FLIP ROBO

HOUSE PRICE PREDICTION

Submitted by:

M. SAI GIRISH

ACKNOWLEDGMENT

I like to extend my thanks for my mentor Dr. Deepika sharma, SME Ms. Kushboo Garg and Flip Robo for giving me this project to work with. I extend my warm thanks to all who helped me unknowingly by contributing their work online for open.

REFERENCES:

www.kaggle.com

www.google.com

www.youtube.com

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. 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

While buying or selling a house, there are many factors that plays role in price determining. Type of house, land, basement, Internal and external conditions, sq. feet, extra amenities, area, etc.

- ## Motivation for the Problem Undertaken

Project provided by Flip Robo Technologies

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

The target column is price which is continuous variable. So we use Regression model to predict the target column.

Different algorithms have been implemented

Model	R2 Score	CV Score
Linear Regression	0.60	0.72
SVR	0.03	0.06
Random Forest Regressor	0.81	0.82
Gradient Boosting Regressor	0.84	0.82

- Data Sources and their formats

Data provided by Flip Robo Technologiess

Format: CSV

Two csv files provided,

1 -> Train data: 1168 rows & 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
dtypes: float64(3), int64(35), object(43)
memory usage: 739.2+ KB
```

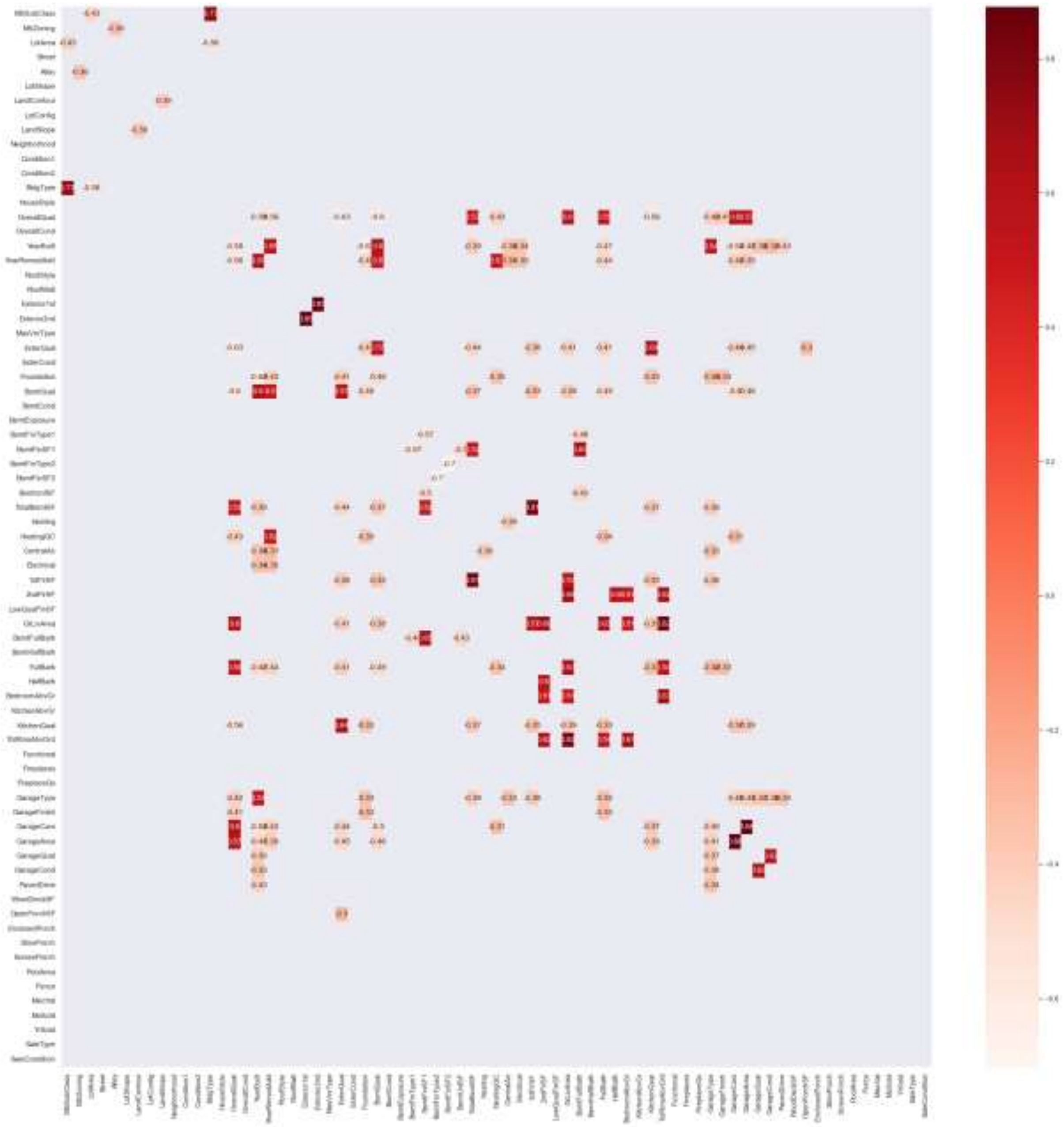
2-> Test data: 292 rows & 80 columns

- Data Preprocessing Done
  - ➔ Checked for the missing values.
  - ➔ Dropped the columns with more number of missing data.
  - ➔ Filled the missed values with appropriate values.
  - ➔ Encoded the categorical data.
  - ➔ Found the correlation
  - ➔ Dropped the columns with multicollinearity.
  - ➔ Removed outliers.
  - ➔ Split features and Target column.
  - ➔ Scaled the data of features.

- Data Inputs- Logic- Output Relationships

  Now the data is split into train and test. Then different algorithms were applied to predict the price.

  Correlation: Plotting the corr, which with threshold > 0.3



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

Assumed missing values as Nan ->i.e. no such thing.

Reduced the number of features to avoid curse of dimensionality using pca (Principal component analysis)

- Hardware and Software Requirements and Tools Used

- Hardware used:
- RAM: 8GB
- Processor: Intel I5
- Software's:
- Jupyter Notebook (Anaconda Framework)
- Python (coding language)
- Libraries / Packages used – pandas, Numpy, Sklearn, Seaborn, Selenium (Web scrapping)

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

1. Clean the dataset from unwanted details.
2. Rename values with meaningful information. Fill missing values.
3. Encoding the categorical data to get numerical input data.
4. Compare different models and identify the suitable model.
5. R2 score is used as the primary evaluation metric.
6. MSE and RMSE are used as secondary metrics.
7. Cross Validation Score was used to ensure there are no overfitting our underfitting models.

- # Testing of Identified Approaches (Algorithms)
- Since the target variable is continuous, we used regression model
- Different regression models we used to predict are:
- Linear Regression
- SVR
- DecisionTreeRegressor
- RandomForestRegressor
- GradientBoostingRegressor

- # Run and Evaluate selected models

```
from sklearn.model_selection import cross_val_score
ml_models=[LinearRegression(),SVR(),RandomForestRegressor(),GradientBoostingRegressor()]
for m in ml_models:
    m.fit(x_train,y_train)
    predm=m.predict(x_test)
    mse=mean_squared_error(y_test,predm)
    mae=mean_absolute_error(y_test,predm)
    r2=r2_score(y_test,predm)
    print(f'metrics of {m}:')
    print('Training score:', m.score(x_train,y_train))
    print('Testing Score:',m.score(x_test,y_test))
    print(f' mean_absolute_error: {mae}\n mean_squared_error: {mse}\n r2_score: {r2} ')
    score=cross_val_score(m,x_scaled,y, cv=5)
    print(' mean cv score:',score.mean())
    print('**'*20 , '\n')
```

```
metrics of LinearRegression():
Training score: 0.8457180250814662
Testing Score: 0.5960727847377936
 mean_absolute_error: 24845.1792482903
 mean_squared_error: 2752626456.402911
 r2_score: 0.5960727847377936
 mean cv score: 0.7216312376830656
****************************************

metrics of SVR():
Training score: -0.05591064704062787
Testing Score: -0.030699741451764906
 mean_absolute_error: 55381.5052683813
 mean_squared_error: 7023867839.868291
 r2_score: -0.030699741451764906
 mean cv score: -0.062075842649206917
****************************************

metrics of RandomForestRegressor():
Training score: 0.9723800873270745
Testing Score: 0.8146833797764066
 mean_absolute_error: 22123.050787671233
 mean_squared_error: 1262869676.427971
 r2_score: 0.8146833797764066
 mean cv score: 0.82012116876287
****************************************

metrics of GradientBoostingRegressor():
Training score: 0.9589896630742526
Testing Score: 0.8363903995266035
 mean_absolute_error: 20145.969734904575
 mean_squared_error: 1114943726.9094028
 r2_score: 0.8363903995266035
 mean cv score: 0.823444056357889
****************************************
```

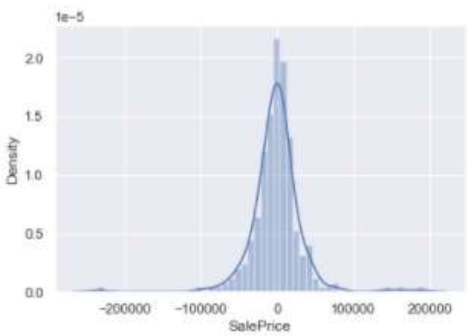
Gradient boosting has the best accuracy, less cv difference with r2 score. We selected it as best model.

```
#Makikng the model
gbr=GradientBoostingRegressor()
gbr.fit(x_train,y_train)
pred=gbr.predict(x_test)
mse=mean_squared_error(y_test,pred)
mae=mean_absolute_error(y_test,pred)
r2=r2_score(y_test,pred)
print(f' mean_absolute_error: {mae}\n mean_squared_error: {mse}\n r2_score: {r2} ')
```

```
 mean_absolute_error: 20269.935780751886
 mean_squared_error: 1085770618.2214737
 r2_score: 0.8406713336596918
```

```
sns.distplot(y_test-pred)
```

```
<AxesSubplot:xlabel='SalePrice', ylabel='Density'>
```



The graph of difference between the original and predicted value is normally distributed.

- ## Interpretation of the Results

.

Model	R2 Score	CV Score
Linear Regression	0.60	0.72
SVR	0.03	0.06
Random Forest Regressor	0.81	0.82
Gradient Boosting Regressor	0.84	0.82

These are the different results of different models.

Gradient boosting is selected as best model and done tuning.

The accuracy increased from 0.84 to 0.86 after tuning.

```
#Makikng the model with tuned parameters
gbr=GradientBoostingRegressor(min_samples_leaf=2, n_estimators=700,min_samples_split = 2 )
gbr.fit(x_train,y_train)
pred=gbr.predict(x_test)
mse=mean_squared_error(y_test,pred)
mae=mean_absolute_error(y_test,pred)
r2=r2_score(y_test,pred)
print(f' mean_absolute_error: {mae}\n mean_squared_error: {mse}\n r2_score: {r2} ')
```

```
 mean_absolute_error: 19200.035634228672
 mean_squared_error: 952343245.6266116
 r2_score: 0.8602507963676194
```

The parameters that played role in increasing the accuracy:

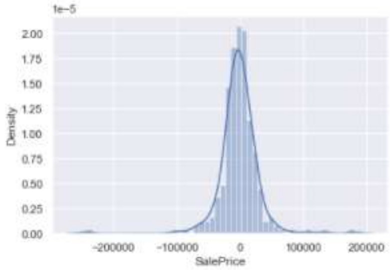
Min_sample_leaf = 2

N_estimators = 700

Min_sample_split=2

```
sns.distplot(y_test-pred)
```

<AxesSubplot:xlabel='SalePrice', ylabel='Density'>




```
sns.regplot(y_test,pred)
```

<AxesSubplot:xlabel='SalePrice'>