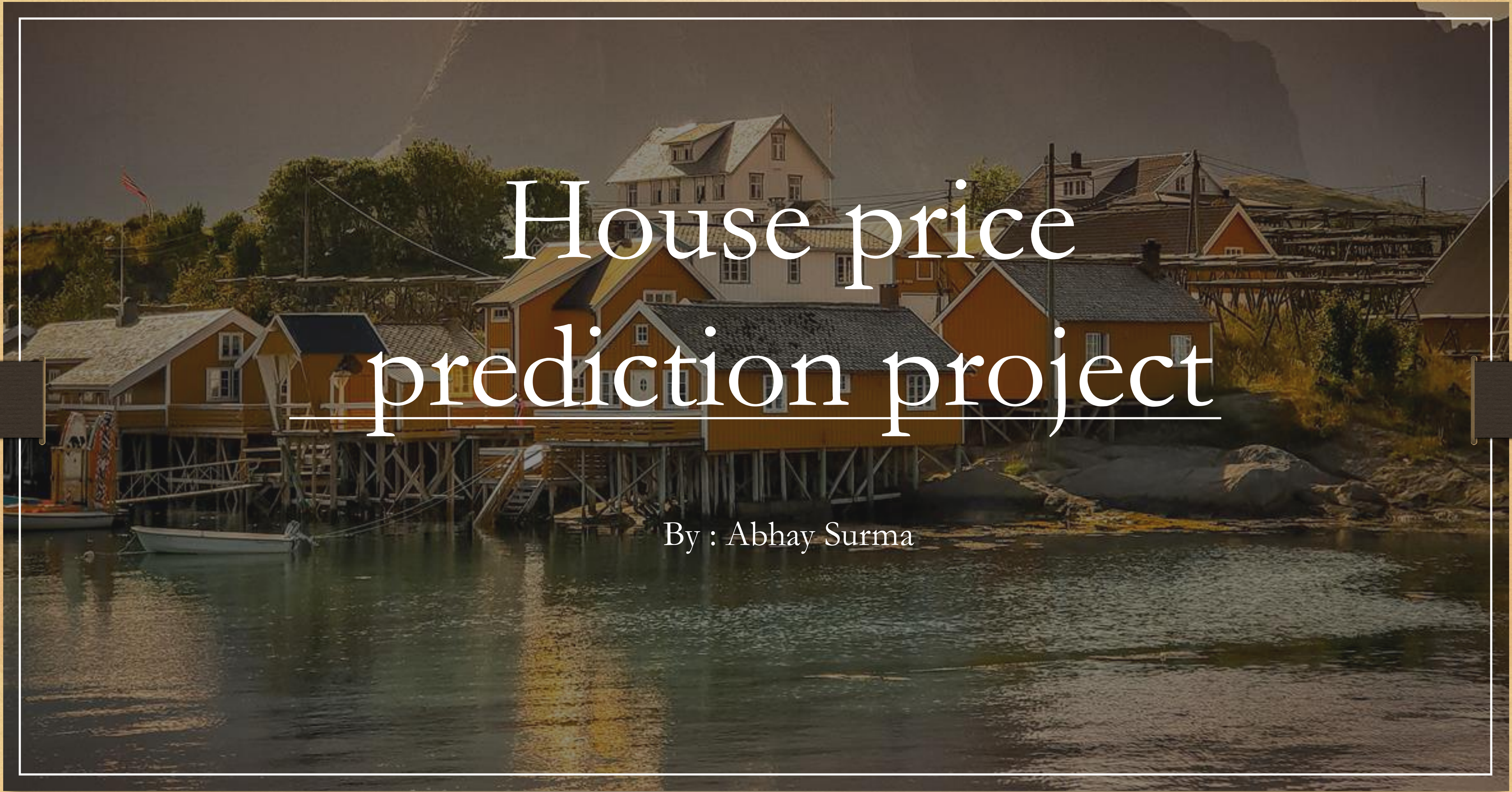


House price prediction project

By : Abhay Surma

Business Problem Framing

- Housing and Real estate Markets are one of the major contributor's in a country's economy . It is very large market and various companies are working in this domain. Data Science can play a vital role in solving problems related to this domain and can help the countries in their overall revenue , profits and improving their marketing strategies . Machine learning techniques can be used for achieving business goals for this housing companies . Our Problem is related to one of such U.S based housing company named Surprise Housing which want to enter Australian Market . The company want to use Data Analytics to purchase houses at a price below their actual values and flip them at a higher prices. The company has collected a dataset from the sale of houses in Australia . The company is looking at prospective properties to buy houses to enter the market . we will build a model using Machine Learning to predict the actual value of the prospective properties and it will help the company to decide whether to invest in property or not.

Conceptual Background of the Domain Problem

- Trends in housing prices indicate the current economic situation and also are a concern to the buyers and sellers. There are many factors that have an impact on house prices, such as the number of bedrooms and bathrooms. House price depends upon its location as well. A house with great accessibility to highways, schools, malls, employment opportunities, would have a greater price as compared to a house with no such accessibility. Predicting house prices manually is a difficult task and generally not very accurate, hence there are many systems developed for house price prediction.

Undertaken

- Growing unaffordability of housing has become one of the major challenge for countries around the world. In order to gain a better understanding of the commercialized housing market we are currently facing , we want to figure out what are the top influential factors of the housing price. Apart from the more obvious driving forces such as the inflation and the scarcity of land, there are also a number of variable that are worth looking into. Therefore , we choose to study thr house price prediction., which enables us to dig into the variables in depth and to provide a model that could more accurately estimate house prices. In this way, people could make better decision when it comes to home investment.
- Our objective is to discuss the major factors that effect housing price and make precise prediction for it. We use 80 explanatory variables including almost every aspect of residential homes in Australia. Methods of both statistical , regression models and machine learning models are applied and firther compared according to their performance to better estimate the final price of each house. The model provides price prediction based on similar comparable of people’s dream house, which allow both buyers and sellers to better negotiate home prices according to market treand.

Loading Necessary libraries and Train and test dataset

Now we will check the columns and shape of the dataset Simultaneously with the same process we will check our test data set,

1168 rows and 81 columns - train data set
and 292 rows and 80 columns - test data set.

Importing data

```
data = pd.read_csv('train.csv')
```

```
test_data = pd.read_csv('test.csv')
```

```
data.head()
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Cond
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	

```
data.dtypes
```

```
Id                int64
MSSubClass        int64
MSZoning         object
LotFrontage     float64
LotArea           int64
Street           object
Alley            object
LotShape         object
LandContour      object
Utilities        object
```

Let’s check null in dataset

i) Nan value can't be imported for columns -

- Alley, FireplaceQu, PoolQC, Fence, MiscFeature

ii) other columns which are having less nan value and we can import values for that column by using any imputer technique -

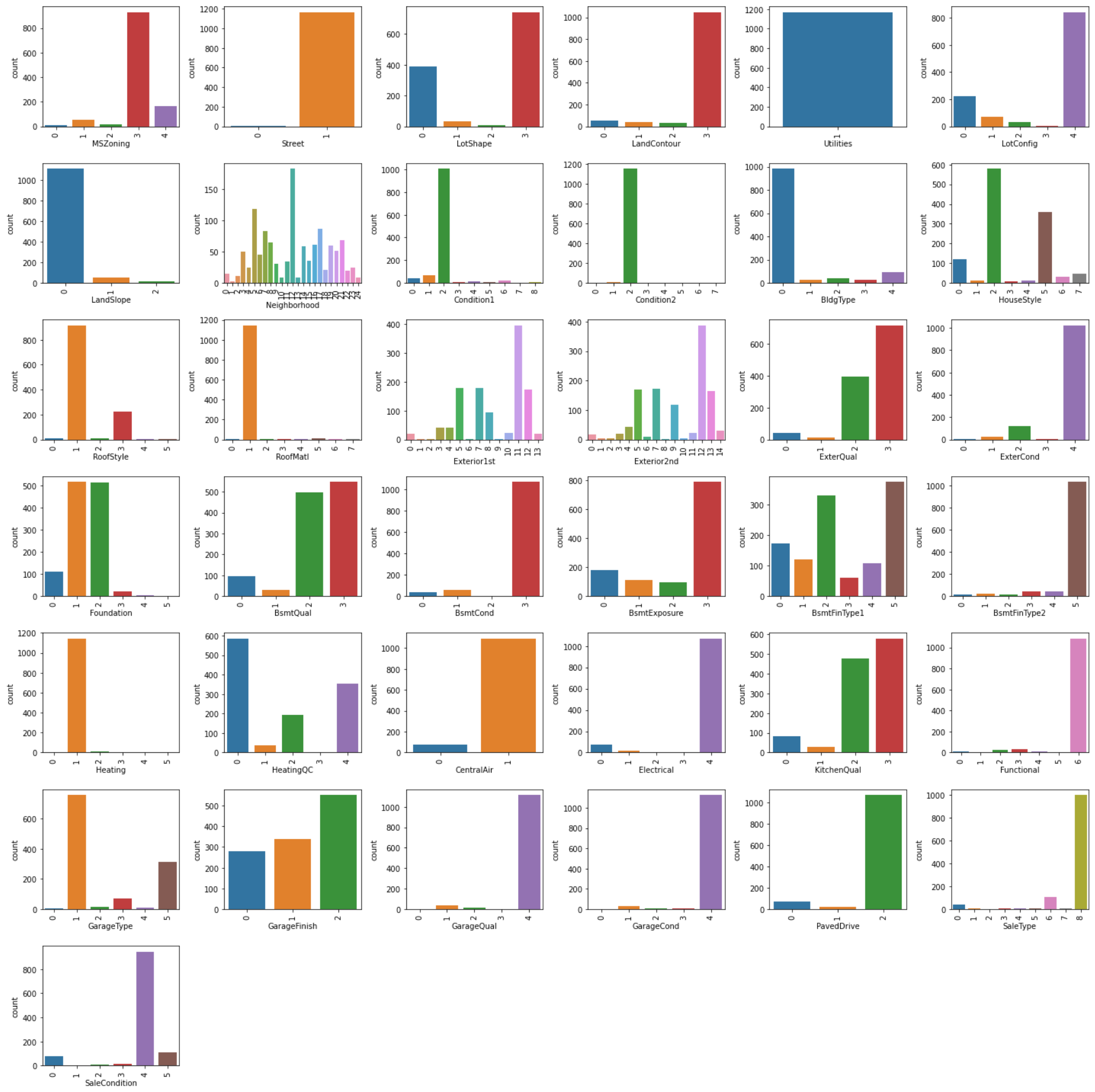
- LotFrontage
- MasVnrType
- MasVnrArea
- BsmtQual
- BsmtFinType1
- BsmtFinType2
- GarageType
- GarageYrBlt
- GarageFinish
- GarageQual
- GarageCond

```
features_with_na = [features for features in dataset.co
for feature in features_with_na:
  print(feature,np.round(dataset[feature].isnull().mea
  #printing percentage of missing values
```

```
LotFrontage 0.1832
Alley 0.9341
MasVnrType 0.006
MasVnrArea 0.006
BsmtQual 0.0257
BsmtCond 0.0257
BsmtExposure 0.0265
BsmtFinType1 0.0257
BsmtFinType2 0.0265
FireplaceQu 0.4717
GarageType 0.0548
GarageYrBlt 0.0548
GarageFinish 0.0548
GarageQual 0.0548
GarageCond 0.0548
PoolQC 0.994
Fence 0.7971
MiscFeature 0.9623
```

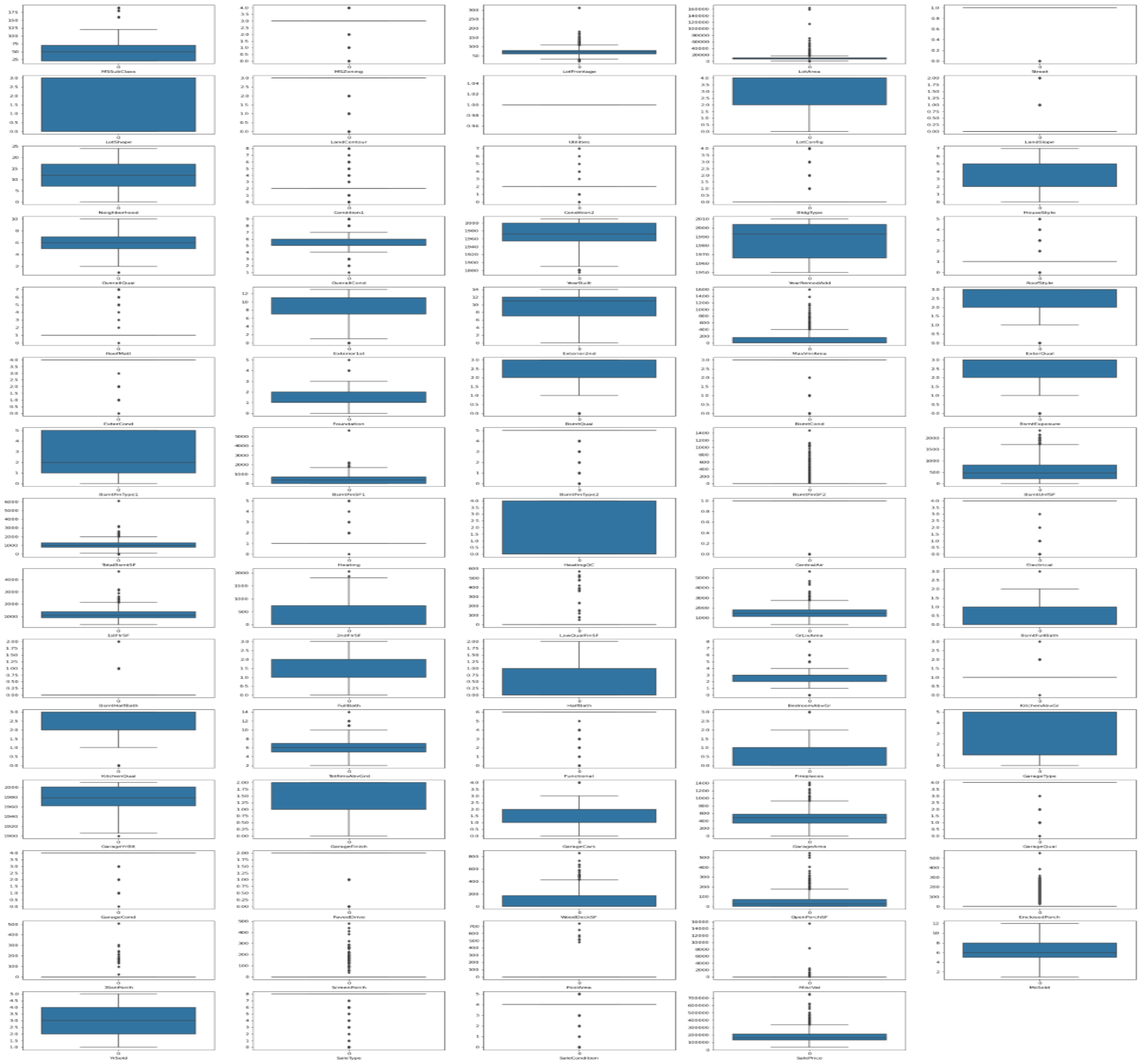
Visualization

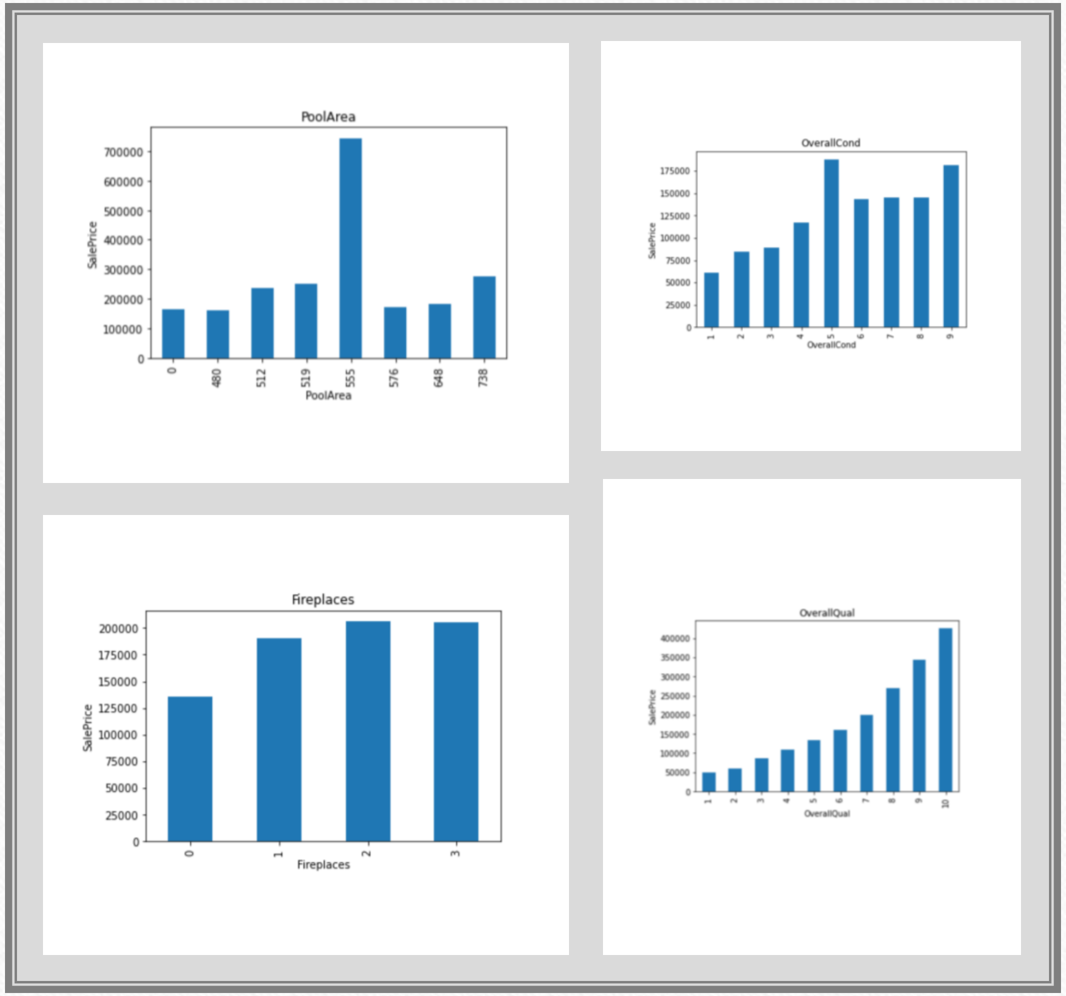
Univariate Analysis:



OUTLIERS -

We can't remove the outliers we don't have the good amount of data as outliers are 13% of data we will not drop it.



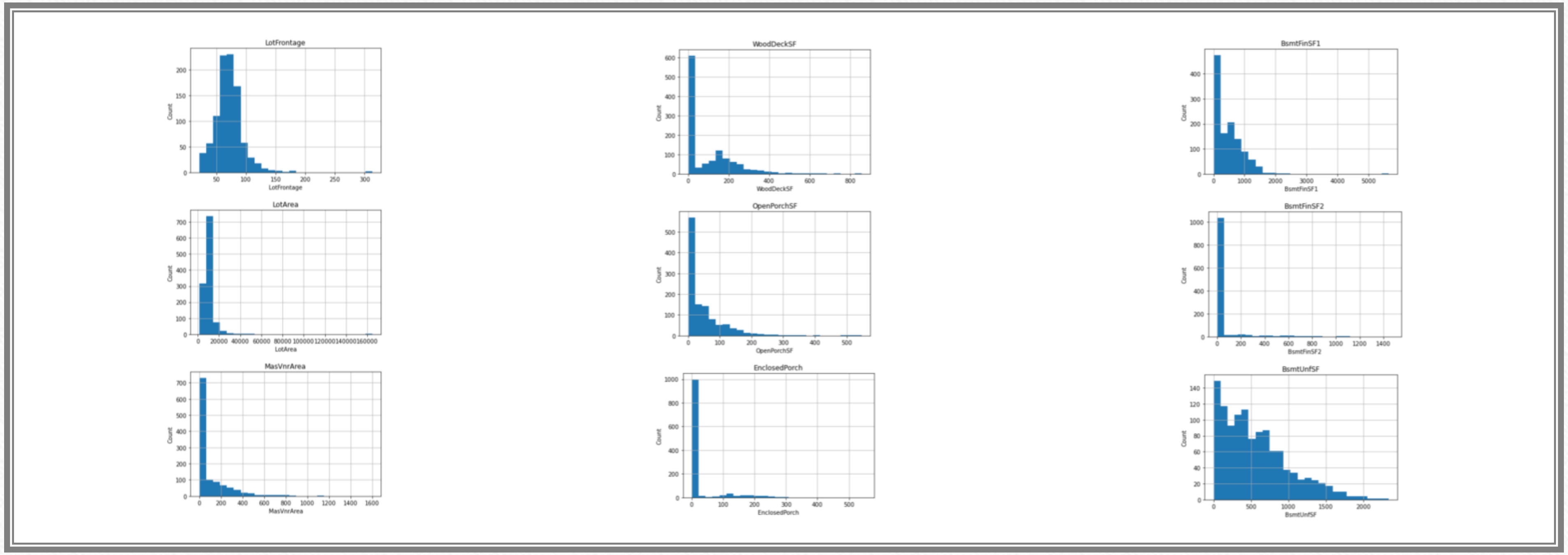


As we can see the overall quality is increasing the price is high.

If the overall condition of the house is average the price is more that others.

As there is increase in no of fire places the price of house is rising.

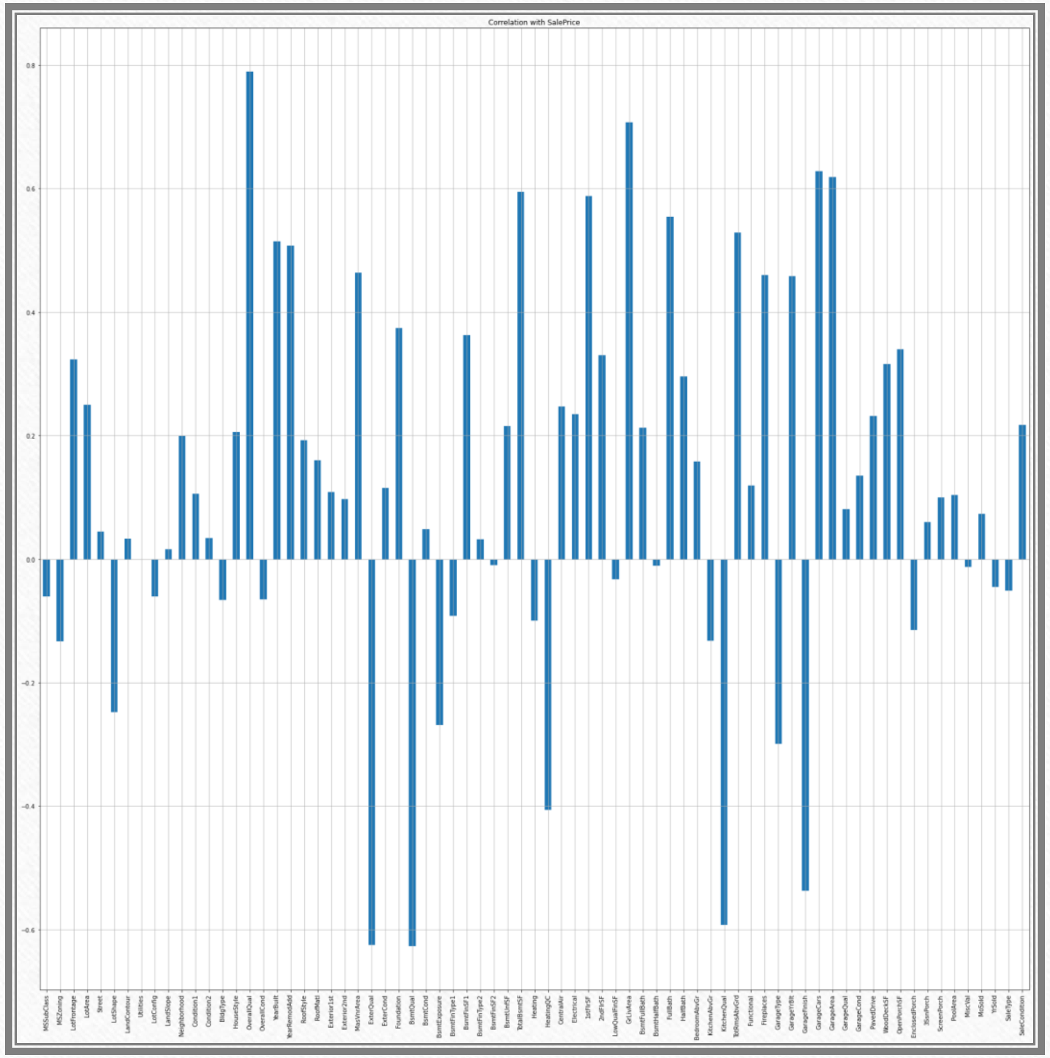
As the house has a pool area of 555 sq feet the price of the house is more as compared to others.



Here you can see that the data is skewed so we will perform power transformation to reduced skewness of the data.

Statistical Analysis

- In this we found that Variables like OverallQual (overall material and finish of the house) , Year Built , TotRmsAbvGrd (Total rooms above grade (does not include bathrooms) , GarageCars (Size of garage in car capacity) ,GarageArea (Size of garage in square feet) , GrLivArea (Above grade (ground) living area square feet) , FullBath (Full bathrooms above grade) have positive relationship with the sales Price.
- YearBuilt, YearRemodAdd, GarageYrBuilt are negatively related with sale price



Data preprocessing of train data and test data

- First of all we will drop columns Alley , MiscFeature , PoolQC , Fence and GarageYrBlt because more than 80 % data in these columns are missing if we replace these missing data with some data it can give us wrong prediction in the final model thus making our model less effective so better to drop these columns.

The data of the categorical columns needs to be encoded using LabelEncoder

```
features = ['MSZoning', 'Street',
       'LotShape', 'LandContour', 'Utilities', 'LotConfig', 'LandSlope',
       'Neighborhood', 'Condition1', 'Condition2', 'BldgType', 'HouseStyle','RoofStyle',
       'RoofMatl', 'Exterior1st', 'Exterior2nd', 'ExterQual',
       'ExterCond', 'Foundation', 'BsmtQual', 'BsmtCond', 'BsmtExposure',
       'BsmtFinType1', 'BsmtFinType2', 'Heating', 'HeatingQC', 'CentralAir', 'Electrical',
       'KitchenQual', 'Functional', 'GarageType', 'GarageFinish', 'GarageQual',
       'GarageCond', 'PavedDrive','SaleType', 'SaleCondition']
```

```
from sklearn.preprocessing import LabelEncoder
labenc = LabelEncoder()
for col in data[data.columns[data.dtypes == 'object']]:
    data[col] = labenc.fit_transform(data[col])
data['YrSold'] = data.YrSold.map({2007:2,2009:4,2006:1, 2008: 3, 2010: 5}) # encoding years in YrSold Column
data['Utilities'] =data.Utilities.map({0:1})
data.dtypes[data.dtypes != 'object']
```

```
MSSubClass        int64
MSZoning          int32
LotFrontage     float64
LotArea           int64
Street            int32
```

Data preprocessing of train data and test data

- KNN imputer for continous data
- Simple imputer for categorical data

```
from sklearn.impute import KNNImputer
imp = KNNImputer(n_neighbors=3)
data[['LotFrontage']] = imp.fit_transform(data[['LotFrontage']])
data[['LotFrontage']].isnull().sum()
data[['GarageYrBlt']] = imp.fit_transform(data[['GarageYrBlt']])
data[['MasVnrArea']] = imp.fit_transform(data[['MasVnrArea']])
```

```
imp = KNNImputer(n_neighbors=3)
test_data[['LotFrontage']] = imp.fit_transform(test_data[['LotFrontage']])
test_data[['LotFrontage']].isnull().sum()
test_data[['GarageYrBlt']] = imp.fit_transform(test_data[['GarageYrBlt']])
test_data[['MasVnrArea']] = imp.fit_transform(test_data[['MasVnrArea']])
```

imputing mode by simple imputer

```
from sklearn.impute import SimpleImputer
si = SimpleImputer(missing_values = np.nan,strategy = 'most_frequent',verbose = 0 )
si = si.fit(data[['MasVnrType','Electrical','MasVnrArea','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2',
data[['MasVnrType','Electrical','MasVnrArea','BsmtQual','BsmtCond','BsmtExposure','BsmtFinType1','BsmtFinType2','GarageType'
data.isnull().sum()
```

```
MSSubClass      0
```

Data Scaling - Using power transformer method –" yeo johnson"

```
features=['LotFrontage','LotArea','MasVnrArea','BsmtFinSF1','BsmtFinSF2','BsmtUnfSF','TotalBsmtSF','1stFlrSF','2ndFlrSF','Gr
```

```
from sklearn.preprocessing import PowerTransformer
pt = PowerTransformer()
data[features]  = pt.fit_transform(data[features].values)
data.head()
```

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	Bld
0	120	3	0.093658	-1.213954	1	0	3	1	4	0	13	2	2	
1	20	3	1.117135	1.100521	1	0	3	1	4	1	12	2	2	
2	60	3	0.998803	0.158048	1	0	3	1	1	0	15	2	2	
3	20	3	1.495566	0.496002	1	0	3	1	4	0	14	2	2	
4	20	3	0.093658	1.196626	1	0	3	1	2	0	14	2	2	

```
from sklearn.preprocessing import PowerTransformer
pt = PowerTransformer()
test_data[features]  = pt.fit_transform(test_data[features].values)
test_data.head()
```

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	Utilities	LotConfig	LandSlope	Neighborhood	Condition1	Condition2	Bld
0	20	2	0.982253	0.842656	1	0	1	1.0	0	0	21	2	0	
1	120	2	0.035790	-0.739104	1	0	3	1.0	1	0	21	2	0	
2	20	2	0.035790	0.524304	1	3	3	1.0	4	0	4	2	0	
3	70	2	0.457389	0.548484	1	3	0	1.0	4	0	5	2	0	
4	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	

Selecting K – best features -

```
### Selecting K best features
```

```
from sklearn.feature_selection import SelectKBest, f_classif
```

```
bestfeat = SelectKBest(score_func = f_classif, k = 'all')
fit = bestfeat.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
```

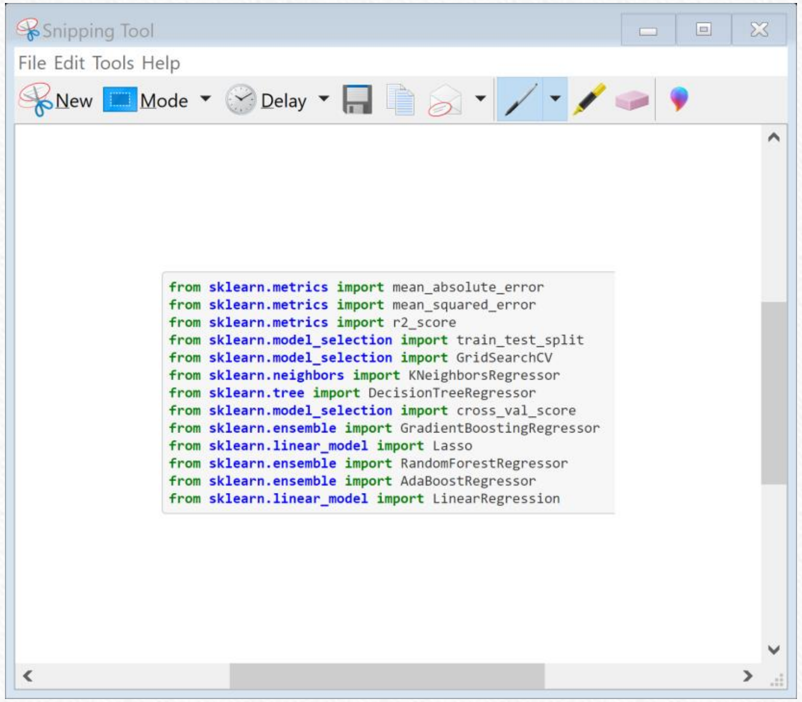
```
fit = bestfeat.fit(X,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
dfcolumns.head()
featureScores = pd.concat([dfcolumns,dfscores],axis = 1)
featureScores.columns = ['Feature', 'Score']
print(featureScores.nlargest(75,'Score'))
```

	Feature	Score
13	OverallQual	5.303071
62	MiscVal	3.564855
22	ExterQual	3.514221
25	BsmtQual	2.876879
45	KitchenQual	2.617125
51	GarageCars	2.578547
41	FullBath	2.435854
52	GarageArea	2.242545
50	GarageFinish	2.187163
15	YearBuilt	2.133300
33	TotalBsmtSF	2.070692
2	Street	1.835751

Model/s Development and Evaluation

- As we know that it is a Regression problem in which our output is 'Sale Price' so we will use the regression model like linear Regression , , Gradient Boosting Regressor , Random Forest Regressor , Xgboost we will train our training data using these algorithms and then we will test on the finalised test data set for final house price prediction . The algorithm which is giving better accuracy and Cross value score will be chosen as final model.

First we will load the necessary libraries .



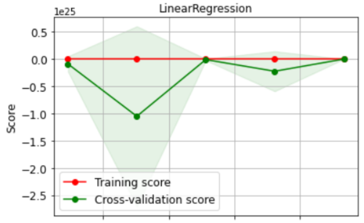

```python
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.linear_model import Lasso
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import AdaBoostRegressor
from sklearn.linear_model import LinearRegression
```

Random_state =522

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=0.10,random_sta
```

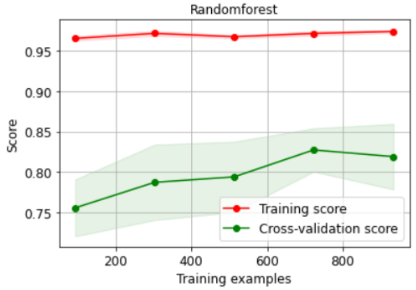
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error
lr = LinearRegression()
lr.fit(x_train,y_train)
pred = lr.predict(x_test)
print(r2_score(y_test,pred)*100)
print("mean_absolute_error",mean_absolute_error(y_test,pred))
mse = mean_squared_error(y_test,pred)
print('RMSE :', np.sqrt(mse))
skplt.estimators.plot_learning_curve(lr,X,y,cv=5,scoring='r2',text_fontsize='l
print(plt.show())
```

```
82.22988770136858
mean_absolute_error 24800.1660667033
RMSE : 39938.05691815641
```




```
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor()
rf.fit(x_train,y_train)
pred = rf.predict(x_test)
print(r2_score(y_test,pred)*100)
print(mean_absolute_error(y_test,pred))
mse = mean_squared_error(y_test,pred)
print('RMSE :', np.sqrt(mse))
skplt.estimators.plot_learning_curve(rf,X,y,cv=5,scoring='r2',text_
print(plt.show())
```

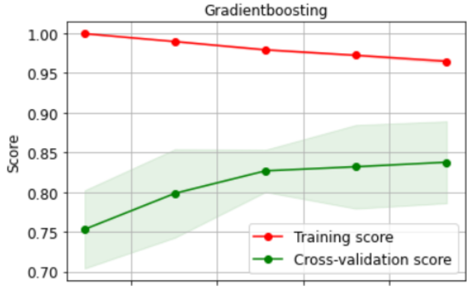
```
87.50108664651577
20549.342905982907
RMSE : 33494.84513194701
```



None

```
from sklearn.ensemble import GradientBoostingRegressor
gb = GradientBoostingRegressor()
gb.fit(x_train,y_train)
pred = gb.predict(x_test)
print(r2_score(y_test,pred)*100)
print(mean_absolute_error(y_test,pred))
mse = mean_squared_error(y_test,pred)
print('RMSE :', np.sqrt(mse))
skplt.estimators.plot_learning_curve(gb,X,y,cv=5,scoring='
print(plt.show())
```

```
88.59447423453717
20395.02989545487
RMSE : 31996.28086397267
```



Now we will find best Algorithm for our model

Best model selected by cross validation,R2- score and RMSE Value is Gradient boosting

Hypertuning the model by GridsearchCV

```
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import GridSearchCV
gbr=GradientBoostingRegressor()
x_train,x_test,y_train,y_test = train_test_split(X,y,test_size=.10,random_state=522)
params = {'n_estimators':[100,150],
        'min_samples_split':[2,6],
        'min_samples_leaf':[1,5],
          'learning_rate': np.arange(0.1,0.5,0.1)
        }
grd = GridSearchCV(gbr,param_grid = params)
grd.fit(x_train,y_train)
print('best_pram', grd.best_params_)

rf=grd.best_estimator_ #reinstantiating with best params

rf.fit(x_train,y_train)
pred = rf.predict(x_test)
print(r2_score(y_test,pred)*100)
print(mean_absolute_error(y_test,pred))
mse = mean_squared_error(y_test,pred)
print('RMSE :', np.sqrt(mse))
```

```
best_pram {'learning_rate': 0.1, 'min_samples_leaf': 5, 'min_samples_split': 2, 'n_estimators': 150}
87.99887426171998
20277.413048933617
RMSE : 32821.07778455571
```

```python
from sklearn.ensemble import GradientBoostingRegressor
gb = GradientBoostingRegressor(learning_rate=0.15,min_samples_leaf=1,min_samples_split=2,n_estimators=150,max_depth=5,max_fe
gb.fit(x_train,y_train)
pred = gb.predict(x_test)
print(r2_score(y_test,pred)*100)
print(mean_absolute_error(y_test,pred))
mse = mean_squared_error(y_test,pred)
print('RMSE :', np.sqrt(mse))
```

```
91.37720676472144
18535.35613679279
RMSE : 27820.557689028996
```

Best parameters and model selected with it

Now we will predict the sale price for our Test data

```
Out[91]: 0       129496.528725
         1       269888.651048
         2       259546.271793
         3       191102.211020
         4       215940.989701
         5       222576.247267
         6       128036.837735
         7       156339.858089
         8       139828.660292
         9       115303.935787
         10      120652.017724
         11      236279.319459
         12      203767.127903
         13      127454.513070
         14      139662.679025
         15      165243.836145
         16      119604.687454
         17      192318.594570
         18      151892.623458
```

Values of test data

CONCLUSION

Key Findings and Conclusions of the Study

Based on the in-depth analysis of the Housing Project,

The Exploratory analysis of the datasets, and the analysis of the Outputs of the models the following observations are made:

Structural attributes of the house Structural attributes of the house like lot size, lot shape, quality and condition of the house, garage capacity, rooms, Lot frontage, number of bedrooms, bathrooms, overall finishing of the house etc play a big role in influencing the house price.

CONCLUSION

Key Findings and Conclusions of the Study

- Due to the Training dataset being very small, the outliers had to be retained for proper training of the models.

- Therefore, Gradient boosting Regressor, being robust to outliers and being indifferent to non-linear features, performed well despite having to work on small dataset.

Learning Outcomes of the Study in respect of Data Science

- The goal is to achieve the system which will reduce the human effort to find a house having reasonable price. The proposed system. House Price Prediction model approximately try to achieve the same one. Proposed system focused on predict the house price according to the area for that image processing and machine learning methods are used. The experimental results showed that this technique that are used while developing system will give accurate prediction of house price