FLIP ROBO

Housing: Price Prediction Project

Submitted by:

Arpita Rai

Int 33

ACKNOWLEDGMENT

Firstly, would like to thank and convey my heartfelt gratitude to Flip Robo Technologies for providing this opportunity to work on project "Housing Price Prediction" and would also like to thank my SME Mr. Shwetank Mishra for providing the dataset and guiding to complete this project.

I would also like to thank Data Trained and their team who extended Machine Learning tools and its working. This project would not have been accomplished without their learning.

INTRODUCTION

- ## Business Problem Framing

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. Our problem is related to one such housing company. 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.

We need 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

1. To analyse and compare model’s performance in order to choose the best model.
2. To build machine learning models to predict sale price of house
3. To visualize data with plot graphs and map.
4. To analyse data.

- ## Review of Literature

  We have used **Machine Learning** to build the model. Machine learning algorithms use historical data as input to predict new output values. The type of algorithm data scientists choose to use depends on what type of data they want to predict.
  We used **regression models** for predicting Sale price of houses by using various features to have lower Root mean Squared error. While using features in a regression model some feature engineering is required for better prediction. Often a set of features linear regression, random forest regression and decision tree regression is used for making better model fit.

- ## Motivation for the Problem Undertaken

  Surprise Housing Company has decided to enter the Australian market. Now the company is looking at prospective properties to buy houses to enter the market
  We need to build a model using Machine Learning in order to
  1. Predict the actual value of the prospective properties and decide whether to invest in them or not.
  2. We also need to build model the price of houses with the available independent variables.
  3. We need to find important features which affect the price positively or negatively.

Analytical Problem Framing

- ## Mathematical/ Analytical Modeling of the Problem

  To analyze the data, there are many techniques.

  The regression models are used to examine relationships between variables.

  The most traditional regression models are

  a. linear regression
  b. decision tree regression,
  c. random forest regression
  d. gradient boosting regression
  e. KNN-Neighbors.

- ## Data Sources and their formats

  The dataset is given by a US-based housing company named Surprise Housing . 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.

```
df = pd.read_excel(r'C:\Users\Arpita\Desktop\train.xlsx')
df
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal
0	127	120	RL	NaN	4928	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
1	889	20	RL	95.0	15865	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
2	793	60	RL	92.0	9920	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
3	110	20	RL	105.0	11751	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
4	422	20	RL	NaN	16635	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1163	289	20	RL	NaN	9819	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
1164	554	20	RL	67.0	8777	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
1165	196	160	RL	24.0	2280	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0
1166	31	70	C (all)	50.0	8500	Pave	Pave	Reg	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
1167	617	60	RL	NaN	7861	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0

1168 rows × 81 columns

- ## Data Preprocessing Done

  1. Importing libraries
  2. Importing data
  3. Checking Total Numbers of Rows and Column

4. Checking All Column Name
5. Checking Data Type of All Data
6. Checking for Null Values
7. Information about Data
8. Checking total number of unique value
9. Checking all value of each columns
10. Handling Null Values by filling with mean and mode

```
#importing the required libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import zscore
from sklearn.preprocessing import StandardScaler,MinMaxScaler
from sklearn.preprocessing import LabelEncoder,OneHotEncoder
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.linear_model import LinearRegression,Ridge
from sklearn.datasets import load_boston
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error,mean_absolute_error,accuracy_score
from sklearn.model_selection import cross_val_score
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
import warnings
warnings.filterwarnings("ignore")
from sklearn.model_selection import GridSearchCV
import pickle
```

```
df.head()
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	Mc
0	127	120	RL	NaN	4928	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
1	889	20	RL	95.0	15865	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
2	793	60	RL	92.0	9920	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	
3	110	20	RL	105.0	11751	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0	
4	422	20	RL	NaN	16635	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0	

5 rows × 81 columns

```
df.tail()
```

	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal
1163	289	20	RL	NaN	9819	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
1164	554	20	RL	67.0	8777	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
1165	196	160	RL	24.0	2280	Pave	NaN	Reg	Lvl	AllPub	...	0	NaN	NaN	NaN	0
1166	31	70	C (all)	50.0	8500	Pave	Pave	Reg	Lvl	AllPub	...	0	NaN	MnPrv	NaN	0
1167	617	60	RL	NaN	7861	Pave	NaN	IR1	Lvl	AllPub	...	0	NaN	NaN	NaN	0

5 rows × 81 columns

```
#Checking null values
df.isnull().sum()
```

```
Id                 0
MSSubClass         0
MSZoning           0
LotFrontage      214
LotArea            0
                ... 
MoSold             0
YrSold             0
SaleType           0
SaleCondition      0
SalePrice          0
Length: 81, dtype: int64
```

As we can see that null/NAN values are present in dataset

```
#Dropping of unnecessary columns
df.drop(['Id','Alley','PoolQC','MiscFeature','Fence'],axis=1,inplace=True)
```

```
df.describe()
```

	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	...	WoodDeck
count	1168.000000	954.00000	1168.000000	1168.000000	1168.000000	1168.000000	1168.000000	1161.000000	1168.000000	1168.000000	...	1168.000
mean	56.767979	70.98847	10484.749144	6.104452	5.595890	1970.930651	1984.758562	102.310078	444.726027	46.647260	...	96.206
std	41.940650	24.82875	8957.442311	1.390153	1.124343	30.145255	20.785185	182.595606	462.664785	163.520016	...	126.158
min	20.000000	21.00000	1300.000000	1.000000	1.000000	1875.000000	1950.000000	0.000000	0.000000	0.000000	...	0.000
25%	20.000000	60.00000	7621.500000	5.000000	5.000000	1954.000000	1966.000000	0.000000	0.000000	0.000000	...	0.000
50%	50.000000	70.00000	9522.500000	6.000000	5.000000	1972.000000	1993.000000	0.000000	385.500000	0.000000	...	0.000
75%	70.000000	80.00000	11515.500000	7.000000	6.000000	2000.000000	2004.000000	160.000000	714.500000	0.000000	...	171.000
max	190.000000	313.00000	164660.000000	10.000000	9.000000	2010.000000	2010.000000	1600.000000	5644.000000	1474.000000	...	857.000

8 rows × 37 columns

Filling the missing/null values

```
#filling nan values of the numerical features by using mean
data['LotFrontage']=data['LotFrontage'].fillna(data['LotFrontage'].mean())
data['GarageYrBlt']=data['GarageYrBlt'].fillna(data['GarageYrBlt'].mean())
data['MasVnrArea']=data['MasVnrArea'].fillna(data['MasVnrArea'].mean())
```

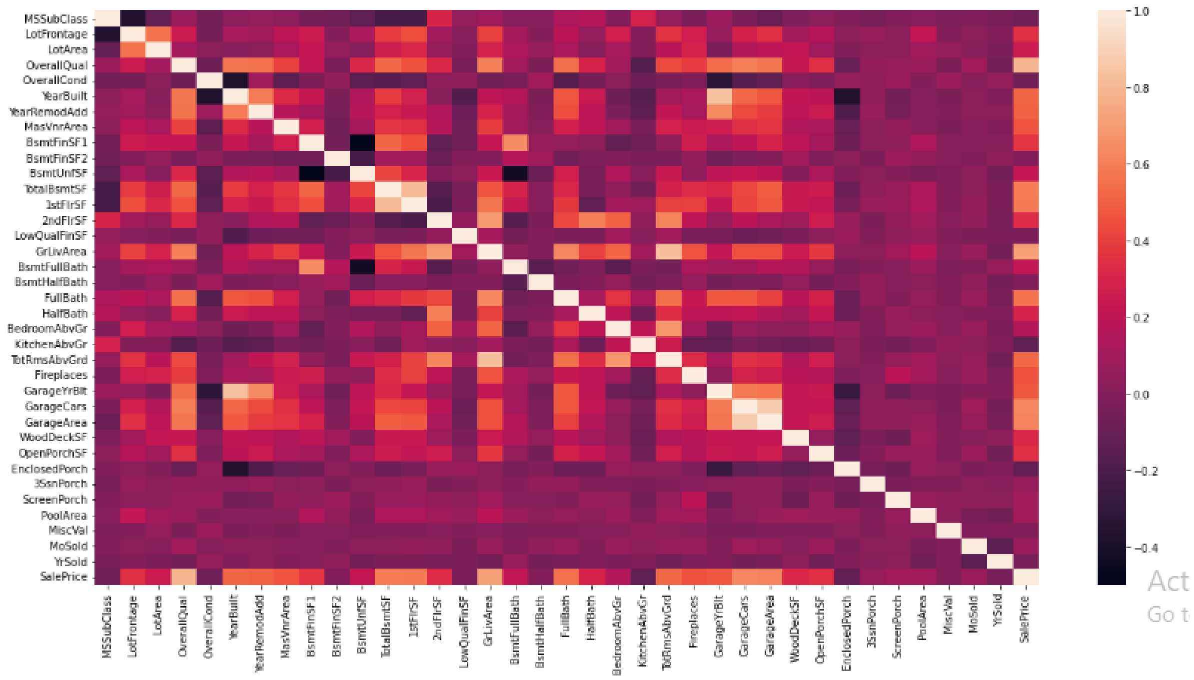
```
#filling the nan values of the categorical features by using mode
data['MasVnrType']=data['MasVnrType'].fillna(data['MasVnrType'].mode()[0])
data['BsmtQual']=data['BsmtQual'].fillna(data['BsmtQual'].mode()[0])
data['BsmtCond']=data['BsmtCond'].fillna(data['BsmtCond'].mode()[0])
data['BsmtExposure']=data['BsmtExposure'].fillna(data['BsmtExposure'].mode()[0])
data['GarageType']=data['GarageType'].fillna(data['GarageType'].mode()[0])
data['GarageFinish']=data['GarageFinish'].fillna(data['GarageFinish'].mode()[0])
data['GarageQual']=data['GarageQual'].fillna(data['GarageQual'].mode()[0])
data['GarageCond']=data['GarageCond'].fillna(data['GarageCond'].mode()[0])
data['FireplaceQu']=data['FireplaceQu'].fillna(data['FireplaceQu'].mode()[0])
data['BsmtFinType1']=data['BsmtFinType1'].fillna(data['BsmtFinType1'].mode()[0])
data['BsmtFinType2']=data['BsmtFinType2'].fillna(data['BsmtFinType2'].mode()[0])
data['Electrical']=data['Electrical'].fillna(data['Electrical'].mode()[0])
```

```
data.drop('GarageYrBlt',axis=1,inplace=True)
```

- Data Inputs- Logic- Output Relationships
    1. **Checking for Correlation**
    2. **Plotting Correlation on heatmap**
    3. **Checking for Outliers**
    4. **Removing Outliers**
    5. **Scaling and splitting data**
    6. **Concatenating both train and test data.**

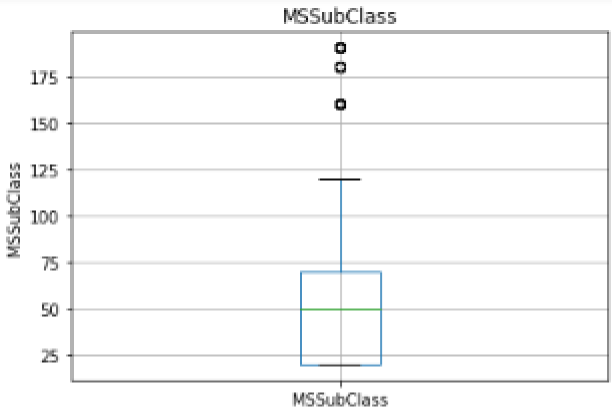
```
#Visualising the correlation on heatmap
plt.figure(figsize=(20,10))
sns.heatmap(corr)
```

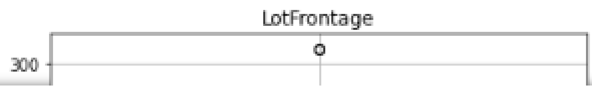
<AxesSubplot:>



Checking and plotting Outliers

```
for feature in continuous_df:
    df1=df.copy()
    if 0 in df1[feature].unique():
        pass
    else:
        df1[feature]=np.log(df1[feature])
        df.boxplot(column=feature)
        plt.ylabel(feature)
        plt.title(feature)
        plt.show()
```





Removing Outliers ¶

```
#using zscore to remove the outliers
z=np.abs(zscore(df[['MSSubClass','LotFrontage','LotArea','OverallQual','OverallCond','YearBuilt','YearRemodAdd','1stFlrSF',
                    'GrLivArea','TotRmsAbvGrd','GarageYrBlt','MoSold','YrSold','SalePrice']]))
threshold=3
print(np.where(z>3))
df_new=df[(z<3).all(axis=1)]
print(z)
```

```
(array([  27,   32,   48,   68,   72,   78,  103,  103,  103,  103,  113,
        119,  127,  141,  141,  141,  153,  184,  192,  192,  192,  210,
        232,  232,  241,  241,  245,  273,  299,  300,  305,  305,  305,
        356,  361,  361,  361,  362,  363,  369,  389,  389,  394,  395,
        401,  403,  413,  423,  423,  452,  488,  504,  504,  515,  517,
        521,  553,  561,  572,  581,  582,  590,  592,  592,  592,  592,
        600,  602,  611,  614,  614,  614,  615,  625,  642,  650,  655,
        689,  691,  691,  691,  705,  713,  723,  762,  769,  797,  804,
        821,  830,  833,  839,  839,  846,  846,  858,  867,  882,  884,
        914,  966,  980, 1032, 1038, 1039, 1049, 1074, 1082, 1102, 1104,
       1120, 1123, 1123, 1142, 1144], dtype=int64), array([ 4,  9,  3,  7,  0, 13,  4,  8,  9, 13,  2,  2,  0,  7,  8, 13,  4,
        0,  4,  5,  8,  0,  8, 13,  7, 13,  2,  8,  5, 13,  7,  8, 13,  2,
        2,  7,  8,  0,  0, 13,  0,  4, 13,  4,  4,  0,  0,  9, 13,  8,  4,
        7, 13,  4,  9,  4,  0, 13,  9,  0,  4,  0,  2,  7,  8,  9,  2,  4,
```

- Hardware and Software Requirements and Tools Used

Python is considered a high-level language. So we have build the model using Python on Jupyter Notebook.

Imported Libraries such as

1. Numpy- It is a popular array – processing package of Python.

2. Pandas- The Pandas is used to execute a Data frame i.e., test set.csv, train set.csv, skewness, co-efficient, predicted values of model approach, conclusion.
3. Sklearn- power transform, label encoder, standard scaler, linear, random forest, decision tree, Gradient boosting Regressor, k-nearest neighbours, r2 score, mean absolute error, mean squared error, train test split, grid search cv and ensemble technique.
4. Matplot- It is a Python library that uses Python Script to write 2-dimensional graphs and plots.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

In this project, we have to predict the actual value of the prospective properties.

Extensive EDA has to be performed to gain relationships of important variable and price.

The sale price we want to predict is a continuous data, so need to understand it with regression problem.

- Testing of Identified Approaches (Algorithms)
  1. Decision Tree Regressor
  2. Random Forest Regressor
  3. Gradient Boosting Regressor
  4. Cross Validation Score
  5. Hyper Parameter Tuning
  6. Saving the model through Pickle

- Run and Evaluate selected models

```
x_train=df_train.drop(['SalePrice'],axis=1)
y_train=df_train['SalePrice']
```

Feature Selection

```
#using MinMaxScaler to scale the data
scaler=MinMaxScaler()
scaled=scaler.fit_transform(df_new)
scaled
```

```
array([[0.58823529, 0.17119339, 0.01695763, ..., 0.        , 1.        ,
        0.        ],
       [0.        , 0.25342466, 0.06807824, ..., 0.        , 1.        ,
        0.        ],
       [0.23529412, 0.24315068, 0.04029073, ..., 0.        , 1.        ,
        0.        ],
       ...,
       [0.        , 0.15556542, 0.04646521, ..., 0.        , 1.        ,
        0.        ],
       [0.17647059, 0.09931507, 0.01729416, ..., 0.        , 1.        ,
        0.        ],
       [0.82352941, 0.        , 0.00305219, ..., 0.        , 1.        ,
        0.        ]])
```

Best Model Selection

```
#Using DecisionTreeRegressor
dt=DecisionTreeRegressor()
dt.fit(x_train,y_train)
pred_dt=dt.predict(df_test)
result_dt=dt.score(x_train,y_train)*100
print(result_dt)
```

```
100.0
```

```
#Using GradientBoostingRegressor
gbr = GradientBoostingRegressor()
gbr.fit(x_train, y_train)
pred_gbr=gbr.predict(df_test)
result_gbr=gbr.score(x_train,y_train)*100
print(result_gbr)
```

```
96.87077704613827
```

```
#Using RandomForestRegressor
rf=RandomForestRegressor()
rf.fit(x_train,y_train)
pred_rf=rf.predict(df_test)
print(rf.score(x_train,y_train)*100)
```

```
97.76417206732529
```

Cross Validation score:

```
print(cross_val_score(dt,x_train,y_train,cv=5).mean()*100)
```

```
72.40074050581194
```

```
print(cross_val_score(gbr,x_train,y_train,cv=5).mean()*100)
```

```
87.46020597665411
```

```
print(cross_val_score(rf,x_train,y_train,cv=5).mean()*100)
```

```
84.01249747180361
```

Hyper Parameter Tuning

```
parameters={"max_features":[None,'sqrt','auto'],
           "n_estimators":[100,200],
           "max_depth":[3,6],                    #these are the parameters corresponding to the Gradient Boosting Regressor
           "subsample":[1.0],
           "criterion":['friedman_mse','mse']}
grid=GridSearchCV(estimator=gbr, param_grid=parameters,cv=5)
grid.fit(x_train,y_train)
```

```
GridSearchCV(cv=5, estimator=GradientBoostingRegressor(),
             param_grid={'criterion': ['friedman_mse', 'mse'],
                         'max_depth': [3, 6],
                         'max_features': [None, 'sqrt', 'auto'],
                         'n_estimators': [100, 200], 'subsample': [1.0]})
```

```
#obtaining the best score
(grid.best_score_)*100
```

```
88.11493473685879
```

```
#finding best parameters
print(grid.best_params_)
```

```
{'criterion': 'mse', 'max_depth': 3, 'max_features': None, 'n_estimators': 200, 'subsample': 1.0}
```

```
#using new parameters and checking the score
Final_mod=GradientBoostingRegressor(max_features='sqrt',criterion='mse',max_depth=3,subsample=1.0,n_estimators=100)
Final_mod.fit(x_train,y_train)
```

```
GradientBoostingRegressor(criterion='mse', max_features='sqrt')
```

Saving the Model

```
import pickle
filename = 'Housing_Use_case.pkl'
pickle.dump(Final_mod, open(filename,"wb"))
print("Model saved")
```

```
Model saved
```

CONCLUSION

- Key Findings and Conclusions of the Study
  1. We compared the predicted and actual price the house
  2. Gradient Boosting Regressor was the best suited model
  3. It gave accuracy of 87.5
  4. The target column is not having a negative correlation with any of the existing feature and it has a positive relation with OverallQual.
  5. We removed Outliers by using ZSCORE method.
  6. If the house is built 140 years ago then its price is less and between zero to twenty years the price is high
  7. The newer house(which are built or renovated 10-20 year ago)are having highest sales price.
  8. There is a huge difference between 75th percentile and the maximum value in the features like MSSubClass,LotFrontage,LotArea etc.
  9. Mean is greater than median in features like MSSubClass, MasVnrArea,BsmtFinSF1,BsmtFinSF2.

- Learning Outcomes of the Study in respect of Data Science

  Use appropriate models for analysis, assess the quality of input, derive insight from results, and investigate potential issues.
  Formulate and use appropriate models of data analysis to solve hidden solutions to business-related challenges

Try to remove skewness,Outliers to get a clean data.
Try to balance the data by filling or removing the Null/NAN values.

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

More variables can be added, we can try different models with different subset of features and/or rows .

Machine learning require large amount of data.

This project has scope for improvement.