FLIP ROBO

Housing:Price Prediction

NAME OF THE PROJECT

Submitted by:

Salman Pradhan

ACKNOWLEDGMENT

I wish to thank Flip Robo Technologies for assigning this project to me and thanks to Mr.Shubham Yadav sir for his valuable technical support on this project.

INTRODUCTION

- Business Problem Framing

  We have to build a model using Machine Learning in order to predict the actual value of the prospective properties and decide whether to invest in them or not.

  This model will then be used by the management to understand how exactly the prices vary with the variables. They can accordingly manipulate the strategy of the firm and concentrate on areas that will yield high returns

- Conceptual Background of the Domain Problem

  Created model can be used to predict house prices, it might be a good tool for Real Estate Investors who might look into house price predictions before buying a house and making an investment, which might help them to save or earn money. Single house buyers also can used model before purchasing a house in order to know if they are buying not overpriced house for their area. As we know real estate market is blooming, which means that property prices are very high.

- Review of Literature

  As realestate business is a profitable business now a days,investors want to invest their money in this domain.

- Motivation for the Problem Undertaken

  DataScience help us to make predictions at areas like health sectors, auto industry, education, media etc. For our project we decided to implement Prediction of House Prices.we have to create a model which will be used in Real Estate Investment.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  The prediction will base on the given dataset which is the training set. After training the machine, it can predict the final price of each home..

- Data Sources and their formats

  We have two data set files. One is train.cvs which store data for training and testing our model. And also we have test.cvs in which we have to predict our output feature(SalePrice).

train.csv

test.csv

In train.csv we have 1168 rows and 81 features and in our test.csv we have 292 rows and 80 features.Our predicted output feature is ‘SalePrice’.

Here's a brief version of what we'll find in the data description file.

1. Id.
2. MSSubClass: The building class
3. MSZoning: The general zoning classification
4. LotFrontage: Linear feet of street connected to property
5. LotArea: Lot size in square feet
6. Street: Type of road access
7. Alley: Type of alley access
8. LotShape: General shape of property
9. LandContour: Flatness of the property
10. Utilities: Type of utilities available
11. LotConfig: Lot configuration
12. LandSlope: Slope of property
13. Neighborhood: Physical locations within Ames city limits
14. Condition1: Proximity to main road or railroad
15. Condition2: Proximity to main road or railroad (if a second is present)
16. BldgType: Type of dwelling
17. HouseStyle: Style of dwelling
18. OverallQual: Overall material and finish quality
19. OverallCond: Overall condition rating
20. YearBuilt: Original construction date
21. YearRemodAdd: Remodel date
22. RoofStyle: Type of roof
23. RoofMatl: Roof material
24. Exterior1st: Exterior covering on house
25. Exterior2nd: Exterior covering on house (if more than one material)
26. MasVnrType: Masonry veneer type
27. MasVnrArea: Masonry veneer area in square feet

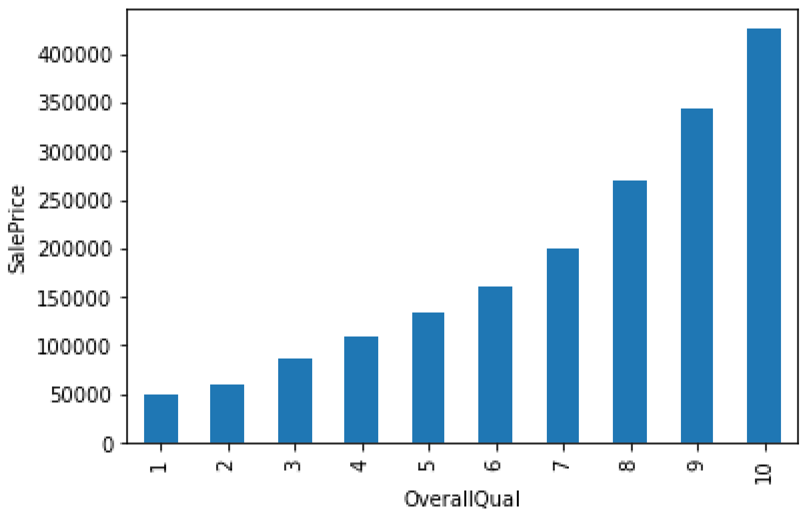
28. ExterQual: Exterior material quality
29. ExterCond: Present condition of the material on the exterior
30. Foundation: Type of foundation
31. BsmtQual: Height of the basement
32. BsmtCond: General condition of the basement
33. BsmtExposure: Walkout or garden level basement walls
34. BsmtFinType1: Quality of basement finished area
35. BsmtFinSF1: Type 1 finished square feet
36. BsmtFinType2: Quality of second finished area (if present)
37. BsmtFinSF2: Type 2 finished square feet
38. BsmtUnfSF: Unfinished square feet of basement area
39. TotalBsmtSF: Total square feet of basement area
40. HeatingQC: Heating quality and condition
41. CentralAir: Central air conditioning
42. Electrical: Electrical system
43. 1stFlrSF: First Floor square feet
44. 2ndFlrSF: Second floor square feet
45. LowQualFinSF: Low quality finished square feet (all floors)
46. GrLivArea: Above grade (ground) living area square feet
47. BsmtFullBath: Basement full bathrooms
48. BsmtHalfBath: Basement half bathrooms
49. FullBath: Full bathrooms above grade
50. HalfBath: Half baths above grade
51. Bedroom: Number of bedrooms above basement level
52. Kitchen: Number of kitchens
53. KitchenQual: Kitchen quality
54. TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)
55. Functional: Home functionality rating
56. Fireplaces: Number of fireplaces
57. FireplaceQu: Fireplace quality
58. GarageType: Garage location
59. GarageYrBlt: Year garage was built
60. GarageFinish: Interior finish of the garage
61. GarageCars: Size of garage in car capacity
62. GarageArea: Size of garage in square feet
63. GarageQual: Garage quality

64. GarageCond: Garage condition
65. PavedDrive: Paved driveway
66. WoodDeckSF: Wood deck area in square feet
67. OpenPorchSF: Open porch area in square feet
68. EnclosedPorch: Enclosed porch area in square feet
69. 3SsnPorch: Three season porch area in square feet
70. ScreenPorch: Screen porch area in square feet
71. PoolArea: Pool area in square feet
72. PoolQC: Pool quality
73. Fence: Fence quality
74. MiscFeature: Miscellaneous feature not covered in other categories
75. MiscVal: $Value of miscellaneous feature
76. MoSold: Month Sold
77. YrSold: Year Sold
78. SaleType: Type of sale
79. SaleCondition: Condition of sale
80. SalePrice - the property's sale price in dollars. This is the target variable that you're trying to predict.

- ## Data Analysis

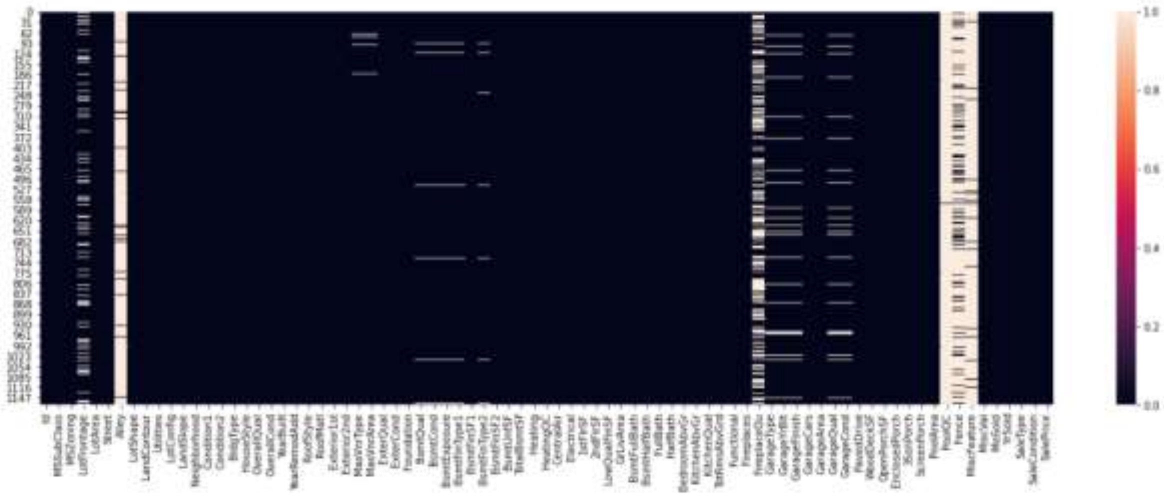
  First we have to check relationship of categorical features with SalePrice.

  OverallQual is directly varies with SalePrice.



Second, we need to do some pretreatment to our train.csv data.As we see there are so many null values present in the dataset which are unavoidable. To solve this, we can take some measures to remove or replace the null values. After cleaning the data, model will be easier to fit and we will get a better result.

```
plt.figure(figsize=(22,7))
sns.heatmap(df.isnull())
plt.show()
```

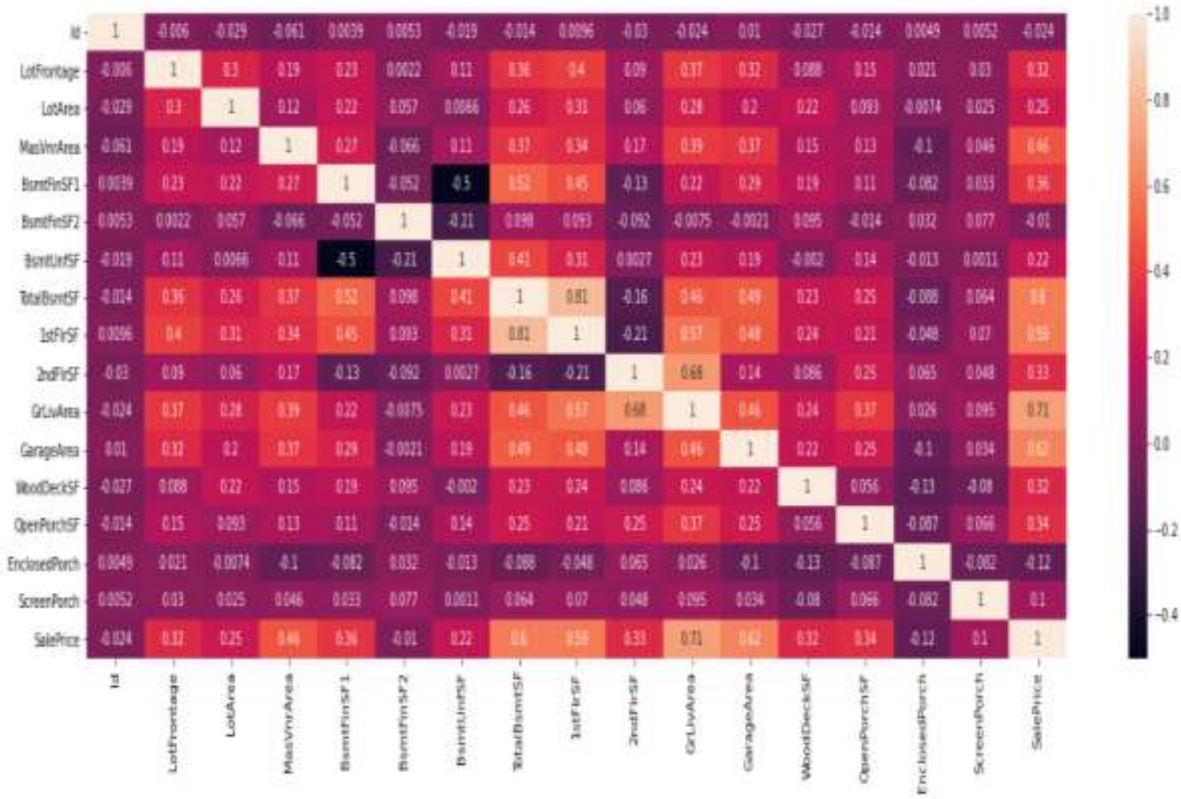


We have to replace null values of catagorical features like this

```
df['Alley'].fillna('NA',inplace=True)#replace null values with NA as no alley acess
df['MasVnrType'].fillna('None',inplace=True)#replace null values with None
df['BsmtQual'].fillna('TA',inplace=True)#replace null values with mode of the column
df['BsmtCond'].fillna('TA',inplace=True)#replace null values with mode of the column
df['BsmtExposure'].fillna('No',inplace=True)#replace null values with mode of the column
df['BsmtFinType1'].fillna('Unf',inplace=True)#replace null values with mode of the column
df['BsmtFinType2'].fillna('Unf',inplace=True)#replace null values with mode of the column
df['FireplaceQu'].fillna('NA',inplace=True)#replace null values with NA(as not available) of the column
df['GarageType'].fillna('NA',inplace=True)#replace null values with NA(as not available) of the column
df['GarageFinish'].fillna('NA',inplace=True)#replace null values with NA(as garrage not available) of the column
df['GarageQual'].fillna('NA',inplace=True)#replace null values with NA(as garrage not available) of the column
df['GarageCond'].fillna('NA',inplace=True)#replace null values with NA(as garrage not available) of the column
df['PoolQC'].fillna('NA',inplace=True)#replace null values with NA(as pool not available) of the column
df['Fence'].fillna('NA',inplace=True)#replace null values with NA(as Fence not available) of the column
df['MiscFeature'].fillna('NA',inplace=True)#replace null values with NA(as not available)
```

We replace numerical continious feature by their mean and GarageYrBlt column with the year the house was built.
After removing all the null values we have to check corelation of independent features with SalePrice

```
plt.figure(figsize=(22,7))
sns.heatmap(df[continious].corr(),annot=True)
plt.show()
```

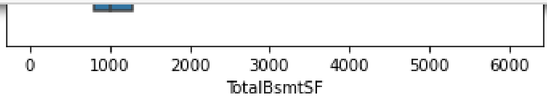


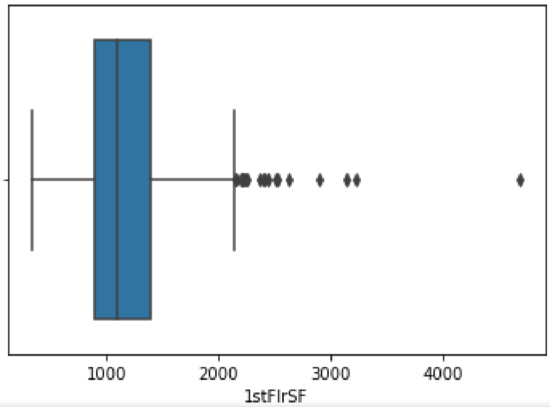
As we clearly see that GrLivArea,GarageArea is highly positively corelated to SalePrice.

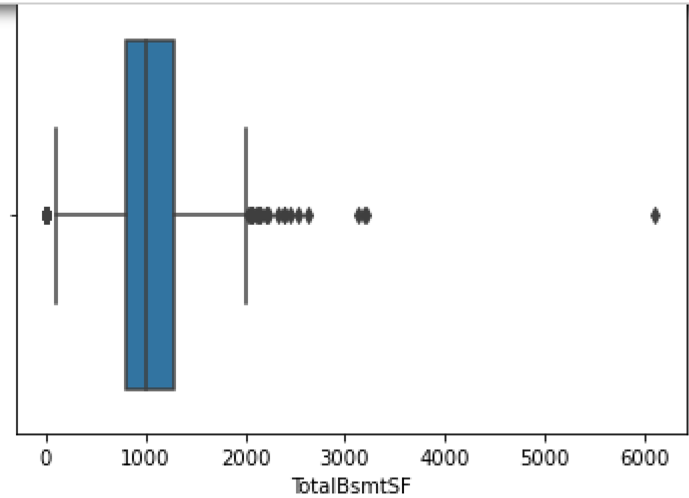
- ## Data Preprocessing Done

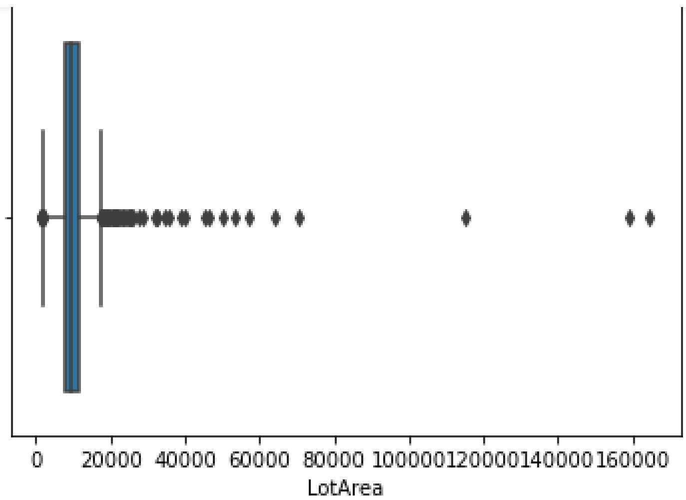
First of all We have to remove outlier.As we can see outliers directly by using boxplot in numerical continuous features.After detecting outliers we have to remove it

```
for i in df[continious].columns:
    sns.boxplot(df[continious][i])
    plt.show()
```







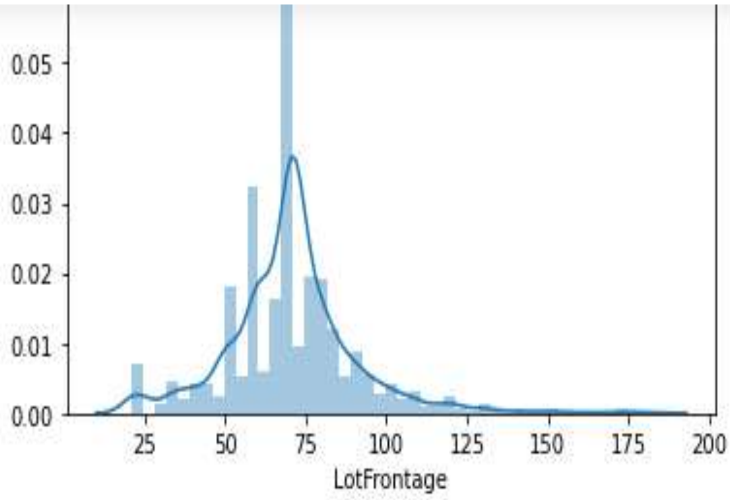


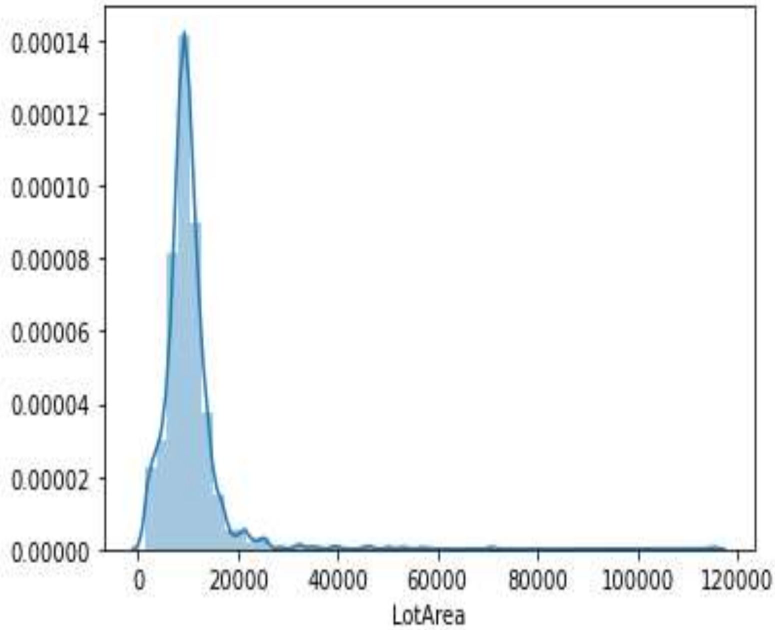
After detecting outliers we have to remove it for better accuracy of model.

```python
#So we have to drop these outlier rows
df=df.drop([592,1038,1053,1123],axis=0)#drop outlier rows in the dataset
df=df.reset_index()#reindexing dataset
df=df.drop('index',axis=1)#drop index column of dataset
df
```

After removing outliers dataloss is 0.34%.so its negligible as compared to dataset

Now our next step is to check skewness of numerical continuous columns.If some columns have skewness ness,we have to remove it by using power_transform or log transform technique





```
df[['LotFrontage','LotArea','1stFlrSF','GrLivArea','SalePrice']].skew()
```

```
LotFrontage    0.83754
LotArea        6.79545
1stFlrSF       1.01825
GrLivArea      1.15780
SalePrice      1.96065
dtype: float64
```

Some skewness is present .so we have to remove skewness by using power transform method and logarithmic transform

```
from sklearn.preprocessing import power_transform
df['LotFrontage']=power_transform(df[['LotFrontage']])
df['GrLivArea']=power_transform(df[['GrLivArea']])
```

```
df['LotArea']=np.log(df['LotArea'])
df['1stFlrSF']=np.log(df['1stFlrSF'])
```

Then we have to normalize independent features of our dataset by using standardscaler or minmaxscaler.Then our dataset is ready for model building.

```
from sklearn.preprocessing import MinMaxScaler#scaling independent columns
scaler=MinMaxScaler()
x=scaler.fit_transform(x)
```

We have to take 90% of our train dataset for model building and take other 10% for looking accuracy in the dataset.

After choosing best model our next target is to apply on test.csv dataset to predict saleprice.We have to do data cleaning,datapreprocessing and feature scaling to make our dataset ready.After doing all we predict the saleprice of our dataset.

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

I have considerd null values of Garageyrblt column with the year house was built and most of null values of categorical column have replaced by ‘NA’ as given by datadescription.txt file.

- ## Hardware and Software Requirements and Tools Used

- **NumPy**: Base n-dimensional array package
- **Matplotlib**: Comprehensive 2D/3D plotting
- **Seaborn**: For plotting graph

- **Pandas**: Data structures and analysis
- **SciPy**: Fundamental library for scientific computing
- **Scikit-learrn**: provides a range of algorithm

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

1.Understand business problem
2.Get data
3.Data analysis
4.Data Cleaning
5.Visualization with SalePrice
6.Data preprocessing
7.Feature scaling
8.Model building

- Testing of Identified Approaches (Algorithms)

1.LinearRegression
2.Lasso
3.Ridge
4.ElasticNet
5.RandomForestRegressor
6.ExtraTreesRegressor
7.GradientBoostingRegressor
8.XGBRegressor

- ## Run and Evaluate selected models

We have to first choose best random_state for models.

```
]: from sklearn.model_selection import train_test_split
   from sklearn.linear_model import LinearRegression
   from sklearn.metrics import r2_score,mean_squared_error,mean_absolute_error
```

```
]: maxacc=0
   maxrs=0
   lr=LinearRegression()
   for i  in range(1,100):
       x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.1,random_state=i)
       lr.fit(x_train,y_train)
       y_pred=lr.predict(x_test)
       acc=r2_score(y_test,y_pred)
       if acc>maxacc:
           maxacc=acc
           maxrs=i
   print('best accuracy score is',maxacc,'on random_state',maxrs)

   best accuracy score is 0.9267238648916273 on random_state 59
```

Then we have to use random_state 59 on other algorithms to see where my accuracy is good compared to all algorithms.

```
model=[lr,ls,svr,rd,els,ex]
```

```
for i in model:
    i.fit(x_train,y_train)
    y_pred=i.predict(x_test)
    print(i)
    print(r2_score(y_test,y_pred))
    print(mean_squared_error(y_test,y_pred))
```

```
LinearRegression()
0.9267238648916273
393664143.5917082
Lasso()
0.926849976843782
392986627.595972
SVR()
-0.0951325514544874
5883421899.232688
Ridge()
0.9256394523375655
399489974.0447404
ElasticNet()
0.5920990524490574
2191381640.8357534
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
             importance_type='gain', interaction_constraints='',
             learning_rate=0.300000012, max_delta_step=0, max_depth=6,
             min_child_weight=1, missing=nan, monotone_constraints='()',
             n_estimators=100, n_jobs=4, num_parallel_tree=1, random_state=0,
             reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
             tree_method='exact', validate_parameters=1, verbosity=None)
0.872275990902445
686176510.034621
```

```
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor()
rf.fit(x_train,y_train)
y_Pred=rf.predict(x_test)
acc=r2_score(y_test,y_pred)
acc
```

```
0.872275990902445
```

```
from sklearn.ensemble import GradientBoostingRegressor
gb=GradientBoostingRegressor()
gb.fit(x_train,y_train)
y_Pred=gb.predict(x_test)
acc=r2_score(y_test,y_pred)
acc
```

```
0.872275990902445
```

```
from sklearn.ensemble import ExtraTreesRegressor
ext=ExtraTreesRegressor()
ext.fit(x_train,y_train)
y_Pred=ext.predict(x_test)
acc=r2_score(y_test,y_pred)
acc
```

```
0.872275990902445
```

From using above models we see that lasso is giving good accuracy compared to all regression and ensemble algorithm.So we have to Hyperparameter Tuning in Lasso Regression.

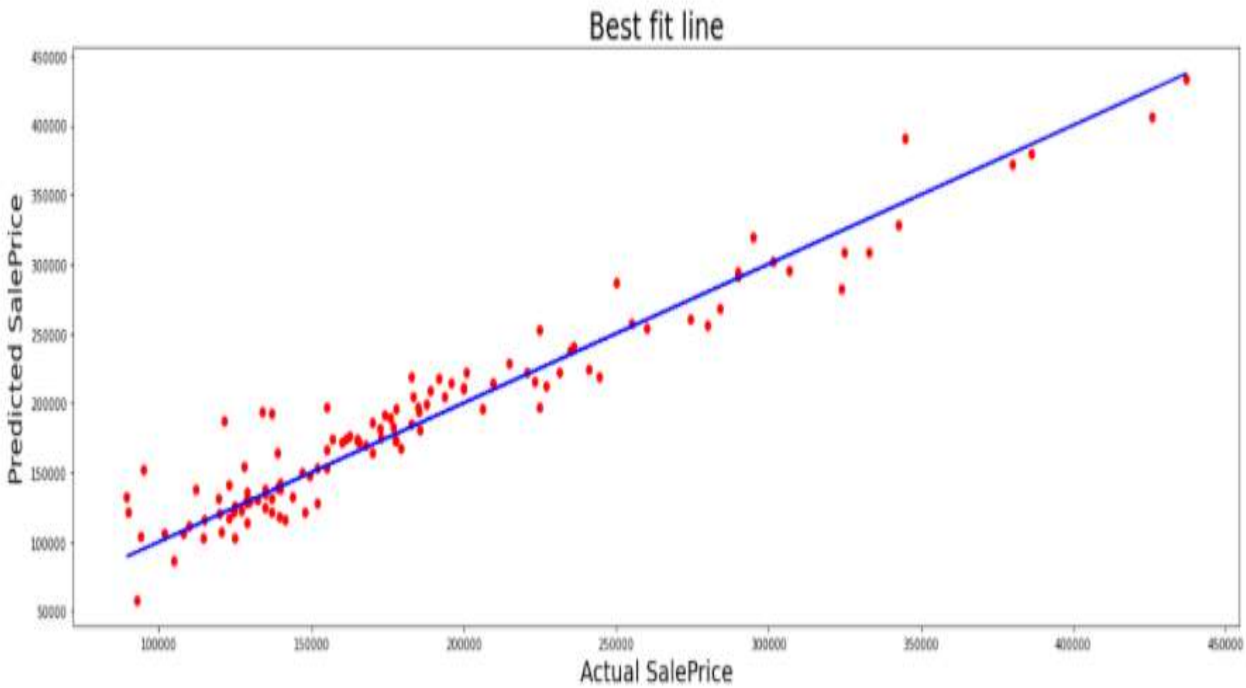
- ## Key Metrics for success in solving problem under consideration

I have used r2_score (to see accuracy of model)and mean_squared_error (to check how close regression line is to set of points)to see which model is best for our dataset.I conclude that Lasso Regression is the best model for the dataset.

- ## Visualizations

This is the best fit line of test data and predicted data.

```python
#Plotting y_test and y_pred
plt.figure(figsize=(22,7))
plt.scatter(x=y_test,y=y_pred,color='r')
plt.plot(y_test,y_test,color='b')
plt.xlabel('Actual SalePrice',fontsize=20)
plt.ylabel('Predicted SalePrice',fontsize=20)
plt.title('Best fit line',fontsize=25)
plt.show()
```



- Interpretation of the Results

Our model gives us accuracy score of 92.68% and after hypertuning my model I get accuracy of 92.72%.

CONCLUSION

- Key Findings and Conclusions of the Study

GrLivArea,GarageArea is highly positively corelated to saleprice.

SalePrice varies directly with the Overall quality

- ## Learning Outcomes of the Study in respect of Data Science

  In the first place, we finish analyzing our original data. We display some plots and calculation. And we find out the most mattered features to apply in our further research.

  Second, we have done some pretreatment to our original data. Due to null values are unavoidable from the dataset. To solve this, we take some measures to remove or replace the null values .

  Next, we apply the data to train a model. We use training set to train the model and give the test set as input. For each method, we print out the prediction of dependent variable and the sores of the model.

  Finally, we evaluate our results. The last step, we can compare the training results and the real value. Thus we print out a result about the accuracy of the model. And the higher the accuracy could be, the better our model would be.

  After choosing our best model,we predict our test.csv dataset after data cleaning,data preprocessing,feature scaling.

- ## Limitations of this work and Scope for Future Work

  We have to keep on fixing the model. Through adjust the parameters to improve our own model. Also, we can figure out some deep relationship in the data so that we can obtain more significant information and more accuracy.

  As data is increasing day by day,we have to make our dataset more clean according to the investors for increasing accuracy.