HOUSING PRICE PREDICTION PROJECT

Submitted By:-

Joshua Paul

ACKNOWLEDGMENT

Thanks for giving me the opportunity to work in FlipRobo Technologies as Intern and would like to express my gratitude to Data Trained Institute as well for trained me in Data Science Domain. This helps me to do my projects well and understand the concepts.

Resources used – Seaborn.pydata, matplolib.org for visualization, Scikit-learn for machine learning algorithms, Blogs for conceptual referring.

INTRODUCTION

- Business Problem Framing

  A US-based housing company named Surprise Housing has decided to enter the Australian market. The company uses data analytics to purchase houses at a price below their actual values and flip them at a higher price. For the same purpose, the company has collected a data set from the sale of houses in Australia.

  The company is looking at prospective properties to buy houses to enter the market. You are required 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.

  For this company wants to know:

  - Which variables are important to predict the price of variable?
  - How do these variables describe the price of the house?

- Conceptual Background of the Domain Problem

  Houses are one of the necessary needs of each and every person around the globe and therefore housing and real estate market is one of the markets which is one of the major contributors in the world's economy. It is a very large market and there are various companies working in the domain. The company is looking at prospective properties to buy houses to enter the market.

- Review of Literature

  I analyzed the given data and checked which attributes are performing well to predict the sale price. I dropped some columns which are not useful to train the model and filled null values in the given trained data. Removed skewness and outliers from the data, did an encoding for categorical data to train the model.

- Motivation for the Problem Undertaken

  Houses are one of the necessary need of each and every person around the globe and therefore housing and real estate market is one of the markets which is one of the major contributors in the world's economy. It is a very large market and there are various companies working in the domain. Data science comes as a very important tool to solve problems in the domain to help the companies increase their overall revenue, profits, improving their marketing strategies and focusing on changing trends in house sales and purchases. Predictive modelling, Market mix modelling, recommendation systems are some of the machine learning techniques used for achieving the business goals for housing companies.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem
- Here our target variable is the “Sale Price” and need to predict the prices of the house. As the data is continuous and our problem is Regression.
- Data Sources and their formats

  The Data is provided by Flip Robo Technologies, and it has Train and Test Data Set and need to train our data in Train dataset and need to load the Test dataset to make the predictions. Data is having Null values and We need to treat the missing values and can be discussed in pre-processing.

- Data Preprocessing Done
  The data set contains null values and handled those null values with sicikt-learn imputation pre-processing. Dropped columns ‘Id’(unique number), in columns where the data set contains nearly80% null values ‘PoolQC’, ‘Fence’, ‘MiscFeature’,etc.
- Data Inputs- Logic- Output Relationships

  Mainly the material used for the total constructions (exterior, foundation, interior), Garage(quality, basement height) and the age of the house are the main things decides the house sale price.
- State the set of assumptions (if any) related to the problem under consideration By

  looking at the data set, I understood that the main factor determining the price of the house depends on the age and quality of the house.

- Hardware and Software Requirements and Tools Used

  Model training was done on Jupiter Notebook.

  Kernel Version is Python3.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
  I used seaborn to analyze the categorical and numerical data to check skewness and outliers. With the function df.describe() we can check the descriptive statistics of all numerical columns.

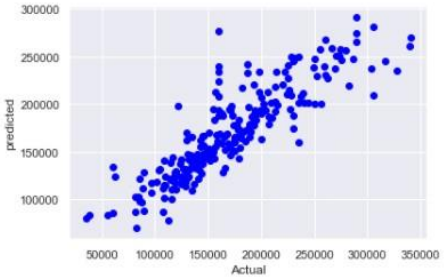
- Testing of Identified Approaches (Algorithms)
  RandomForest
  KNeighbour
  AdaBoost
  Gradient Boost
  Linear Regression

- Run and Evaluate selected models

In [105]:
```
# Random forest algorithm

from sklearn.ensemble import RandomForestRegressor

rfr = RandomForestRegressor()
rfr.fit(x_train,y_train)
y_pred = rfr.predict(x_test)
scr_rfr = cross_val_score(rfr,x,y,cv=5)

print("r2_Score", r2_score(y_test,y_pred))
print("CV Score", scr_rfr.mean())
print("MSE",mean_squared_error(y_test,y_pred))
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
print("Train Score", rfr.score(x_train,y_train))
print("Test Score", rfr.score(x_test,y_test))
```

```
r2_Score 0.7683850815041416
CV Score 0.7457957638102433
MSE 797948675.3029962
RMSE 28247.985331754124
Train Score 0.9601569156118303
Test Score 0.7683850815041416
```

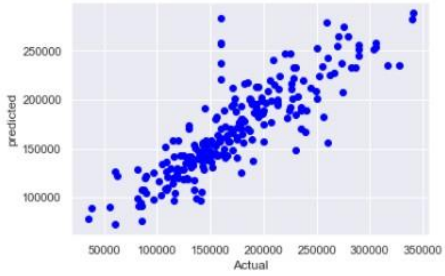
In [107]:
```
plt.scatter(y_test,y_pred, color = 'blue')      #Scatter Matrix for Actual VS predicted for the model
plt.xlabel("Actual")
plt.ylabel("predicted")
plt.show()
```



In [108]:
```
#KNeighborsRegressor

knr = KNeighborsRegressor(n_neighbors = 5)
knr.fit(x_train,y_train)
y_pred = knr.predict(x_test)
scr_knr = cross_val_score(knr,x,y,cv=5)

print("r2_Score", r2_score(y_test,y_pred))
print("CV Score", scr_knr.mean())
print("MSE",mean_squared_error(y_test,y_pred))
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
print("Train Score", knr.score(x_train,y_train))
print("Test Score", knr.score(x_test,y_test))
```

```
r2_Score 0.7240238145058926
CV Score 0.5519686312843624
MSE 950779997.5075214
RMSE 30834.72064909169
Train Score 0.7720316309484047
Test Score 0.7240238145058926
```

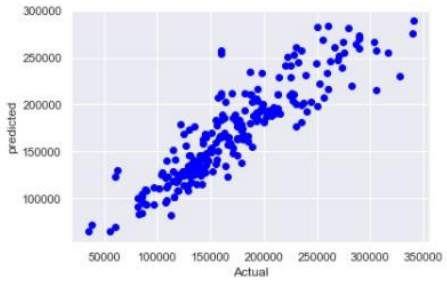
In [110]:
```
plt.scatter(y_test,y_pred, color = 'blue')         #Scatter Matrix for Actual VS predicted for the model
plt.xlabel("Actual")
plt.ylabel("predicted")
plt.show()
```



In [111]:
```
#GradientBoostingRegressor
from sklearn.ensemble import GradientBoostingRegressor
gbr = GradientBoostingRegressor()
gbr.fit(x_train, y_train)
y_pred = gbr.predict(x_test)
scr_gbr = cross_val_score(gbr,x,y,cv=5)

print("r2_Score", r2_score(y_test,y_pred))
print("CV Score", scr_gbr.mean())
print("MSE",mean_squared_error(y_test,y_pred))
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
print("Train Score", gbr.score(x_train,y_train))
print("Test Score", gbr.score(x_test,y_test))
```

```
r2_Score 0.7893941046484229
CV Score 0.751452371962621
MSE 725569390.3404477
RMSE 26936.395273689606
Train Score 0.9156286309462046
Test Score 0.7893941046484229
```

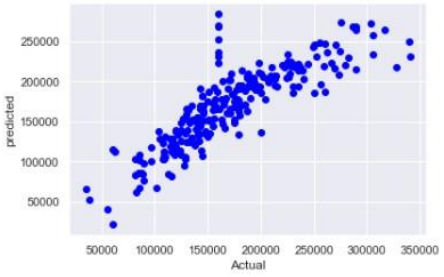
In [112]:
```
plt.scatter(y_test,y_pred, color = 'blue')    #Scatter Matrix for Actual VS predicted for the model
plt.xlabel("Actual")
plt.ylabel("predicted")
plt.show()
```



In [113]:
```
# Linear Regression

from sklearn.linear_model import LinearRegression

lr = LinearRegression()
lr.fit(x_train,y_train)
y_pred = lr.predict(x_test)
scr_lr = cross_val_score(lr,x,y,cv=5)

print("r2_Score", r2_score(y_test,y_pred))
print("CV Score", scr_lr.mean())
print("MSE",mean_squared_error(y_test,y_pred))
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
print("Train Score", lr.score(x_train,y_train))
print("Test Score", lr.score(x_test,y_test))
```

```
r2_Score 0.7194377394441989
CV Score 0.6952036967821129
MSE 966579724.6757178
RMSE 31089.865304882198
Train Score 0.7244934918276908
Test Score 0.7194377394441989
```

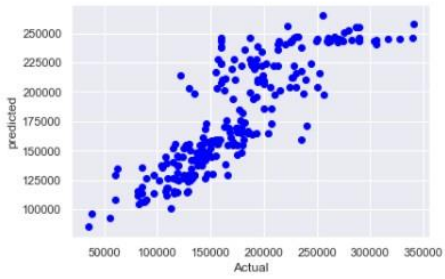
In [114]:
```
plt.scatter(y_test,y_pred, color = 'blue')     #Scatter Matrix for Actual VS predicted for the model
plt.xlabel("Actual")
plt.ylabel("predicted")
plt.show()
```



In [115]:
```
# Ada boost model

from sklearn.ensemble import AdaBoostRegressor
abr = AdaBoostRegressor()
abr.fit(x_train, y_train)
y_pred = abr.predict(x_test)
scr_abr = cross_val_score(abr,x,y,cv=5)

print("r2_Score", r2_score(y_test,y_pred))
print("CV Score", scr_abr.mean())
print("MSE",mean_squared_error(y_test,y_pred))
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
print("Train Score", abr.score(x_train,y_train))
print("Test Score", abr.score(x_test,y_test))
```

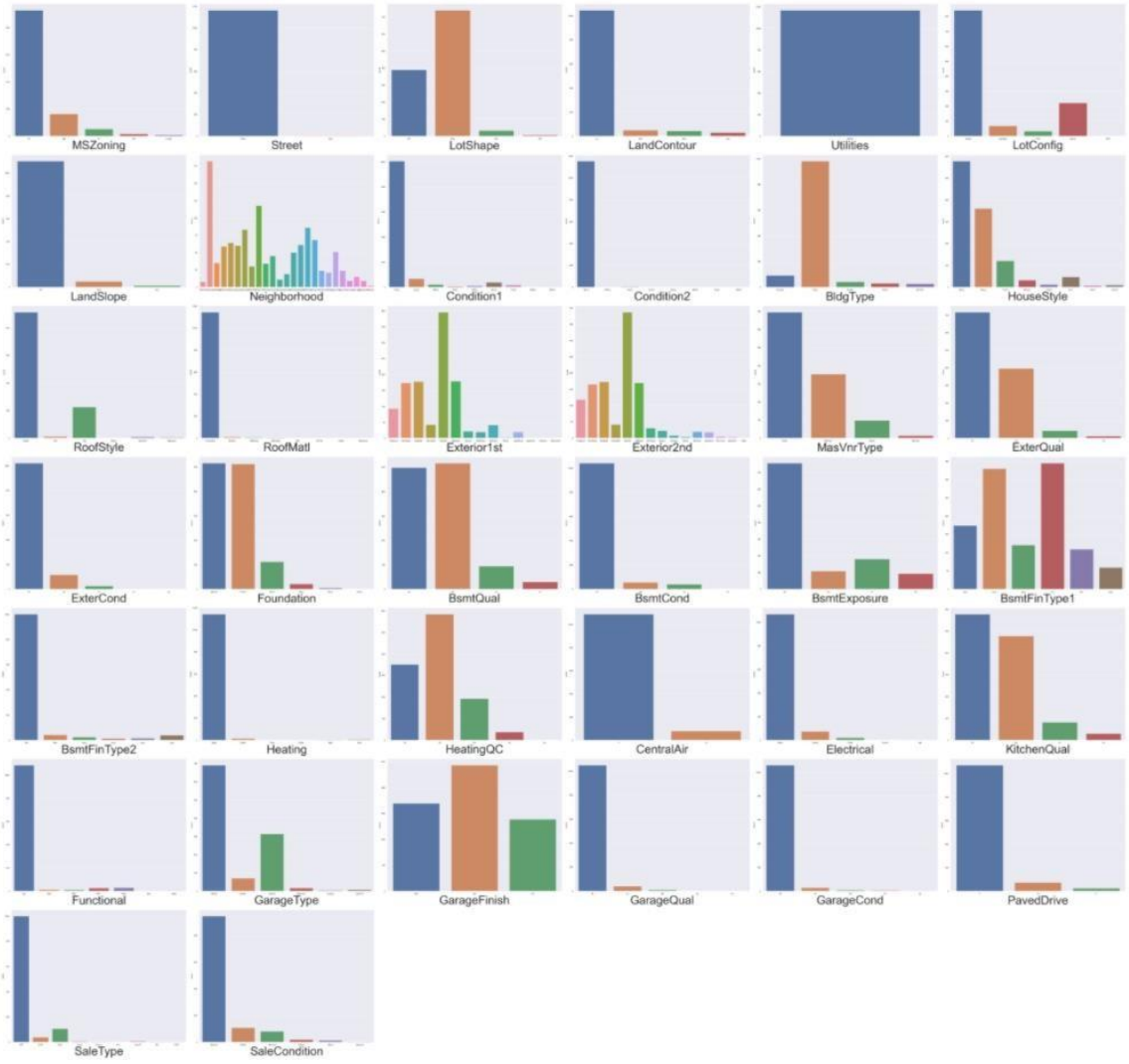
```
r2_Score 0.6957026831801524
CV Score 0.6804073545627369
MSE 1048350608.9828812
RMSE 32378.242833465825
Train Score 0.7373689924029309
Test Score 0.6957026831801524
```

In [116]:
```
plt.scatter(y_test,y_pred, color = 'blue')    #Scatter Matrix for Actual VS predicted for the model
plt.xlabel("Actual")
plt.ylabel("predicted")
plt.show()
```



- Key Metrics for success in solving problem under consideration
  Since this is regression model RMSE (Root mean squared error) value is considered to check how good the model is performing. From the above mentioned algorithms we can see that Gradient Boost is the Best Model.
- Visualizations

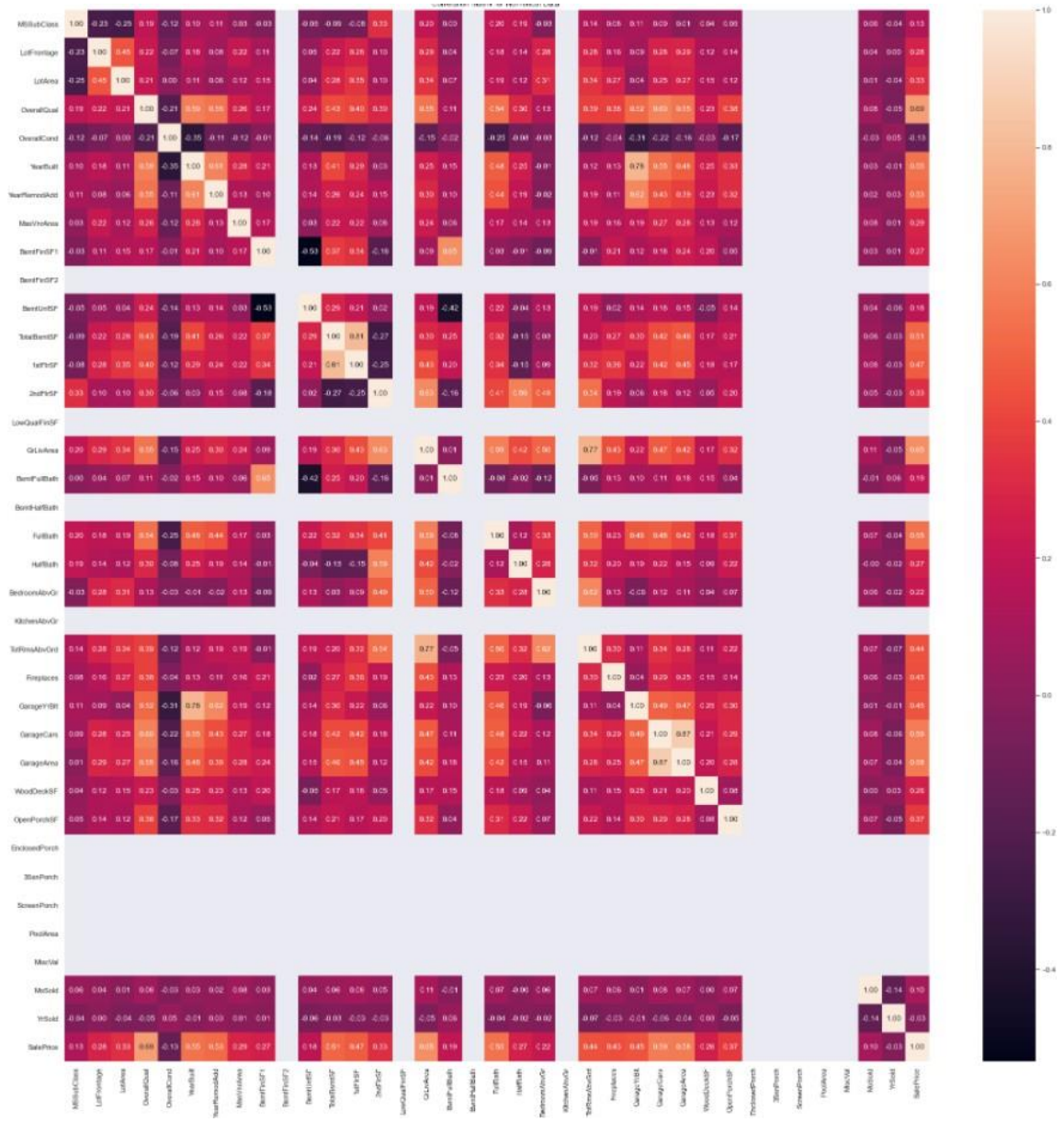
Box plot and Dist plot (checking for outliers and skewness)




MSSubClass
LotFrontage
LotArea
OverallQual
OverallCond
YearBuilt
YearRemodAdd
MasVnrArea
BsmtFinSF1
BsmtFinSF2
BsmtUnfSF
TotalBsmtSF
1stFlrSF
2ndFlrSF
LowQualFinSF
GrLivArea
BsmtFullBath
BsmtHalfBath
FullBath
HalfBath
BedroomAbvGr
KitchenAbvGr
TotRmsAbvGrd
Fireplaces
GarageYrBlt
GarageCars
GarageArea
WoodDeckSF
OpenPorchSF
EnclosedPorch
3SsnPorch
ScreenPorch
PoolArea
MiscVal
MoSold
YrSold
SalePrice


MSSubClass
LotFrontage
LotArea
OverallQual
OverallCond
YearBuilt
YearRemodAdd
MasVnrArea
BsmtFinSF1
BsmtFinSF2
BsmtUnfSF
TotalBsmtSF
1stFlrSF
2ndFlrSF
LowQualFinSF
GrLivArea
BsmtFullBath
BsmtHalfBath
FullBath
HalfBath
BedroomAbvGr
KitchenAbvGr
TotRmsAbvGrd
Fireplaces
GarageYrBlt
GarageCars
GarageArea
WoodDeckSF
OpenPorchSF
EnclosedPorch
3SsnPorch
ScreenPorch
PoolArea
MiscVal
MoSold
YrSold
SalePrice

Correlation



- Interpretation of the Results

In [119]:
```
# Applying randomized search CV to increase the accuracy,

rg = RandomizedSearchCV(gbr, param_distributions = param, cv= 5)
rg.fit(x_train,y_train)
rg.best_params_
```

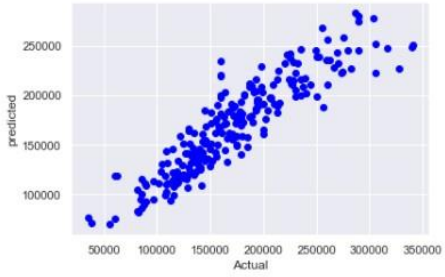
Out[119]:
```
{'n_estimators': 100,
 'min_samples_leaf': 20,
 'loss': 'huber',
 'learning_rate': 0.1,
 'criterion': 'mae'}
```

In [120]:
```
final = GradientBoostingRegressor(loss = 'lad',n_estimators= 100,criterion= 'mse', learning_rate = 0.2 , min_samples_leaf = 35 )
final.fit(x_train, y_train)
y_pred = final.predict(x_test)

print("r2_Score", r2_score(y_test,y_pred))
print("MSE",mean_squared_error(y_test,y_pred))
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
print("Train Score", final.score(x_train,y_train))
print("Test Score", final.score(x_test,y_test))
```

```
r2_Score 0.8085626066397584
MSE 659530981.1099015
RMSE 25681.335267269526
Train Score 0.8229817108814775
Test Score 0.8085626066397584
```

Our model has given the accuracy score ie r2 score of 81% which is high than the CV score of model and r2 score of model.

In [121]:
```
plt.scatter(y_test,y_pred, color = 'blue')   #Scatter Matrix for Actual VS predicted for the model
plt.xlabel("Actual")
plt.ylabel("predicted")
plt.show()
```



CONCLUSION

- Key Findings and Conclusions of the Study

- As this project is about predicting the price of house in US market, The problem is Regression and I have used 5 algorithms to build the model and among all Gradient Boosting algorithm is the best model.

  I have used hyper parameter tuning to increase the accuracy score as well and price which predicted was slightly varying from the actual price.

  We have 2 datasets for training and testing the data and it has more columns but less rows and removed the columns which is less and no importance using feature importance feature engineering.
  So, this will helps the clients to understand the variables which are important to predict the price of house and helps them to get success in their business.

  - Learning Outcomes of the Study in respect of Data Science
  - Random forest and Boosting algorithms work best and Randomized search CV is faster than Grid search CV.

  As this dataset has 2 data files and we need to do all pre- processing, EDA to both datasets and train dataset has target variable and test dataset will not have target variable which we need to predict it.
  This is kind of different and time taking process but helps me to learn more and most important is that I am getting hands-on experience more on Data Science Concepts.

- Limitations of this work and Scope for Future Work

  If there is skewness in the given data first we have to apply log then square root transformation taking the threshold value as +/- 0.5, if these both transformation are not working good, then the distribution of the data in that column is not uniform. For such cases we have to build model keeping skewed data.

THANK YOU