FLIP ROBO

HOUSING: PRICE PREDICTION

Submitted by:

Komal Vijay Ghatvilkar

ACKNOWLEDGMENT

The success & outcome of this project were possible by the guidance and support from FlipRobo.

It was not possible to done without research from different machine learning sites on Google.

I referred DataTrained material for more information that helped me completion of the project.

Thank you........!!!

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.

The 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. We 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.

We are required to model the price of houses with the available independent variables. 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. Further, the model will be a good way for the management to understand the pricing dynamics of a new market.

We have done the following analysis of the dataset where we Imported necessary libraries so that we can work on datasets with the Jupyter notebook.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

Data contains 1460 entries each having 81 variables. After reading the dataset I merged both train & test files with concatenate method so that we can do the EDA easily.

```
price = pd.concat([df, df1],ignore_index=True)
```

I checked the description of data with .info() method & find null values contains in dataset.

```
price.info()
```

Found some null values so performed the task and convert or remove the Nan values with mean or some other values.

```
price.isnull().sum()

price['LotFrontage'] = price['LotFrontage'].fillna(price['LotFrontage'].mean())
price['SalePrice'] = price['SalePrice'].fillna(price['SalePrice'].mean())
price['GarageYrBlt'] = price['GarageYrBlt'].fillna(price['GarageYrBlt'].mean())
```

```
price['MasVnrArea'] = price['MasVnrArea'].fillna(price['MasVnrArea'].mean())
price['Alley'] = price['Alley'].fillna('No alley access')
price['BsmtQual'] = price['BsmtQual'].fillna('No Basement')
price['BsmtCond'] = price['BsmtCond'].fillna('No Basement')
price['BsmtExposure'] = price['BsmtExposure'].fillna('No Basement')
price['BsmtFinType1'] = price['BsmtFinType1'].fillna('No Basement')
price['BsmtFinType2'] = price['BsmtFinType2'].fillna('No Basement')
price['FireplaceQu'] = price['FireplaceQu'].fillna('No Fireplace')
price['GarageType'] = price['GarageType'].fillna('No Garage')
price['GarageFinish'] = price['GarageFinish'].fillna('No Garage')
price['GarageQual'] = price['GarageQual'].fillna('No Garage')
price['GarageCond'] = price['GarageCond'].fillna('No Garage')
price['MasVnrType'] = price['MasVnrType'].fillna('None')
price['Electrical'] = price['Electrical'].fillna('Mixed')
```

I removed some unnecessary columns which was containing mostly Nan values only. Those are :

```
price1=price.drop(['Id','PoolQC','Fence','MiscFeature'],axis=1)
```

To perform further task tried with conversion of String data into Integers for equalize the data type for process with LabelEncoder.

```
from sklearn.preprocessing import LabelEncoder
LE=LabelEncoder()
```

After .describe() done found the statistical description of data & with the correlation among all the columns checked the correlation and found most of the data is positively correlated with each other.

```
price1.describe()
price1.corr()
```

In data visualization done the following visualizations :
First used Correlation Matrix for showing the correlation between all columns with Heatmap.

From the output of correlation matrix, we can see that it is symmetrical i.e. the bottom left is same as the top right. It is also observed that most of the variables are positively correlated with each other.

```
corr_mat=price1.corr()
# Size of the canvas
plt.figure(figsize=[50,20])
#Plot Correlation Matrix
sns.heatmap(corr_mat,annot=True) # annot represnts each value encoded in heatmap
plt.title('Correlation Matrix')
plt.show()
```

The result of the Correlation Matrix is on following GitHub link.

https://github.com/komalghatvilkar/Internship/blob/main/HOUSING:%20PRICE%20PREDICTION/correlation%20matrix.png

Second, used Density plot & Histogram both for all dataset for visualizing the data individually.
The visualization plot shows that each variable distributed differently and as we can see some data has categorical values, so used histogram also for better understanding to show the distribution.

```
price1.hist(bins=30,figsize=(40,40))
plt.show()

price1.plot(kind='density',subplots=True,layout=(10,15),sharex=False,fontsize=1,fi
gsize=(40,30))
plt.show()
```

The result of the Density plot & Histogram is on following GitHub link.

https://github.com/komalghatvilkar/Internship/blob/main/HOUSING:%20PRICE%20PREDICTION/Density%20Plot.png

https://github.com/komalghatvilkar/Internship/blob/main/HOUSING:%20PRICE%20PREDICTION/Histogram.png

Firstly checked and removed the skewness of the data. After that checked the outliers if any, found many so tried to remove the outliers. I used boxplot to check the outliers in dataset.

```
price1.skew().sort_values(ascending=False)

from sklearn.preprocessing import power_transform
price_new=power_transform(price1)
price1=pd.DataFrame(price_new,columns=price1.columns)

price1.boxplot(figsize=[20,15])
plt.subplots_adjust(bottom=0.25)
plt.show()

from scipy.stats import zscore
z=np.abs(zscore(price1))
z.shape

threshold=3
print(np.where(z>3))

price_new=price1[(z<3).all(axis=1)]
print(price1.shape)
print(price_new.shape)

loss_percent=(1460-688)/1460*100
print(loss_percent)

price_new.boxplot(figsize=[20,15])
plt.subplots_adjust(bottom=0.25)
plt.show()
```

The result of the Boxplot is on following GitHub link.

https://github.com/komalghatvilkar/Internship/blob/main/HOUSING:%20PRICE%20PREDICTION/Outliers.png

After checking the outliers We split the data into x & y. I have taken the output as the column “SalePrice” because our problem statement is to find the sales price of houses.

```
x=price_new.drop(['SalePrice'],axis=1)
y=price_new['SalePrice']
```

And then did the Train Test Split and find the best accuracy & random state which gives me the following results :-

```
from sklearn.preprocessing import MinMaxScaler
mms=MinMaxScaler()
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
from sklearn.metrics import r2_score
from sklearn.model_selection import train_test_split
for i in range(0,100):

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.20,random_state=i)
        lr.fit(x_train,y_train) # Fitting the data will train the model
        pred_train=lr.predict(x_train)  #  Predicting  the  data  #  Predicted  traget
variable
        pred_test=lr.predict(x_test)
        print(f'At    Random    State    {i},    the    training    accuracy    is    :-
{r2_score(y_train,pred_train)}')
        print(f'At    Random    State    {i},    the    training    accuracy    is    :-
{r2_score(y_test,pred_test)}')
        print("\n")
```

```
At Random State 0, the training accuracy is :- 0.7317075043890658
At Random State 0, the training accuracy is :- 0.5190497154704797

At Random State 1, the training accuracy is :- 0.7295174908018054
At Random State 1, the training accuracy is :- 0.593101328593335

At Random State 2, the training accuracy is :- 0.7251525569570767
At Random State 2, the training accuracy is :- 0.6074742314092839

At Random State 3, the training accuracy is :- 0.7054591694963075
At Random State 3, the training accuracy is :- 0.6920882422571926

At Random State 96, the training accuracy is :- 0.6943592901090109
At Random State 96, the training accuracy is :- 0.746720460443741

At Random State 97, the training accuracy is :- 0.7014732795202088
At Random State 97, the training accuracy is :- 0.7226833684268036

At Random State 98, the training accuracy is :- 0.7148857068330131
At Random State 98, the training accuracy is :- -475201371191.58545

At Random State 99, the training accuracy is :- 0.6927298467186471
At Random State 99, the training accuracy is :- 0.7203640940063292
```

Then performed the model building and used best possible model building technique for regression dataset that is Linear Regression to perform The task.

Next did Cross Validation with “cross_val_score” for the models used & it shows the output :-

```
Train_accuracy = r2_score(y_train,pred_train)
Test_accuracy = r2_score(y_test,pred_test)

from sklearn.model_selection import cross_val_score
for j in range(2,10):
    cv_score=cross_val_score(lr,x,y,cv=j)
    cv_mean=cv_score.mean()
    print(f'At cross fold {j} the cv score is  {cv_mean} and accuracy score for training is {Train_accuracy} and accuracy score for testing is {Test_accuracy}')
    print('\n')
```

At cross fold 2 the cv score is -9.977252231451208e+16 and accuracy score for training is 0.6927298467186471 and accuracy score for testing is 0.7203640940063292

At cross fold 3 the cv score is -44630928458645.88 and accuracy score for training is 0.6927298467186471 and accuracy score for testing is 0.7203640940063292

At cross fold 4 the cv score is -569633217756058.8 and accuracy score for training is 0.6927298467186471 and accuracy score for testing is 0.7203640940063292

At cross fold 5 the cv score is 0.6011789544393826 and accuracy score for training is 0.6927298467186471 and accuracy score for testing is 0.7203640940063292

At cross fold 6 the cv score is -3.495669644565994e+45 and accuracy score for training is 0.6927298467186471 and accuracy score for testing is 0.7203640940063292

At cross fold 7 the cv score is -9.860764066684406e+46 and accuracy score for training is 0.6927298467186471 and accuracy score for testing is 0.7203640940063292

At cross fold 8 the cv score is -1.0721906295429018e+32 and accuracy score for training is 0.6927298467186471 and accuracy score for testing is 0.7203640940063292

At cross fold 9 the cv score is -1.600020811684443e+21 and accuracy score for training is 0.6927298467186471 and accuracy score for testing is 0.7203640940063292

Then with the matplotlib.pyplot showed the distribution of data & the result shows best fit line & relationship between two variables.

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8,6))

plt.scatter(x=y_test,y=pred_test,color='blue')

plt.plot(y_test,y_test,color='yellow')

plt.xlabel('Average price', fontsize=14)

plt.ylabel('Predicted price', fontsize=14)

plt.title('Linear Regression', fontsize=18)

plt.show()
```

Then checked the regularization with GridSearchCV & With Lasso technique to perform regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model and found the cross_val_score.

```
from sklearn.model_selection import GridSearchCV

from sklearn.model_selection import cross_val_score

from sklearn.linear_model import Lasso

parameters={'alpha':[.0001,.001,.01,.1,1,10],'random_state':list(range(0,20))}

ls=Lasso()

clf=GridSearchCV(ls,parameters)

clf.fit(x_train,y_train)

print(clf.best_params_)

ls=Lasso(alpha=0.01,random_state=0)

ls.fit(x_train,y_train)

ls.score(x_train,y_train)

pred_ls=ls.predict(x_test)

lss=r2_score(y_test,pred_ls)

Lss

cv_score=cross_val_score(ls,x,y,cv=5)

cv_mean=cv_score.mean()

cv_mean*100
```

With AdaBoostRegressor assemble technique to achieve best accuracy of dataset.

```
from sklearn.ensemble import AdaBoostRegressor
AD = AdaBoostRegressor()
AD.fit(x_train,y_train)
AD.score(x_test,y_test)
```

Finally came to the Conclusion as per the results found those are the model is showing the exact result for the dataset with 68% accuracy.

Please find the GitHub links for Pyplot of Linear regression to refer.

https://github.com/komalghatvilkar/Internship/blob/main/HOUSING:%20PRICE%20PREDICTION/Linear%20Regression%20Plot.png

Please find the GitHub links for Jupyter Notebook Solution of dataset to refer.

https://github.com/komalghatvilkar/Internship/blob/main/HOUSING:%20PRICE%20PREDICTION/HOUSING%20-%20PRICE%20PREDICTION.ipynb

The results shows that the dataset is correct 68% & we can rely on the data accordingly.

Thank you...!!