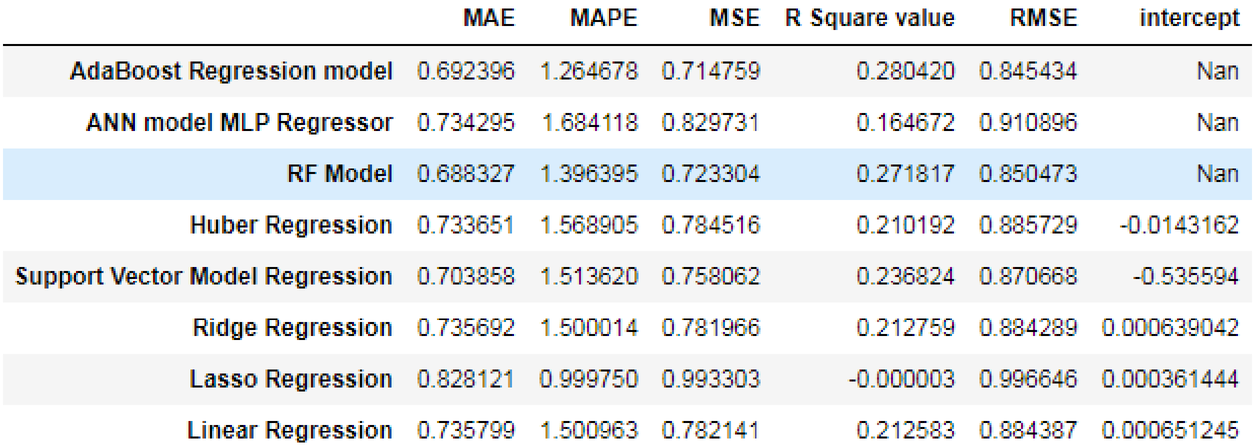


Fig 1.49 Scatterplot for Adaboost Regression model

b.) Test your predictive model against the test set using various appropriate performance metrics.

The performance metrics are predicted against the test set. The performance metrics used are:



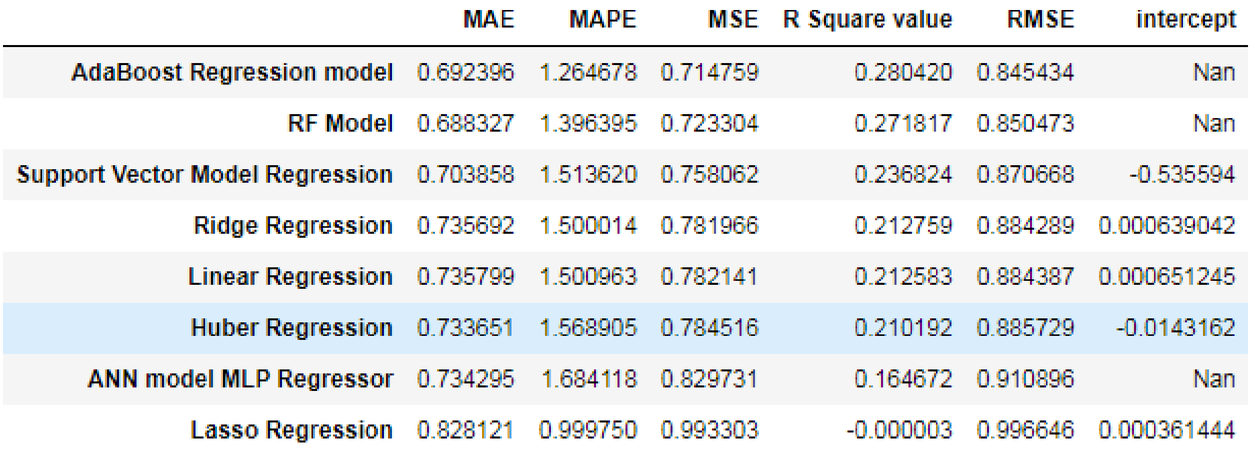
	MAE	MAPE	MSE	R Square value	RMSE	intercept
AdaBoost Regression model	0.692396	1.264678	0.714759	0.280420	0.845434	Nan
ANN model MLP Regressor	0.734295	1.684118	0.829731	0.164672	0.910896	Nan
RF Model	0.688327	1.396395	0.723304	0.271817	0.850473	Nan
Huber Regression	0.733651	1.568905	0.784516	0.210192	0.885729	-0.0143162
Support Vector Model Regression	0.703858	1.513620	0.758062	0.236824	0.870668	-0.535594
Ridge Regression	0.735692	1.500014	0.781966	0.212759	0.884289	0.000639042
Lasso Regression	0.828121	0.999750	0.993303	-0.000003	0.996646	0.000361444
Linear Regression	0.735799	1.500963	0.782141	0.212583	0.884387	0.000651245

Fig 1.50 Performance metrics of all Regression model

The optimum value of Mean Square Error is 0 in the ratio between 0 to 1 (Lower the value better the model).).
The optimum value of R-Square value is 1 in the ratio between 0 to 1 (Higher the value better the model).
The optimum value of Mean Absolute Percentage Error is less than 10%.
The optimum value of Root Mean Square Error (RMSE) value is between 0.2 to 0.5. Lower value of RMSE indicates better fit.

c.)Interpretation of the model(s).

From all the models, we can interpret



	MAE	MAPE	MSE	R Square value	RMSE	intercept
AdaBoost Regression model	0.692396	1.264678	0.714759	0.280420	0.845434	Nan
RF Model	0.688327	1.396395	0.723304	0.271817	0.850473	Nan
Support Vector Model Regression	0.703858	1.513620	0.758062	0.236824	0.870668	-0.535594
Ridge Regression	0.735692	1.500014	0.781966	0.212759	0.884289	0.000639042
Linear Regression	0.735799	1.500963	0.782141	0.212583	0.884387	0.000651245
Huber Regression	0.733651	1.568905	0.784516	0.210192	0.885729	-0.0143162
ANN model MLP Regressor	0.734295	1.684118	0.829731	0.164672	0.910896	Nan
Lasso Regression	0.828121	0.999750	0.993303	-0.000003	0.996646	0.000361444

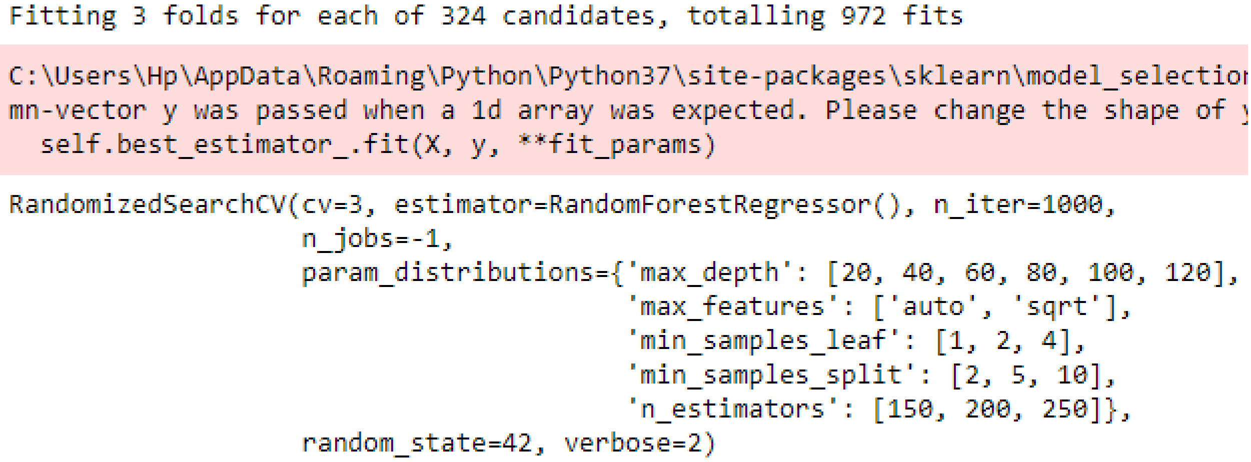
Fig 1.51 Sort Values based on best model performance metrics.

Adaboost Regression model is the best model among the other models from the model building.

Model Tuning

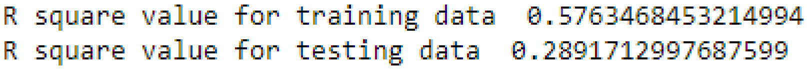
a.Ensemble modelling, wherever applicable.

Random Forest Regressor:



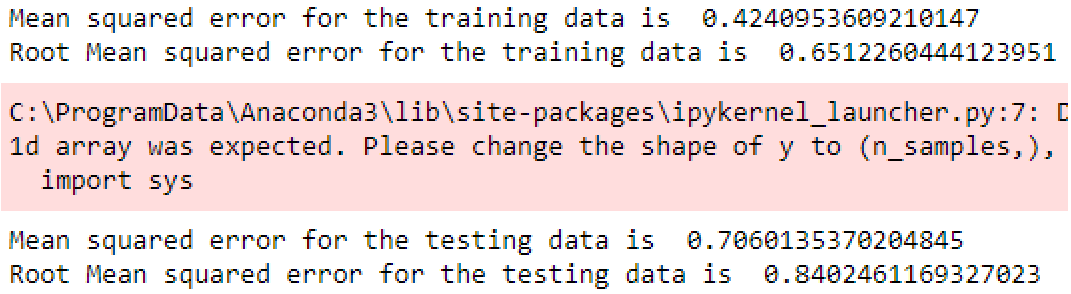
```
Fitting 3 folds for each of 324 candidates, totalling 972 fits

C:\Users\Hp\AppData\Roaming\Python\Python37\site-packages\sklearn\model_selectio
mn-vector y was passed when a 1d array was expected. Please change the shape of y
  self.best_estimator_.fit(X, y, **fit_params)

RandomizedSearchCV(cv=3, estimator=RandomForestRegressor(), n_iter=1000,
                   n_jobs=-1,
                   param_distributions={'max_depth': [20, 40, 60, 80, 100, 120],
                                        'max_features': ['auto', 'sqrt'],
                                        'min_samples_leaf': [1, 2, 4],
                                        'min_samples_split': [2, 5, 10],
                                        'n_estimators': [150, 200, 250]},
                   random_state=42, verbose=2)
```

Fig 1.52 Intializing RF regressor model using gridsearch.



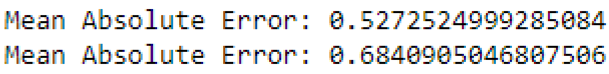
```
R square value for training data  0.5763468453214994
R square value for testing data  0.2891712997687599
```

Fig 1.53 R-Square RF regressor model using gridsearch.



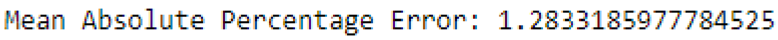
```
Mean squared error for the training data is  0.4240953609210147
Root Mean squared error for the training data is  0.6512260444123951

C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:7: D
1d array was expected. Please change the shape of y to (n_samples,),
  import sys

Mean squared error for the testing data is  0.7060135370204845
Root Mean squared error for the testing data is  0.8402461169327023
```

Fig 1.54 Mean Squared Error and Root Mean Squared Error RF regressor model using gridsearch.



```
Mean Absolute Error: 0.5272524999285084
Mean Absolute Error: 0.6840905046807506
```

Fig 1.55 Mean Absolute Error RF regressor model using gridsearch.



```
Mean Absolute Percentage Error: 1.2833185977784525
```

Fig 1.56 Mean Absolute Percentage Error RF regressor model using gridsearch.

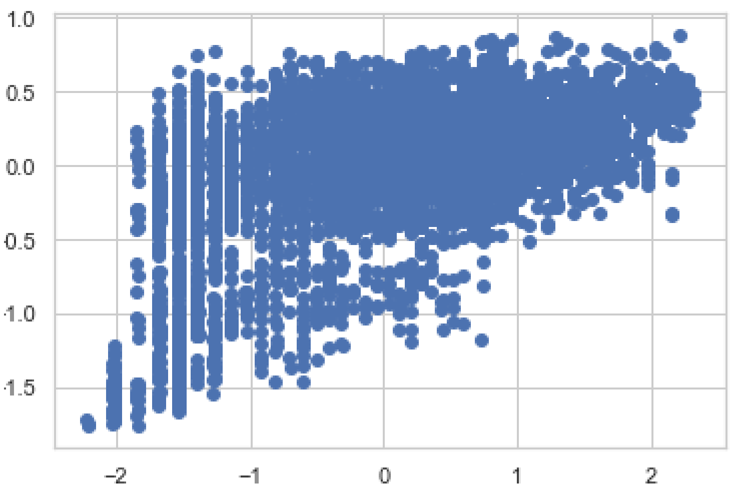
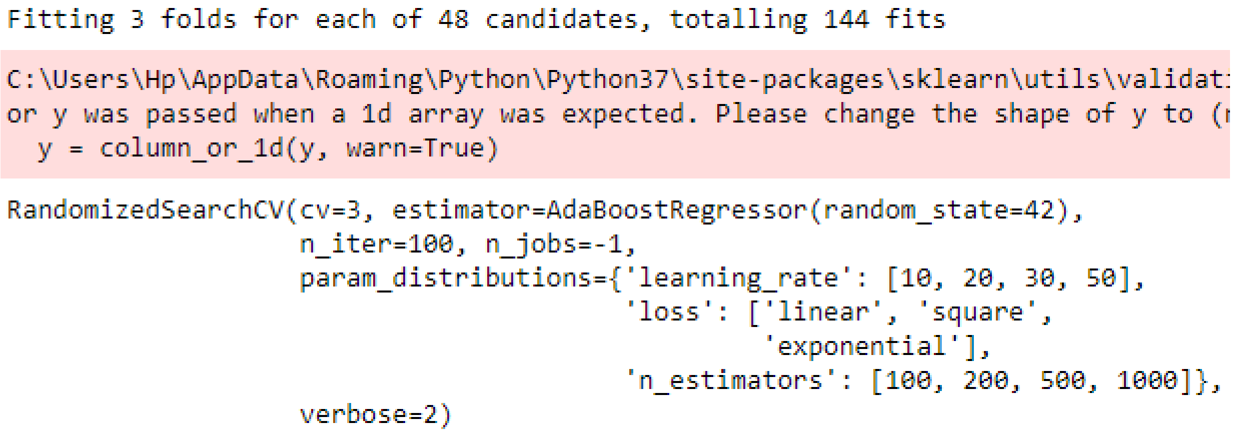


Fig 1.57 Scatterplot RF regressor model using gridsearch.

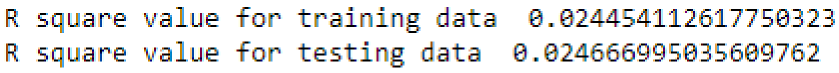
Gradient Boosting Regressor:

Gradient boosting Regression **calculates the difference between the current prediction and the known correct target value**. This difference is called residual. After that Gradient boosting Regression trains a weak model that maps features to that residual.



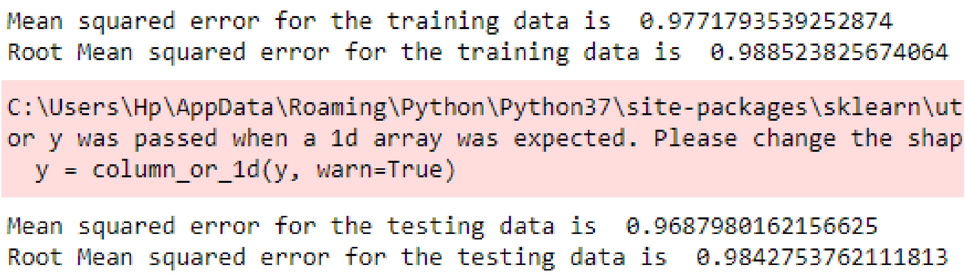
```
Fitting 3 folds for each of 48 candidates, totalling 144 fits

C:\Users\Hp\AppData\Roaming\Python\Python37\site-packages\sklearn\utils\validat
or y was passed when a 1d array was expected. Please change the shape of y to (
  y = column_or_1d(y, warn=True)

RandomizedSearchCV(cv=3, estimator=AdaBoostRegressor(random_state=42),
                   n_iter=100, n_jobs=-1,
                   param_distributions={'learning_rate': [10, 20, 30, 50],
                                        'loss': ['linear', 'square',
                                                 'exponential'],
                                        'n_estimators': [100, 200, 500, 1000]},
                   verbose=2)
```

Fig 1.58 Initialising Gradient boosting regressor model using gridsearch.



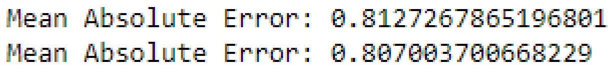
```
R square value for training data  0.024454112617750323
R square value for testing data  0.024666995035609762
```

Fig 1.59 R-Square Gradient boosting regressor model using gridsearch.



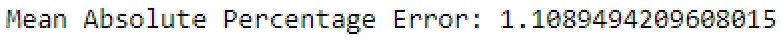
```
Mean squared error for the training data is  0.9771793539252874
Root Mean squared error for the training data is  0.988523825674064

C:\Users\Hp\AppData\Roaming\Python\Python37\site-packages\sklearn\ut
or y was passed when a 1d array was expected. Please change the shap
  y = column_or_1d(y, warn=True)

Mean squared error for the testing data is  0.9687980162156625
Root Mean squared error for the testing data is  0.9842753762111813
```

Fig 1.60 Mean Squared Error and Root Mean Squared Error Gradient boosting regressor model using gridsearch.



```
Mean Absolute Error: 0.8127267865196801
Mean Absolute Error: 0.807003700668229
```

Fig 1.61 Mean Absolute Error Gradient boosting regressor model using gridsearch



```
Mean Absolute Percentage Error: 1.1089494209608015
```

Fig 1.62 Mean Absolute Percentage Error Gradient boosting regressor model using gridsearch

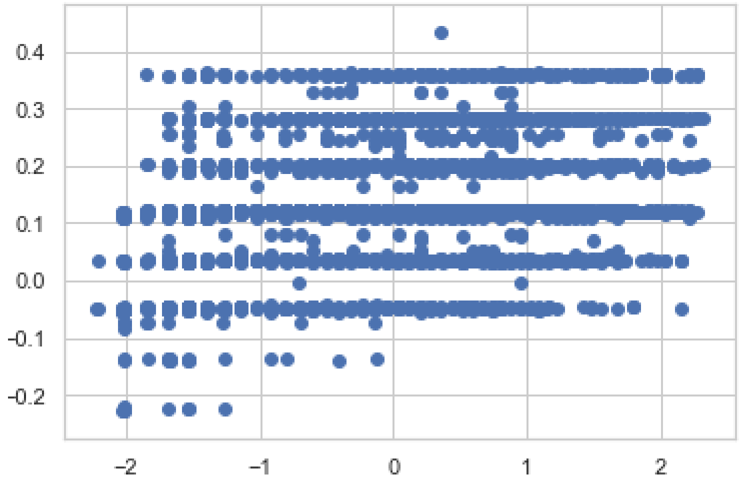
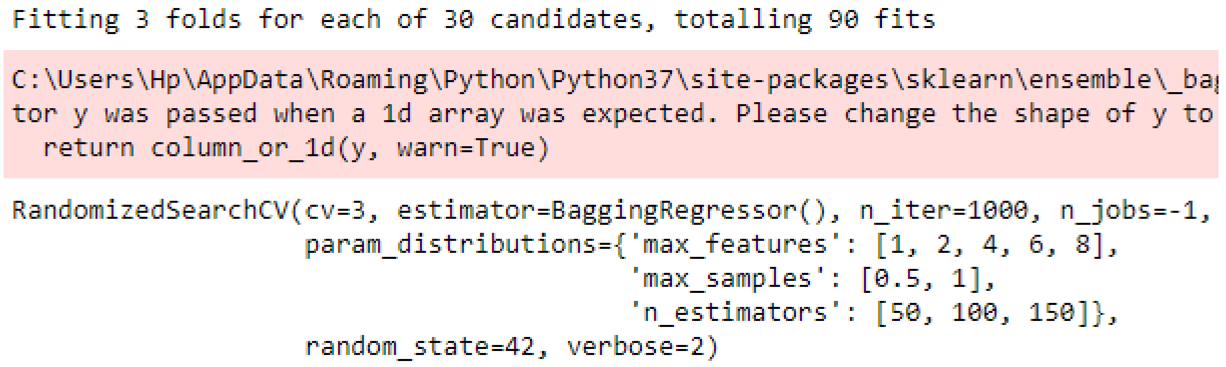


Fig 1.63 Scatterplot of Gradient boosting regressor model using gridsearch

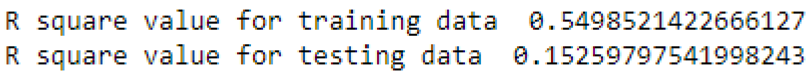
Bagging Regressor:

A Bagging regressor is an **ensemble meta-estimator that fits base regressor each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction.**



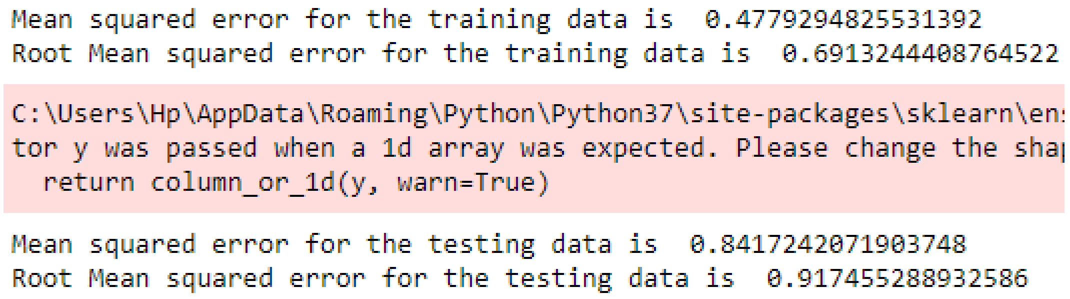
```
Fitting 3 folds for each of 30 candidates, totalling 90 fits

C:\Users\Hp\AppData\Roaming\Python\Python37\site-packages\sklearn\ensemble\_ba
tor y was passed when a 1d array was expected. Please change the shape of y to
  return column_or_1d(y, warn=True)

RandomizedSearchCV(cv=3, estimator=BaggingRegressor(), n_iter=1000, n_jobs=-1,
                   param_distributions={'max_features': [1, 2, 4, 6, 8],
                                        'max_samples': [0.5, 1],
                                        'n_estimators': [50, 100, 150]},
                   random_state=42, verbose=2)
```

Fig 1.64 Initialising Bagging regressor model using gridsearch.



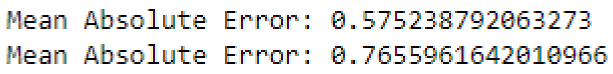
```
R square value for training data  0.5498521422666127
R square value for testing data  0.15259797541998243
```

Fig 1.65 R-Square Bagging regressor model using gridsearch.



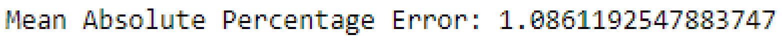
```
Mean squared error for the training data is  0.4779294825531392
Root Mean squared error for the training data is  0.6913244408764522

C:\Users\Hp\AppData\Roaming\Python\Python37\site-packages\sklearn\en
tor y was passed when a 1d array was expected. Please change the sha
  return column_or_1d(y, warn=True)

Mean squared error for the testing data is  0.8417242071903748
Root Mean squared error for the testing data is  0.917455288932586
```

Fig 1.66 Mean Squared Error and Root Mean Squared Error of Bagging regressor model using gridsearch.



```
Mean Absolute Error: 0.575238792063273
Mean Absolute Error: 0.7655961642010966
```

Fig 1.67 Mean Absolute Error of Bagging regressor model using gridsearch.



```
Mean Absolute Percentage Error: 1.0861192547883747
```

Fig 1.68 Mean Absolute Percentage Error of Bagging regressor model using gridsearch.

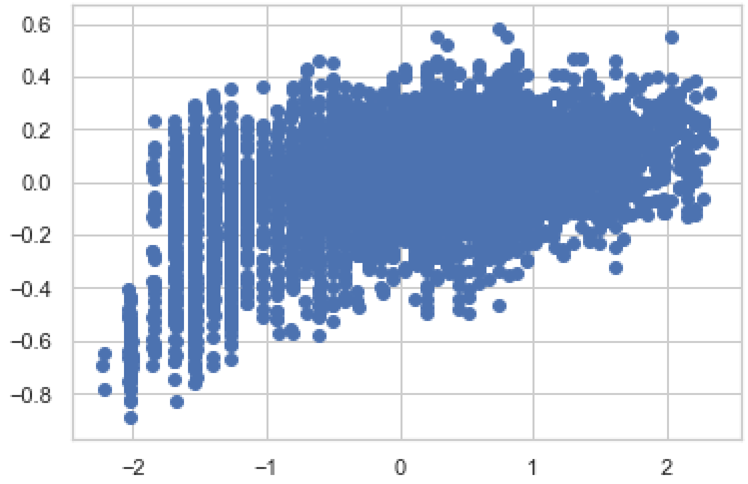
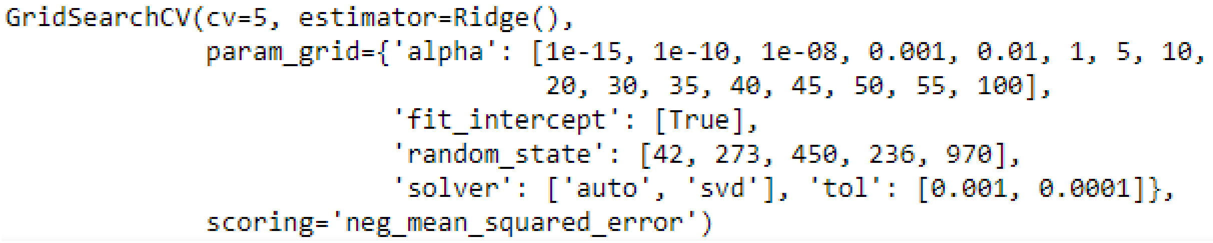


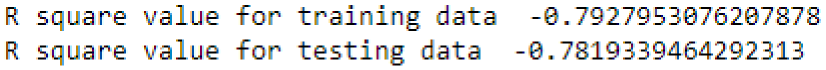
Fig 1.69 Scatterplot of Bagging regressor model using gridsearch.

Ridge Regression:



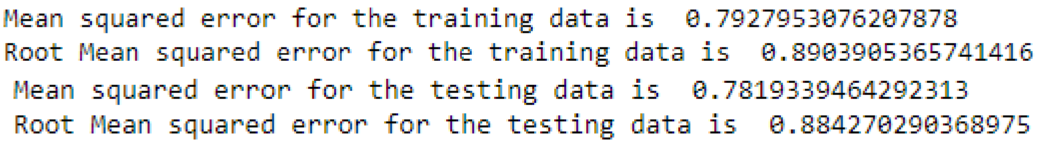
```
GridSearchCV(cv=5, estimator=Ridge(),
             param_grid={'alpha': [1e-15, 1e-10, 1e-08, 0.001, 0.01, 1, 5, 10,
                                   20, 30, 35, 40, 45, 50, 55, 100],
                         'fit_intercept': [True],
                         'random_state': [42, 273, 450, 236, 970],
                         'solver': ['auto', 'svd'], 'tol': [0.001, 0.0001]},
             scoring='neg_mean_squared_error')
```

Fig 1.70 Initialising Ridge regressor model using gridsearch.



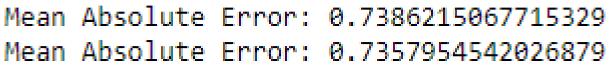
```
R square value for training data  -0.7927953076207878
R square value for testing data  -0.7819339464292313
```

Fig 1.71 R-Square Ridge regressor model using gridsearch.



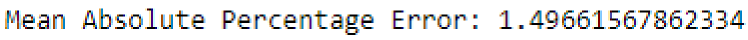
```
Mean squared error for the training data is  0.7927953076207878
Root Mean squared error for the training data is  0.8903905365741416
 Mean squared error for the testing data is  0.7819339464292313
 Root Mean squared error for the testing data is  0.884270290368975
```

Fig 1.72 Mean Squared Error and Root Mean Squared Error of Ridge regressor model using gridsearch.



```
Mean Absolute Error: 0.7386215067715329
Mean Absolute Error: 0.7357954542026879
```

Fig 1.73 Mean Absolute Error of Ridge regressor model using gridsearch.



```
Mean Absolute Percentage Error: 1.49661567862334
```

Fig 1.74 Mean Absolute Percentage Error of Ridge regressor model using gridsearch.

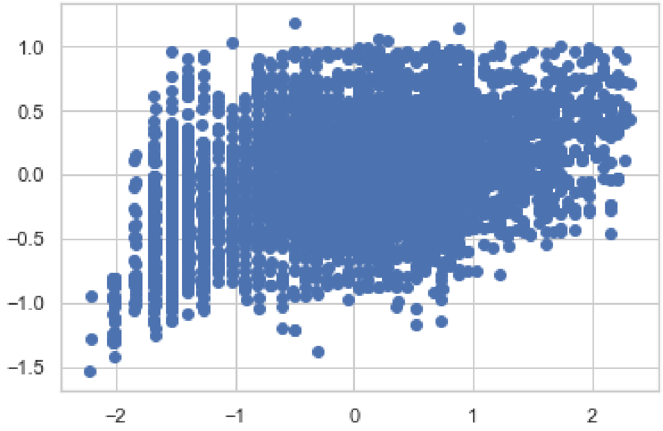


Fig 1.75 Scatterplot of Ridge regressor model using gridsearch.

b. Any other model tuning measures(if applicable)

There are 2 types of model tuning are

- Hyper parameter tuning based on grid search
- Hyper parameter tuning based on random search

c. Interpretation of the most optimum model and its implication on the business

The most optimum model is Random Forest Regressor using hyper parameters model tuning gives the best values among the other Hyper tuning model. The score of the model is the highest among all the models.

The errors are the least when compared to other models and all the performance metrics are within optimum limits.

So, after observing all these points we can conclude that the **Random Forest Regressor using hyper parameters model tuning** is the most optimum model.

Implications of regression models:

The two main uses for regression in business are forecasting and optimization. In addition to helping managers predict such things as future demand for their products, regression analysis **helps fine-tune manufacturing and delivery processes.** In supply chain management, forecasting the optimum weight of the product and forecast the demand and supply of products based on the warehouse

Regression also helps in analysing past data on stocks prices and trends to identify patterns

Regression Analysis helps the business to understand the data points they have and lead to understand the relationships between dependent and independent variable and to make better decisions.

These can help in fine tuning the manufacturing and delivery process. These regression algorithms can also be used in optimizing the warehouse stock.

Regression models predicting consumer behaviour which can help in targeted marketing and product development

Applications of Regression model in real time:

1. **Medical field** - Medical researchers often use linear regression to understand the relationship between drug dosage and blood pressure of patients
2. **Agriculture** - Agricultural scientists often use linear regression to measure the effect of fertilizer and water on crop yields