House Price Predication

Submitted by:

E.Sanjana

ACKNOWLEDGMENT

Following websites are used in completion of the project.

1. https://stackoverflow.com
2. https://scikit-learn.org
3. https://machinelearningmastery.com
4. https://www.geeksforgeeks.org
5. https://pandas.pydata.org
6. https://www.machinelearningplus.com

INTRODUCTION

❖ Business Problem Framing

- Conducting Over all Data Analysis on the given data set and finding Features/Parameters which are highly affecting the selling price of the house.
- Building a model to predict the selling price of a house on the given parameters and determine which model works best, the top 15 features which are being used by the model in predicting the selling price.
- These Predictions made will be help full in setting up an ideal price considering all the factors.

❖ Conceptual Background of the Domain Problem

In General, the Price of a house depends on many factors: Location, Utilities available, Distance from the Major locations of the city, Lot Area (Sq. ft), Type of the house (apartment, stand alone, Etc), These factors will majorly affect the price of the house.

❖ Review of Literature

- In this problem predicting the right Price for the house will be the main objective.
- Which features decide the price.
- Algorithms used in the model building helps us predict the price.
- Removal of outliers and unrealistic values from the dataset helps us in predictions.

❖ Motivation for the Problem Undertaken

- The motive of the problem under taken will be to determine the sale price for the houses with the help of other parameters that can are provided.
- This will further be help full in know which parameters rise or lower the price of a particular house.
- The following Domine – Retail Sector will be and has been a very important part of the human life cycle and buying and selling to house determining a right price for both will play an important role.

Analytical Problem Framing

❖ Mathematical/ Analytical Modeling of the Problem

• Following problem is a Regressor problem where we have to determine the price of a house

• The Regressor algorithms used in the following price prediction problem for the project are:

Base Algorithms

- Linear Regression ()
- DecisionTreeRegression ()
- Lasso ()
- Ridge Regression ()
- KNeighborsRegressor ()
- SVR () – Support Vector Regressor ()

Ensemble Techniques

- AdaBoost Regressor ()
- GradientBoosting Regressor ()
- Bagging Regressor ()
- ExtraTrees Regressor ()
- RandomForest Regressor ()
- XGBoost Regressor ()

Outlier Removal Techniques

For this Particular problem we are not going to remove outliers as the data that we have contains.

Trian with 1168Rows and Test with 291 Rows which is already very low to build a proper model and as soon as we remove outliers, we will be losing few unique attributes which are numbered very few, like a Pool in the house.

Thus, we will be using the data wholly.

❖ Data Sources and their formats

<u>Train Dataset</u>

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1
0	127	120	RL	NaN	4928	Pave	NaN	IR1	Lvl	AllPub	Inside	Gtl	NPkVill	Norm
1	889	20	RL	95.0	15865	Pave	NaN	IR1	Lvl	AllPub	Inside	Mod	NAmes	Norm
2	793	60	RL	92.0	9920	Pave	NaN	IR1	Lvl	AllPub	CulDSac	Gtl	NoRidge	Norm
3	110	20	RL	105.0	11751	Pave	NaN	IR1	Lvl	AllPub	Inside	Gtl	NWAmes	Norm
4	422	20	RL	NaN	16635	Pave	NaN	IR1	Lvl	AllPub	FR2	Gtl	NWAmes	Norm
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1163	289	20	RL	NaN	9819	Pave	NaN	IR1	Lvl	AllPub	Inside	Gtl	Sawyer	Norm
1164	554	20	RL	67.0	8777	Pave	NaN	Reg	Lvl	AllPub	Inside	Gtl	Edwards	Feedr
1165	196	160	RL	24.0	2280	Pave	NaN	Reg	Lvl	AllPub	FR2	Gtl	NPkVill	Norm
1166	31	70	C (all)	50.0	8500	Pave	Pave	Reg	Lvl	AllPub	Inside	Gtl	IDOTRR	Feedr
1167	617	60	RL	NaN	7861	Pave	NaN	IR1	Lvl	AllPub	Inside	Gtl	Gilbert	Norm

1168 rows × 81 columns

There are 1168 Rows of Data and 81 Columns

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1168 entries, 0 to 1167
Data columns (total 81 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Id              1168 non-null   int64
 1   MSSubClass      1168 non-null   int64
 2   MSZoning        1168 non-null   object
 3   LotFrontage     954 non-null    float64
 4   LotArea         1168 non-null   int64
 5   Street          1168 non-null   object
 6   Alley           77 non-null     object
 7   LotShape        1168 non-null   object
 8   LandContour     1168 non-null   object
 9   Utilities       1168 non-null   object
 10  LotConfig       1168 non-null   object
 11  LandSlope       1168 non-null   object
 12  Neighborhood    1168 non-null   object
 13  Condition1      1168 non-null   object
 14  Condition2      1168 non-null   object
 15  BldgType        1168 non-null   object
 16  HouseStyle      1168 non-null   object
 17  OverallQual     1168 non-null   int64
 18  OverallCond     1168 non-null   int64
 19  YearBuilt       1168 non-null   int64
 20  YearRemodAdd    1168 non-null   int64
 21  RoofStyle       1168 non-null   object
 22  RoofMatl        1168 non-null   object
 23  Exterior1st     1168 non-null   object
 24  Exterior2nd     1168 non-null   object
 25  MasVnrType      1161 non-null   object
 26  MasVnrArea      1161 non-null   float64
 27  ExterQual       1168 non-null   object
 28  ExterCond       1168 non-null   object
 29  Foundation      1168 non-null   object
 30  BsmtQual        1138 non-null   object
 31  BsmtCond        1138 non-null   object
 32  BsmtExposure    1137 non-null   object
 33  BsmtFinType1    1138 non-null   object
 34  BsmtFinSF1      1168 non-null   int64
 35  BsmtFinType2    1137 non-null   object
 36  BsmtFinSF2      1168 non-null   int64
 37  BsmtUnfSF       1168 non-null   int64
 38  TotalBsmtSF     1168 non-null   int64
 39  Heating         1168 non-null   object
 40  HeatingQC       1168 non-null   object
 41  CentralAir      1168 non-null   object
 42  Electrical      1168 non-null   object
 43  1stFlrSF        1168 non-null   int64
 44  2ndFlrSF        1168 non-null   int64
 45  LowQualFinSF    1168 non-null   int64
 46  GrLivArea       1168 non-null   int64
 47  BsmtFullBath    1168 non-null   int64
 48  BsmtHalfBath    1168 non-null   int64
 49  FullBath        1168 non-null   int64
 50  HalfBath        1168 non-null   int64
 51  BedroomAbvGr    1168 non-null   int64
 52  KitchenAbvGr    1168 non-null   int64
 53  KitchenQual     1168 non-null   object
 54  TotRmsAbvGrd    1168 non-null   int64
 55  Functional      1168 non-null   object
 56  Fireplaces      1168 non-null   int64
 57  FireplaceQu     617 non-null    object
 58  GarageType      1104 non-null   object
 59  GarageYrBlt     1104 non-null   float64
 60  GarageFinish    1104 non-null   object
 61  GarageCars      1168 non-null   int64
 62  GarageArea      1168 non-null   int64
 63  GarageQual      1104 non-null   object
 64  GarageCond      1104 non-null   object
 65  PavedDrive      1168 non-null   object
 66  WoodDeckSF      1168 non-null   int64
 67  OpenPorchSF     1168 non-null   int64
 68  EnclosedPorch   1168 non-null   int64
 69  3SsnPorch       1168 non-null   int64
 70  ScreenPorch     1168 non-null   int64
 71  PoolArea        1168 non-null   int64
 72  PoolQC          7 non-null      object
 73  Fence           237 non-null    object
 74  MiscFeature     44 non-null     object
 75  MiscVal         1168 non-null   int64
 76  MoSold          1168 non-null   int64
 77  YrSold          1168 non-null   int64
 78  SaleType        1168 non-null   object
 79  SaleCondition   1168 non-null   object
 80  SalePrice       1168 non-null   int64
```

Following are the Columns and the Datatypes and number of values each column has.

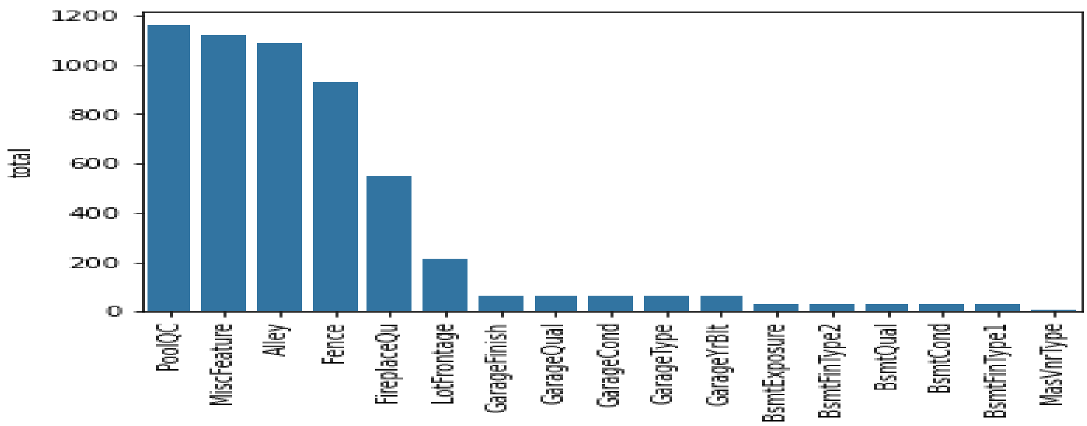
Test Dataset

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1
0	337	20	RL	86.0	14157	Pave	NaN	IR1	HLS	AllPub	Corner	Gtl	StoneBr	Norm
1	1018	120	RL	NaN	5814	Pave	NaN	IR1	Lvl	AllPub	CulDSac	Gtl	StoneBr	Norm
2	929	20	RL	NaN	11838	Pave	NaN	Reg	Lvl	AllPub	Inside	Gtl	CollgCr	Norm
3	1148	70	RL	75.0	12000	Pave	NaN	Reg	Bnk	AllPub	Inside	Gtl	Crawfor	Norm
4	1227	60	RL	86.0	14598	Pave	NaN	IR1	Lvl	AllPub	CulDSac	Gtl	Somerst	Feedr
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
287	83	20	RL	78.0	10206	Pave	NaN	Reg	Lvl	AllPub	Inside	Gtl	Somerst	Norm
288	1048	20	RL	57.0	9245	Pave	NaN	IR2	Lvl	AllPub	Inside	Gtl	CollgCr	Norm
289	17	20	RL	NaN	11241	Pave	NaN	IR1	Lvl	AllPub	CulDSac	Gtl	NAmes	Norm
290	523	50	RM	50.0	5000	Pave	NaN	Reg	Lvl	AllPub	Corner	Gtl	BrkSide	Feedr
291	1379	160	RM	21.0	1953	Pave	NaN	Reg	Lvl	AllPub	Inside	Gtl	BrDale	Norm

292 rows × 80 columns

❖ Data Pre-processing Done

The House price data has some missing values to be handled with we are going fill the data according to the understanding and analysis made on the data.




```
#Test Data Cleaning and filling the Nan Values
df_test['LotFrontage'].fillna(df_test['LotFrontage'].median(),inplace= True)#Filling missing values with Median of the data
df_test['Alley'].fillna('NA',inplace= True)                                    #Filling Nan Values with NA Catagory
df_test['MasVnrType'].fillna(df_test['MasVnrType'].mode()[0],inplace= True) #Filling missing values with Mode of the data
df_test['MasVnrArea'].fillna(df_test['MasVnrArea'].mean(),inplace= True)      #Filling missing values with Mean of the data
df_test['BsmtQual'].fillna('NA',inplace= True)                              #Filling Nan Values with NA Catagory
df_test['BsmtCond'].fillna('NA',inplace= True)                             #Filling Nan Values with NA Catagory
df_test['BsmtExposure'].fillna('NA',inplace= True)                       #Filling Nan Values with NA Catagory
df_test['BsmtFinType1'].fillna('NA',inplace= True)                      #Filling Nan Values with NA Catagory
df_test['BsmtFinType2'].fillna('NA',inplace= True)                     #Filling Nan Values with NA Catagory
df_test['FireplaceQu'].fillna('NA',inplace= True)                     #Filling Nan Values with NA Catagory

#Filling Garage Null Values with NA and Year with 0.0
df_test['GarageType'].fillna('NA',inplace= True)                       #Filling Nan Values with NA Catagory
df_test['GarageFinish'].fillna('NA',inplace= True)                    #Filling Nan Values with NA Catagory
df_test['GarageQual'].fillna('NA',inplace= True)                     #Filling Nan Values with NA Catagory
df_test['GarageCond'].fillna('NA',inplace= True)                    #Filling Nan Values with NA Catagory

#GarageYrBlt
# And 0 in the place of the year in Following GarageYrBlt
df_test['GarageYrBlt'].fillna(0.0,inplace= True)

df_test['PoolQC'].fillna('NA',inplace= True)                  #Filling Nan values with NA
df_test['Fence'].fillna('NA',inplace= True)                  #Filling Nan values with NA
df_test['MiscFeature'].fillna('NA',inplace= True)         #Filling Nan values with NA
df_test['Electrical'].fillna('SBrkr',inplace= True) #Single missing value in the Electrical is replaced with Mode of the dat
```

The following columns had missing values in the dataset, that are filled using multiple techniques.

- LotFrontage – is filled with Median of the column
  - Median of LotForntage: 70.0
  - Mean of LotForntage: 70.98846960167715
  - As the Median and Mean of the data are closer, we are filling the Nan values with Median
- MasVnrType – Is filled with Mode of the column
- MasVnrArea – Is filled with Mean of the column
  - Mean of the Data: 102.31007751937983
- Columns Alley, BsmtQual, BsmtCond, BsmtExposure, BsmtFinType1, BsmtFinType2, FireplaceQu ,PoolQC, Fence, MiscFeature– are Filled with NA (Not Available) which is a Category not been filled in the dataset
- GarageType, GarageFinish, GarageQual, GarageCond all the columns have same number of missing values and had NA category so we Considered there is no Garage and Filled NA in All the missing places.
- GarageYrBlt – Is filled with 0.0 in the place of year the Garage is built as there is no Garage in the following houses.

Encoding the Data

Discreet Categorical Columns Encoding

We Have considered following Columns as Discreet Categorical values and used Onehot Encoding using get_dummies() Method to them

['MSZoning','Street','Alley','Utilities','LotConfig','Neighborhood','Condition1','Condition2','BldgType','HouseStyle','RoofStyle','RoofMatl','Exterior1st','Exterior2nd','MasVnrType','Foundation','Heating','CentralAir','GarageType','MiscFeature','SaleType','SaleCondition']

Ordinal Categorical Columns Encoding

We have Encoded Following columns using ordinal encoding and modified them to int values

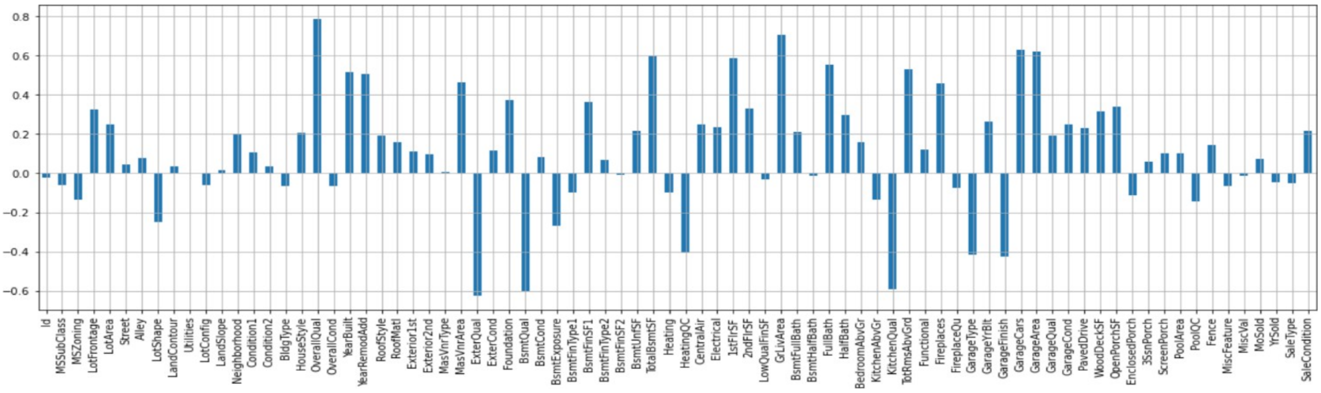
['LotShape','LandContour','LandSlope','ExterQual','ExterCond','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2','HeatingQC','Electrical','KitchenQual','Functional','FireplaceQu','GarageFinish','GarageQual','GarageCond','PavedDrive','PoolQC','Fence']

We have Reindexed the Teat data to Trains Data’s Index and started the model building.

❖ Data Inputs- Logic- Output Relationships

The inputs which play an important role in predicting the house price are the following with hight negative and positive co-relation

- **Features - Co- Relation**
- OverallQual - 0.789185
- GrLivArea - 0.707300
- GarageCars - 0.628329
- GarageArea - 0.619000
- TotalBsmtSF - 0.595042
- GarageFinish - -0.424922
- KitchenQual - -0.592468
- BsmtQual - -0.601307
- ExterQual - -0.624820

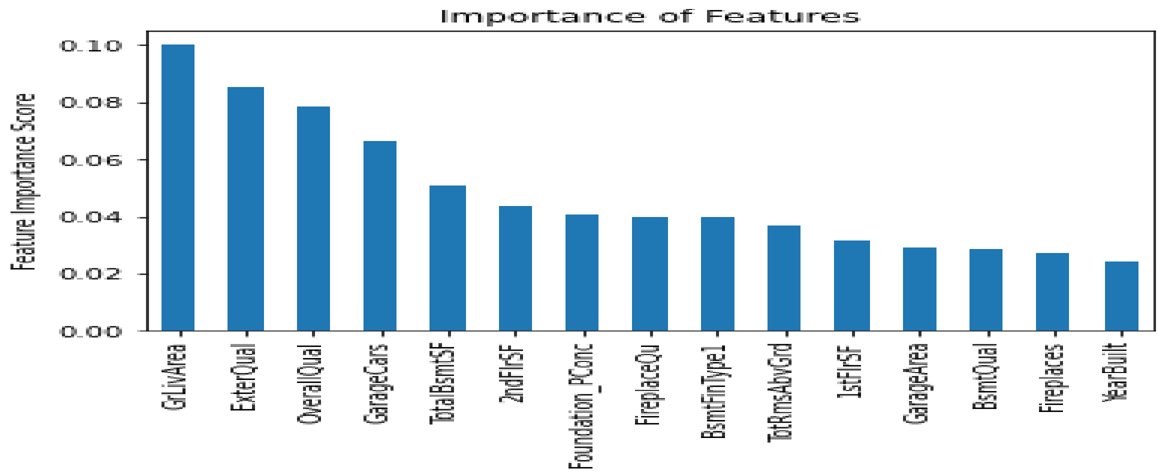


These inputs will help the model in predicting the house price

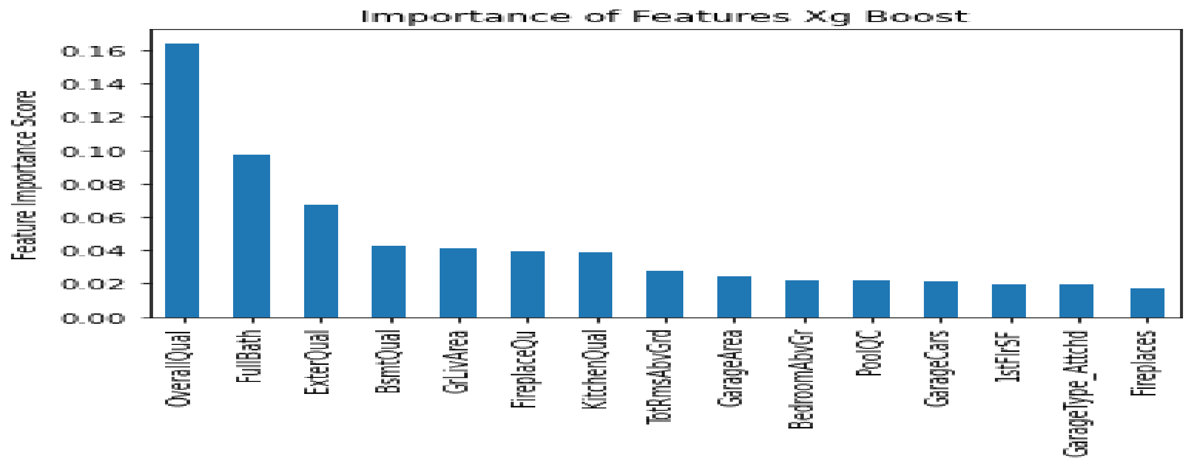
The Logic Algorithms used in this particular project are mentioned above where as the best output we received is from Gradient Boost regressor followed with XG Boost Regressor with 93% and 91% R2 Scores respectively.

Following are the Top15 Inputs Considered by the Algorithm in building the model and predicting the House price accurately.

Gradient Boost Important features.



XG Boost Important Features.



- State the set of assumptions (if any) related to the problem under consideration

❖ Following are the assumptions made to in Data cleaning and Encoding

➢ LotFrontage – is filled with Median of the column

- Median of LotForntage: 70.0
- Mean of LotForntage: 70.98846960167715
- As the Median and Mean of the data are closer, we are filling the Nan values with Median

➢ MasVnrType – Is filled with Mode of the column

➢ MasVnrArea – Is filled with Mean of the column

- Mean of the Data: 102.31007751937983

➢ Columns Alley, BsmtQual, BsmtCond, BsmtExposure, BsmtFinType1, BsmtFinType2, FireplaceQu ,PoolQC, Fence, MiscFeature– are Filled with NA (Not Available) which is a Category not been filled in the dataset

➢ GarageType, GarageFinish, GarageQual, GarageCond all the columns have same number of missing values and had NA category so we Considered there is no Garage and Filled NA in All the missing places.

➢ GarageYrBlt – Is filled with 0.0 in the place of year the Garage is built as there is no Garage in the following houses.

Encoding the Data

- **Discreet Categorical Columns Encoding**
  - We Have considered following Columns as Discreet Categorical values and used Onehot Encoding using get_dummies() Method to them

['MSZoning','Street','Alley','Utilities','LotConfig','Neighborhood','Condition1','Condition2','BldgType','HouseStyle','RoofStyle','RoofMatl','Exterior1st','Exterior2nd','MasVnrType','Foundation','Heating','CentralAir','GarageType','MiscFeature','SaleType','SaleCondition']

Ordinal Categorical Columns Encoding

- We have Encoded Following columns using ordinal encoding and modified them to int values

['LotShape','LandContour','LandSlope','ExterQual','ExterCond','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2','HeatingQC','Electrical','KitchenQual','Functional','FireplaceQu','GarageFinish','GarageQual','GarageCond','PavedDrive','PoolQC','Fence']

Hardware and Software Requirements and Tools Used

- Laptop- Windows 10 – i5 processor – 8 GB RAM.
- Anaconda – Python3.
- NumPy, Pandas, Seaborn.
- Base Algorithms - Linear Regression, DecisionTreeRegression, Lasso, Ridge Regression, KNeighborsRegressor, SVR – Support Vector Regressor.

- Ensemble Techniques - AdaBoost Regressor, GradientBoosting Regressor, Bagging Regressor, ExtraTrees Regressor, RandomForest Regressor, XGBoost Regressor.

- Zscore, StandardScaler.
- Model Selection - train_test_split, KFold, cross_val_score, GridSearchCV
- Metrics – r2_score, Mean Absolute Error, Mean Squared Error

Model/s Development and Evaluation

❖ Identification of possible problem-solving approaches (methods)

➢ Following algorithms are used for estimating the best possible way to get better

predictions

• Base Algorithms - Linear Regression, DecisionTreeRegression, Lasso, Ridge Regression, KNeighborsRegressor, SVR – Support Vector Regressor.

• Ensemble Techniques - AdaBoost Regressor, GradientBoosting Regressor, Bagging Regressor, ExtraTrees Regressor, RandomForest Regressor, XGBoost Regressor.

➢ Following Metrices are used to analyse the best algorithm which fits the data.

• Metrics -R2_Score, Mean Absolute Error, Mean Squared Error

• StandardScaler is used to scale the data – Which Did not Result is Best Score So the data has be used with out scaling

➢ Following are used to split the data and select the best possible parameters for

algorithms.

• Model Selection - train_test_split, KFold, GridSearchCV, RandomizedSearchCV

➢ Following modules are used to plot and analyse the data.

• Plotting Modules – Seaborn, Matplotlib.

❖ Testing of Identified Approaches (Algorithms)

• Base Algorithms –

Linear Regression, DecisionTreeRegression, Lasso, Ridge Regression, KNeighborsRegressor, SVR – Support Vector Regressor.

• Ensemble Techniques –

AdaBoost Regressor, GradientBoosting Regressor, Bagging Regressor, ExtraTrees Regressor, RandomForest Regressor, XGBoost Regressor.

❖ Run and Evaluate selected models

Following as the Base line algorithms used and the R2 Scores.

```
models= [
    ("Lasso",Lasso()),
    ("Linear Regression",LinearRegression()),
    ("Decision Tree", DecisionTreeRegressor()),
    ("Ridge Regression",Ridge()),
    ("KNearest Neighbors",KNeighborsRegressor(1)),
    ("SVR",SVR())
    ]
```

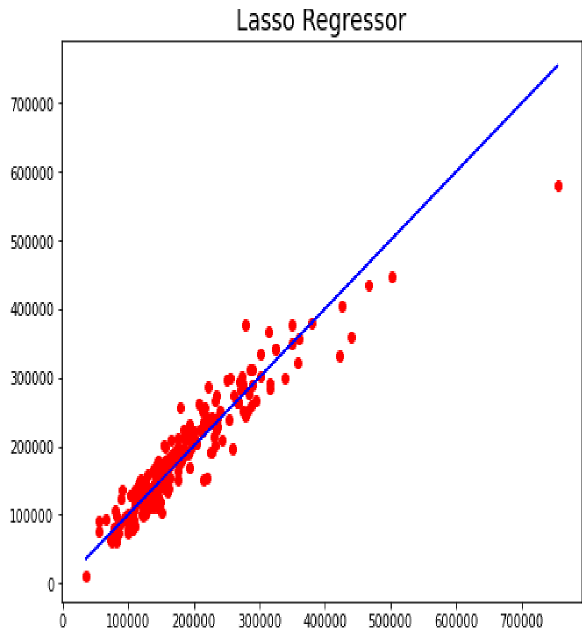
```
results = []
names = []
for name, model in models:
    model.fit(x_train,y_train)
    y_pred = model.predict(x_test)
    R2 = r2_score(y_test,y_pred)
    results.append(R2)
    names.append(name)
    msg = "R2 Score with ""%s: %f " % (name, R2*100)
    print(msg)
```

```
R2 Score with Lasso: 89.741707
R2 Score with Linear Regression: 89.574228
R2 Score with Decision Tree: 80.109223
R2 Score with Ridge Regression: 87.949643
R2 Score with KNearest Neighbors: 57.744550
R2 Score with SVR: -4.589531
```

#Lasso and Linear Regression have performed best let’s go ahead with parameter tuning.

```python
Ls=Lasso(alpha= 5.0)
Ls.fit(x_train,y_train)
Ls.score(x_train,y_train)
y_predLs = Ls.predict(x_test)

Lsscore=cross_val_score(Ls,x,y,cv=3)
Lsr=Lsscore.mean()
print("Cross value Score:",Lsr*100,'\n')

Lss_MAE = round(MAE(y_test, y_predLs),2)
Lss_MSE = round(MSE(y_test, y_predLs),2)
Lss_R2Score = round(r2_score(y_test,y_predLs),4)

print(f" Mean Absolute Error: {Lss_MAE}\n")
print(f" Mean Squared Error: {Lss_MSE}\n")
print(f" Lasso R2 Score: {Lss_R2Score*100}\n")
```

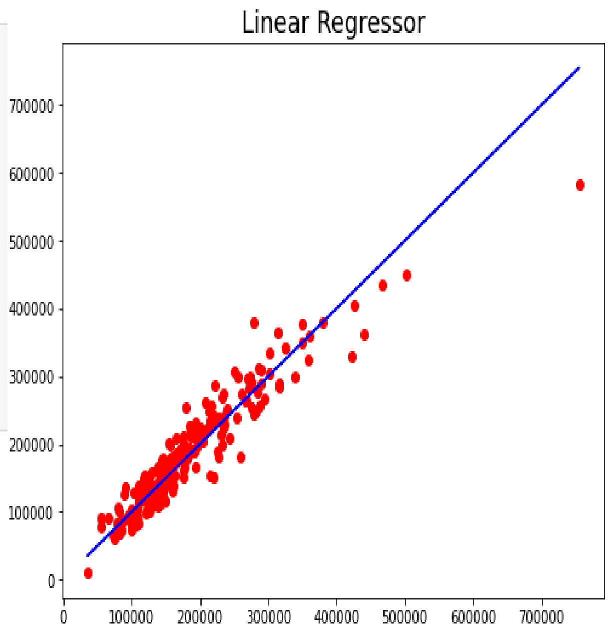
```
Cross value Score: 75.64966389458175

 Mean Absolute Error: 19682.35

 Mean Squared Error: 760473931.95

 Lasso R2 Score: 89.8
```



Linear Regressor Scores

```python
LR=LinearRegression()
LR.fit(x_train,y_train)
LR.score(x_train,y_train)
y_predLr = LR.predict(x_test)

LRscore=cross_val_score(Ls,x,y,cv=3)
LRr=LRscore.mean()
print("Cross value Score:",LRr*100,'\n')

LRs_MAE = round(MAE(y_test, y_predLr),2)
LRs_MSE = round(MSE(y_test, y_predLr),2)
LRs_R2Score = round(r2_score(y_test,y_predLr),4)

print(f" Mean Absolute Error: {LRs_MAE}\n")
print(f" Mean Squared Error: {LRs_MSE}\n")
print(f" Lasso R2 Score: {LRs_R2Score*100}\n")
```

```
Cross value Score: 75.64966389458175

 Mean Absolute Error: 19896.38

 Mean Squared Error: 777392877.26

 Lasso R2 Score: 89.57000000000001
```



Ensembling Techniques used

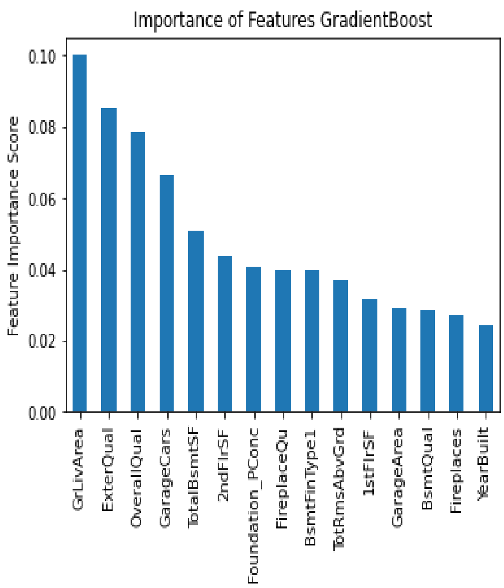
```
[AdaBoostRegressor()]:
R2 Score 0.8464100668841575
MAE 25004.74
MSE 1145236238.94
*****************************************************************


[AdaBoostRegressor(), ExtraTreesRegressor()]:
R2 Score 0.9004082250606303
MAE 17591.04
MSE 742601467.73
*****************************************************************


[AdaBoostRegressor(), ExtraTreesRegressor(), GradientBoostingRegressor()]:
R2 Score 0.9198661613230171
MAE 16344.85
MSE 597514265.14
*****************************************************************


[AdaBoostRegressor(), ExtraTreesRegressor(), GradientBoostingRegressor(), RandomForestRegressor()]:
R2 Score 0.9030595041986053
MAE 17993.76
MSE 722832327.36
*****************************************************************
```

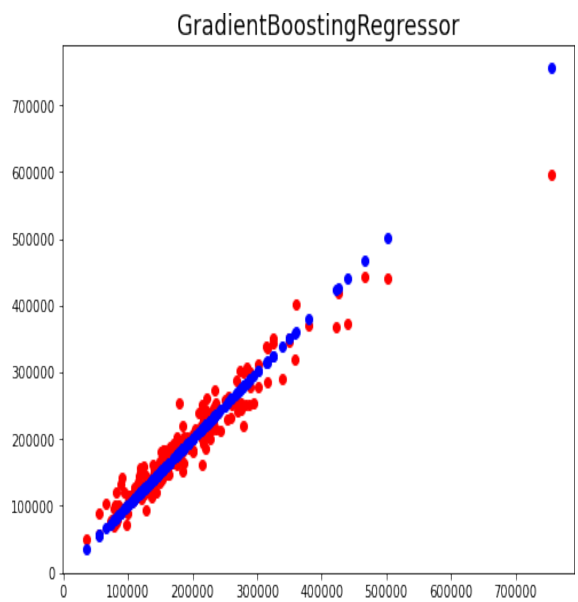
Parameter tuning for Ensembling Techniques

Gradient Boost with Parameters

```
GBR= GradientBoostingRegressor(learning_rate=0.1, n_estimators=1000,max_depth=3, min_samples_split=2,
                               min_samples_leaf=1, subsample=1,max_features='sqrt', random_state=89)
GBR.fit(x_train, y_train)

predictions = GBR.predict(x_test)
GB_MAE = round(MAE(y_test, predictions),2)
GB_MSE = round(MSE(y_test, predictions),2)
GB_R2Score = round(r2_score(y_test,predictions),4)

print(f" Mean Absolute Error: {GB_MAE}\n")
print(f" Mean Squared Error: {GB_MSE}\n")
print(f" GB Score: {GB_R2Score*100}\n")
```

```
Mean Absolute Error: 15642.27

Mean Squared Error: 511314033.51

GB Score: 93.14
```

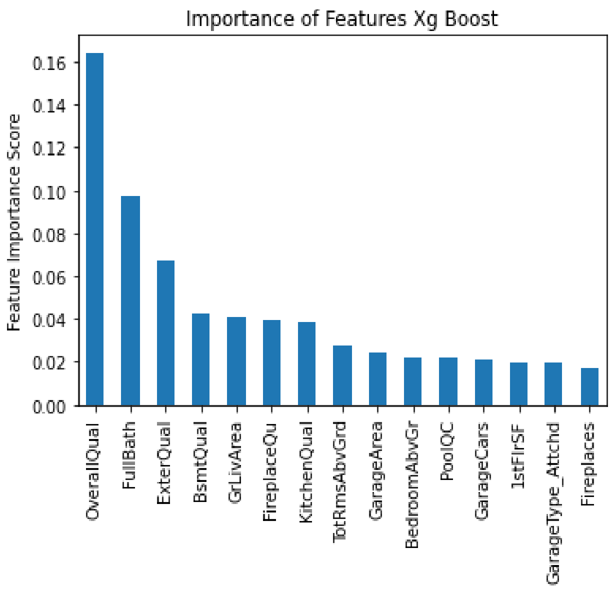


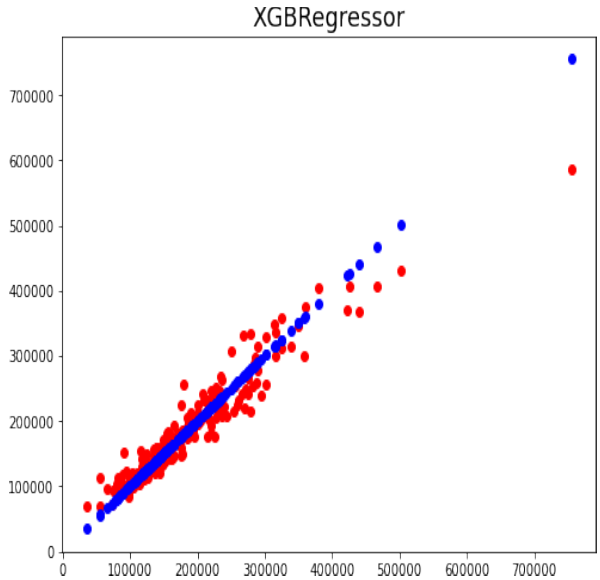


XG Boost Parameter

```
xgb_model = XGBRegressor(
          objective = 'reg:squarederror',
          colsample_bytree = 0.7,
          learning_rate = 0.1,
          max_depth = 3,
          min_child_weight = 1,
          n_estimators = 500,
          subsample = 0.5)

xgb_model.fit(x_train,y_train, early_stopping_rounds=5,
                 eval_set=[(x_test, y_test)], verbose=False)

y_pred_xgb = xgb_model.predict(x_test)
XGB_R2Score = round(r2_score(y_test,y_pred_xgb),4)
XGB_MAE = round(MAE(y_test, y_pred_xgb),2)
XGB_MSE = round(MSE(y_test, y_pred_xgb),2)


print(f" XGB Score: {XGB_R2Score*100}\n")
print(f" Mean Absolute Error: {XGB_MAE}\n")
print(f" Mean Squared Error: {XGB_MSE}\n")
```

```
XGB Score: 91.66

Mean Absolute Error: 16941.96

Mean Squared Error: 622205607.68
```





Key Metrics for success in solving problem under consideration

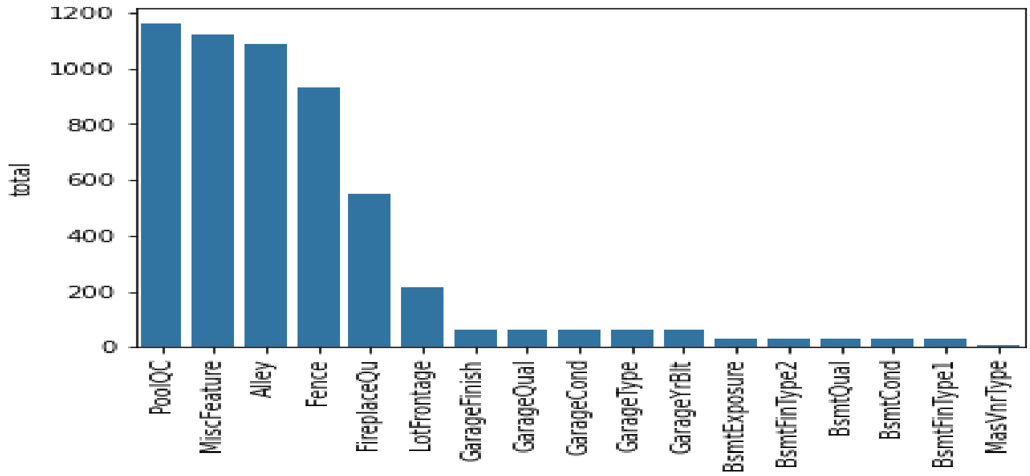
- Following are the Metrices used to evaluate the models and their respective scores.

S.no	Algorithm	R2 Score	MAE(Absolute)	MSE(Squared)
1	GradientBoostRegressor	92.83	15372.12	534544208.02
2	XGBRegressor	91.84	15388.96	609304374.75
3	RandomForestRegressor	89.60	19313.62	775762889.76
4	Lasso Regressor	89.8	19682.35	760473931.95
5	Linear Regressor	89.57	19896.38	777392877.26

- **R2 Score**- “Total variance explained by model/ Total variance”. So, if it is 100%, the two variables are perfectly correlated, i.e., with no variance at all. A low value would show a low level of correlation, meaning a regression model that is not valid.
- **MAE** – Mean Absolute error is the average of the errors. The larger the number greater the error.
- **MSE** - Mean square error is the average of the square of the errors. The larger the number the larger the error.

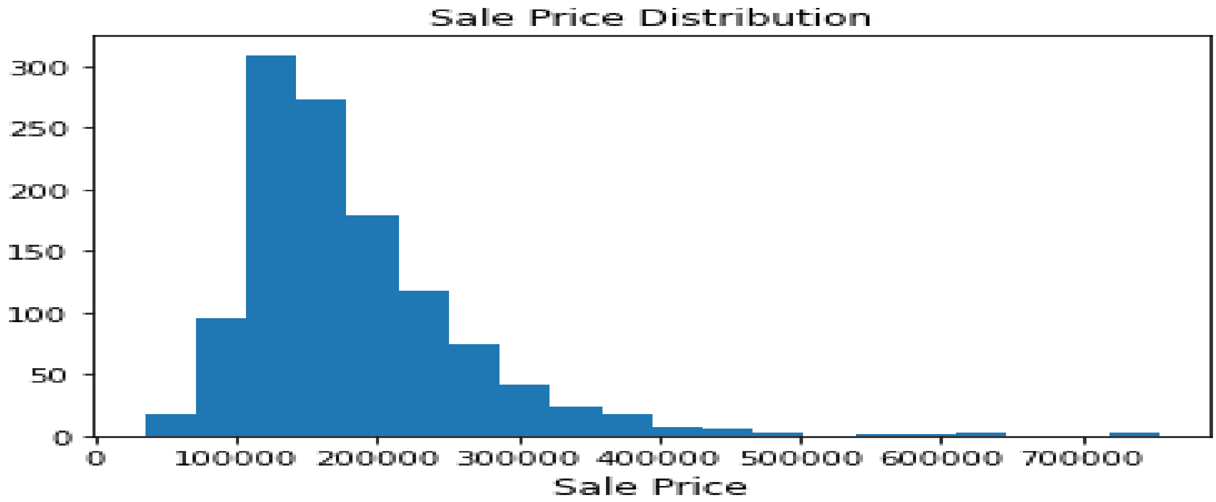
Visualizations

- **Following Graph shows the missing values and their count**



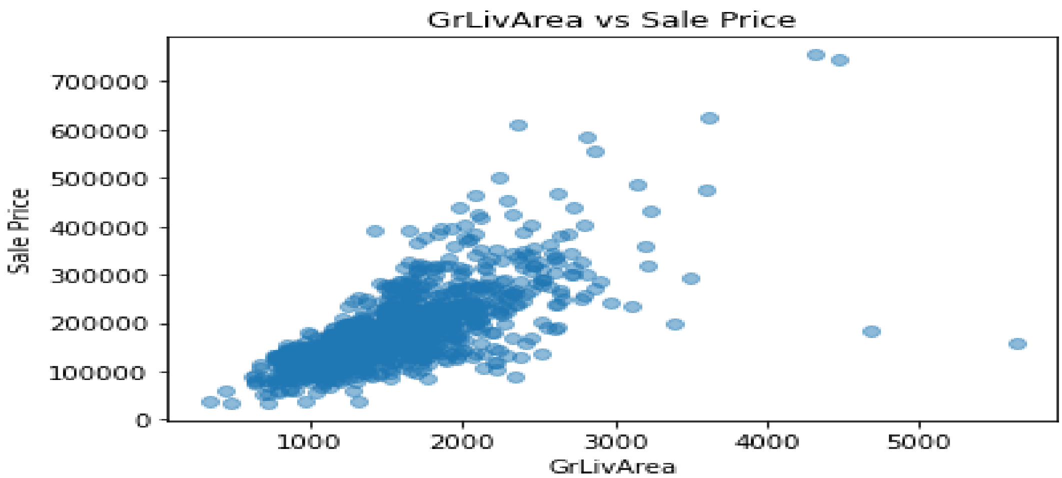
- Following columns have been filled using different techniques according to the analysis

- **Distribution of the Sale Prices**



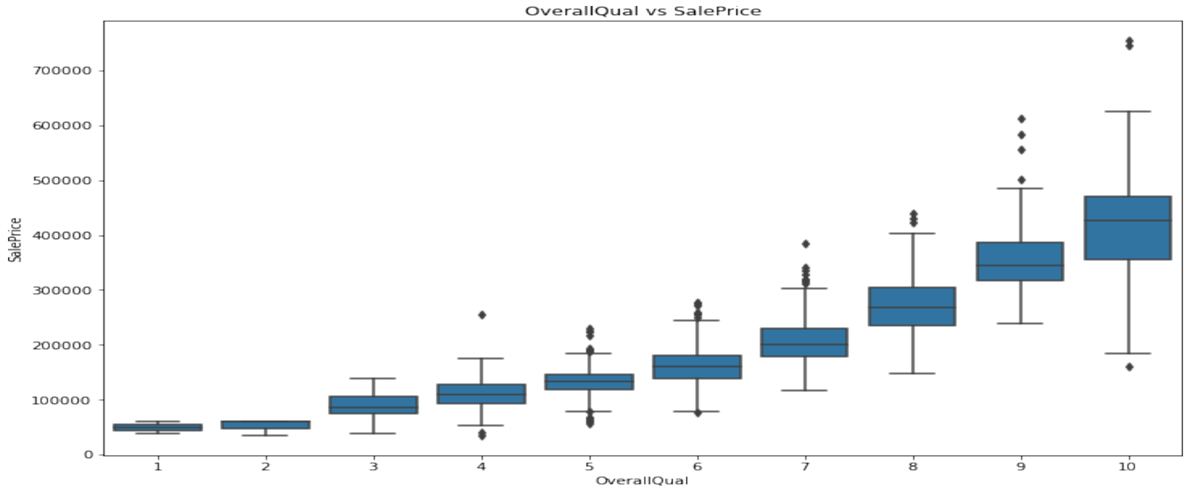
- From the Graph we can observe that most of the houses are prices between 10,000$ - 25,000$
- There are few outliers in the data but in the interest of having diversity rows have not been dropped.

- **Distribution of Living Area**



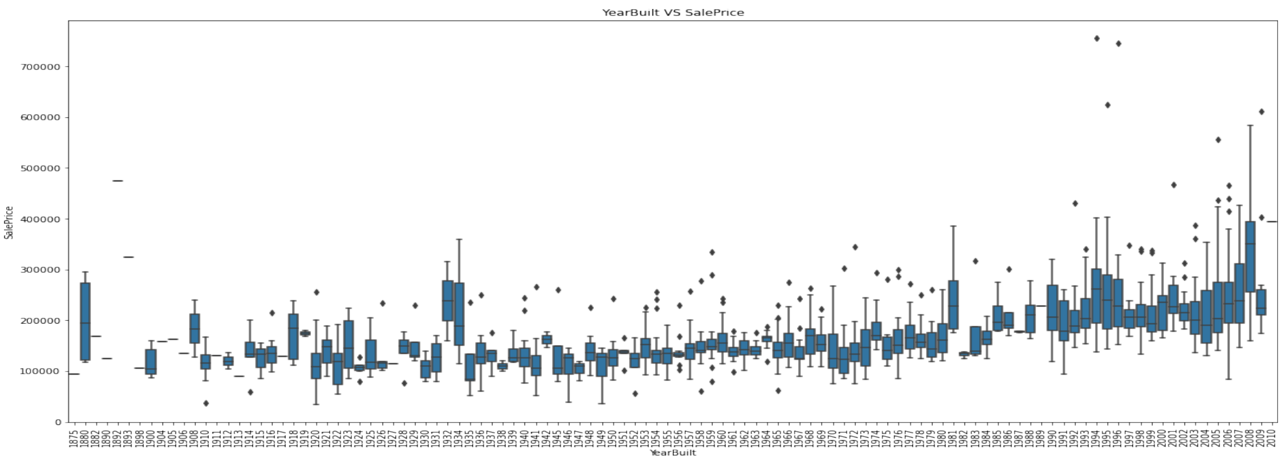
- From the Scatterplot we can observe that the majority of the Dataset Contains Live Area from – 800Sq Ft – 2500Sq Ft
- Houses with Live Area greater than 3000Sq Ft are mostly priced above 20,000$

➢ **Overall Quality to the Sale price of the house**

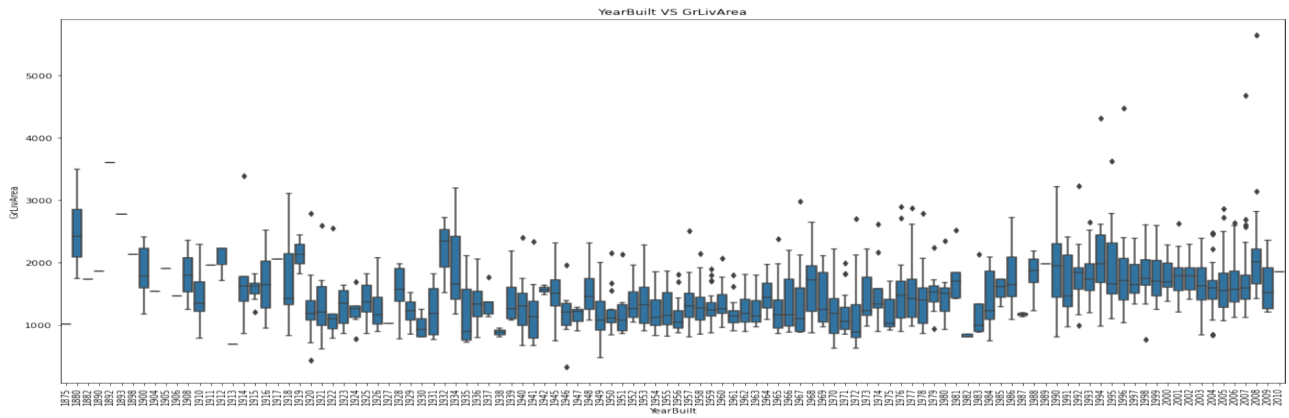


- From this we can observe that as the Quality rating increases the house price is also have a positive effect where the average price of house’s rated 10 is above 40,000$.
- Houses Rated 1-2 with OverallQual as priced less than 10,000$.

➢ **Year Built to the Sale Price of the house**



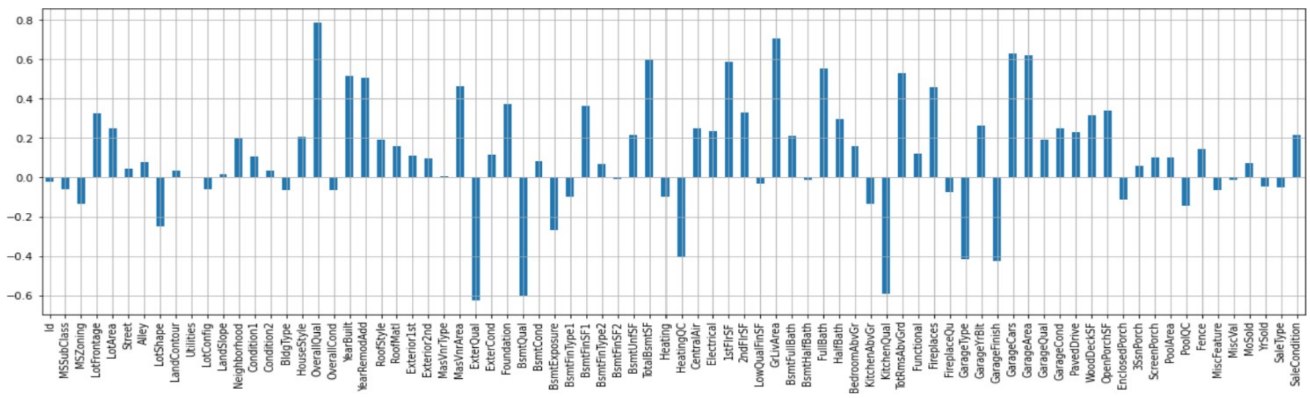
Year to Livearea



Observations made from comparing both the Graphs

- Older Houses are mostly under 20,000$ with few exceptions – which can be because of the Live area which we can observe comparing both Graphs.
- We can say that the Year of Building the house is positively affecting the price of the house even without gear difference in live area.

➢ **Effect of each Variable on Sale price of the house**

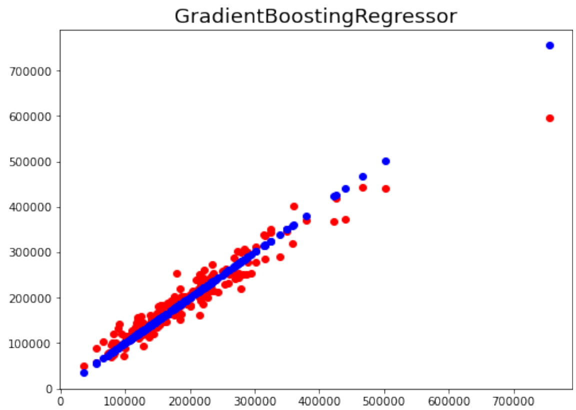


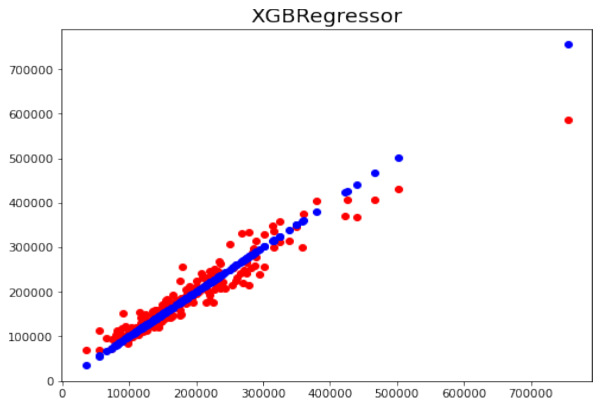
- **Features - Correlation**
- OverallQual - 0.789185
- GrLivArea - 0.707300
- GarageCars - 0.628329
- GarageArea - 0.619000
- TotalBsmtSF - 0.595042
- GarageFinish - -0.424922
- KitchenQual - -0.592468
- BsmtQual - -0.601307
- ExterQual - -0.624820

Following show the Correlation of the feature to the House price

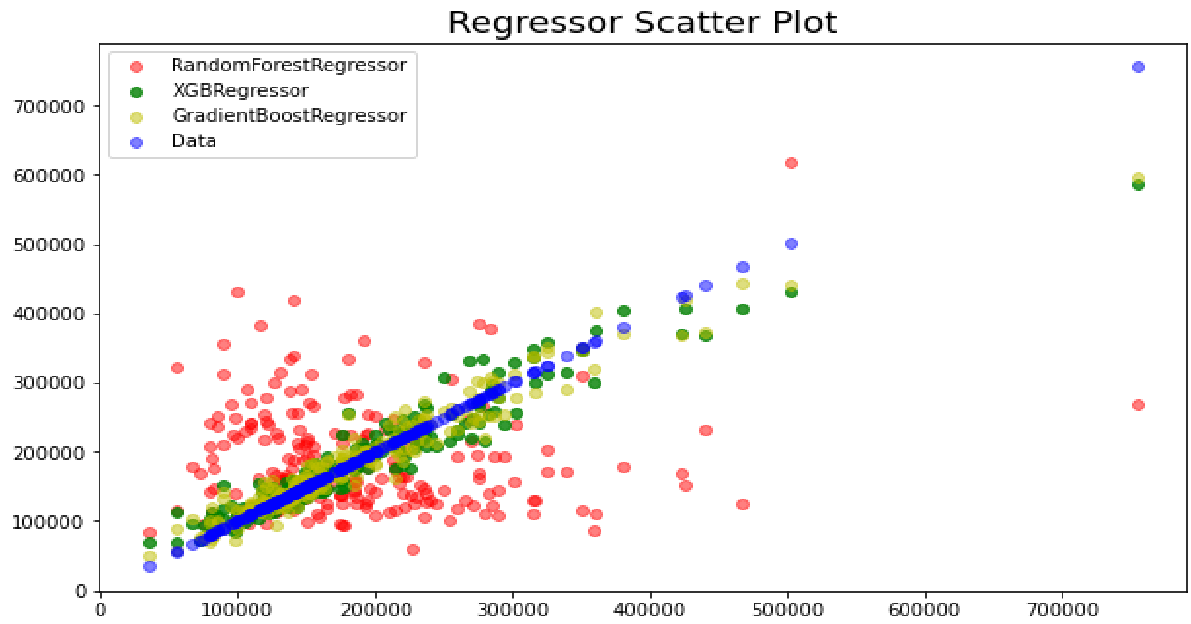
Model Visualizations

Following are the Visualizations made to view results from each model





Following are Best 2 Model which are able to predict the house prices with above 91% accuracy.



Following graph shows us the comparison between Random Forest, XG Boost , Gradient Boost Regressors where Gradient boost performed best with 93% R2 Score.

❖ Interpretation of the Results

➢ **Interpretations made from Visualization, Model building.**

➢ **Distribution of the Sale Prices**

- From the Graph we can observe that most of the houses are prices between 10,000$ - 25,000$
- There are few outliers in the data but in the interest of having diversity rows have not been dropped.

➢ **Distribution of Living Area**

- From the Scatterplot we can observe that the majority of the Dataset Contains Live Area from – 800Sq Ft – 2500Sq Ft
- Houses with Live Area greater than 3000Sq Ft are mostly priced above 20,000$.

➢ **Overall Quality to the Sale price of the house**

- From this we can observe that as the Quality rating increases the house price is also have a positive effect where the average price of house's rated 10 is above 40,000$.
- Houses Rated 1-2 with OverallQual as priced less than 10,000$.

- **Year Built to the Sale Price of the house and Livearea**

Observations made from comparing both the Graphs

- Older Houses are mostly under 20,000$ with few exceptions – which can be because of the Live area which we can observe comparing both Graphs.
- We can say that the Year of Building the house is positively affecting the price of the house even without gear difference in live area.

- **Effect of each Variable on Sale price of the house**

Positive Correlation

Features	Correlation
OverallQual	0.789185
GrLivArea	0.707300
GarageCars	0.628329
GarageArea	0.619000
TotalBsmtSF	0.595042
1stFlrSF	0.587642
FullBath	0.554988
YearBuilt	0.514408

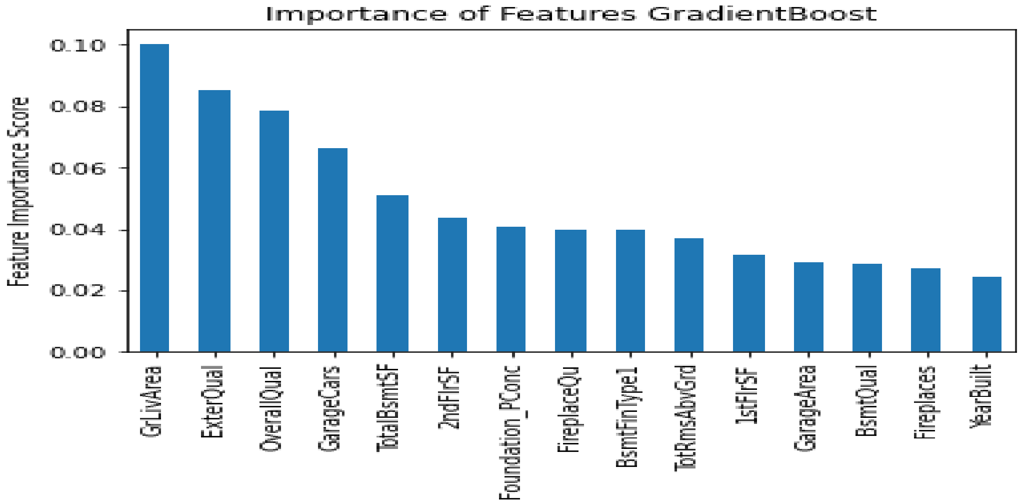
Negative Correlation

Features	Correlation
HeatingQC	-0.406604
GarageType	-0.415370
GarageFinish	-0.424922
KitchenQual	-0.592468
BsmtQual	-0.601307
ExterQual	-0.624820

Following show the Correlation of the feature to the House price

CONCLUSION

❖ Key Findings and Conclusions of the Study



- Following are Features which had a highest effect of the model in predicting the house prices.
- From the initial analysis the columns which effect the house price are mostly present in the final model as well.
- **Features - Correlation**
- OverallQual - 0.789185
- GrLivArea - 0.707300
- GarageCars - 0.628329
- GarageArea - 0.619000
- TotalBsmtSF - 0.595042
- GarageFinish - -0.424922
- KitchenQual - -0.592468
- BsmtQual - -0.601307
- ExterQual - -0.624820
- From this we can observe that as the Quality rating increases the house price is also have a positive effect where the average price of house's rated 10 is above 40,000$.
- Houses Rated 1-2 with OverallQual as priced less than 10,000$.
- Older Houses are mostly under 20,000$ with few exceptions – which can be because of the Live area which we can observe comparing both Graphs.
- From the Scatterplot we can observe that the majority of the Dataset Contains Live Area from – 800Sq Ft – 2500Sq Ft
- We are able to interpret that Gradient Boost regressor worked best in building a model to predict the house price, with an R2 score of 93.14% and Mean Absolute error of 15642.62.

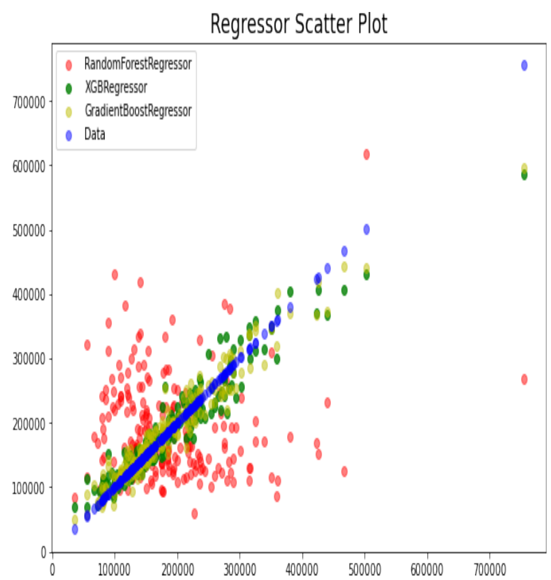
S.no	Algorithm	R2 Score	MAE(Absolute)	MSE(Squared)
1	GradientBoostRegressor	93.14	15642.27	511314033.51

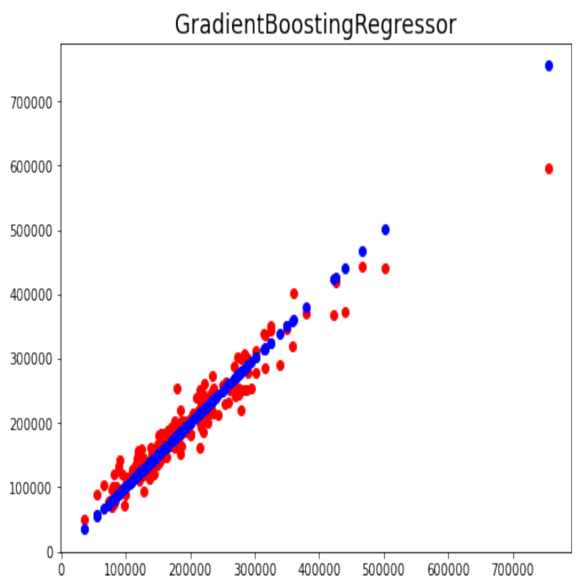
❖ Learning Outcomes of the Study in respect of Data Science

Visualization tools and the metrices used help me attain the best possible model to choose with from the various other models used.

Following are the Metrices used and the Visualization used to choose the gradient boost regressor.

S.no	Algorithm	R2 Score	MAE(Absolute)	MSE(Squared)
1	GradientBoostRegressor	92.83	15372.12	511314033.51
2	XGBRegressor	91.84	15388.96	609304374.75
3	RandomForestRegressor	89.60	19313.62	775762889.76
4	Lasso Regressor	89.8	19682.35	760473931.95
5	Linear Regressor	89.57	19896.38	777392877.26





Challenges:

- The main challenge face with this particular data set was of the missing values which has been solved by using multiple techniques like Visualization, Value counts, Unique.
- General scaling technique used which is standard scalar did not help the model in predicting the prices well. I tried without scaling the data which worked out well with better scores.
- Removal of outliers will result in a situation where we will lose few unique features so the model has been built with out removing outliers, for a reference a model built with outliers removes resulted in similar or less R2 score but with a positive of low MAE value. I Choose for the diversity in model so went ahead with data with outliers as the final model.

❖ Limitations of this work and Scope for Future Work

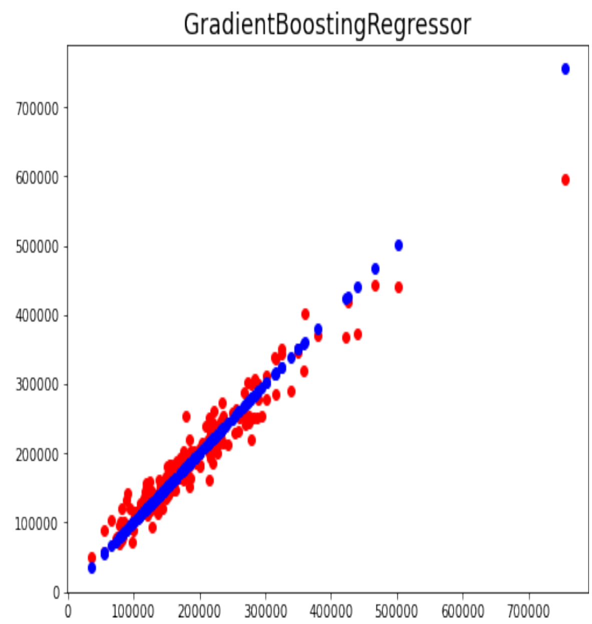
Limitations of the solution

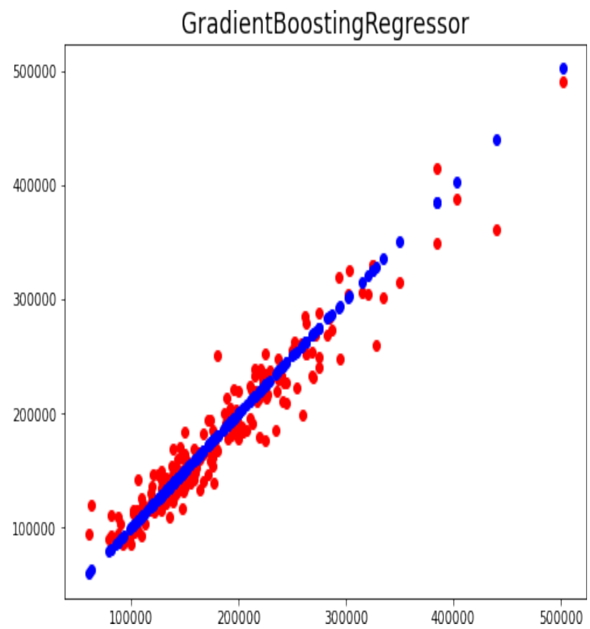
- An R2 score of 93% is Good But not perfect which can be improved with a greater number of diversified rows.
- MAE or the Final model is 15672 which is high as the Sale price of the data is from less than 10,000$ the main reason for the higher MAE is decision of diversity I took to build the model without removal of outliers.
- However, the results of the dataset with removal of outliers have been able to decrease the MAE to 13632 with a R2 score of 92.6% which is a considerable decrease in MAE with about 1800.

➢ Following is the Score of the Gradient Boost Regressor model after the outliers are removed.

```
Mean Absolute Error: 13632.91

Mean Squared Error: 351362319.88

GB Score: 92.62
```

S.no	Algorithm	R2 Score	MAE(Absolute)	MSE(Squared)
With outliers	GradientBoostRegressor	92.83	15372.27	534544208.02
With the outliers removed	GradientBoostRegressor	92.62	13632.91	351362319.88





Following are my conclusion over the improvement of the model.