HOUSING PRICE PREDICTION

Submitted by:

Nazia Sultana

ACKNOWLEDGMENT

I would like to express my sincere thanks of gratitude to my mentors from Data Trained academy and Flip Robo Technologies Bangalore for letting me work on this project. Their suggestions and directions have helped me in the completion of this project successfully.

All the required information & the dataset are provided by Flip Robo Technologies.

INTRODUCTION

- Business Problem Framing
- Thousands of houses are sold every day. There are some questions every buyer asks himself like: What is the actual price that this house deserves? Am I paying a fair price? Also Is it the location? Is it the overall quality of the house? Is it the size? Could it be sold at a good price in future? All these questions come in to our mind when we decide to purchase a house. In this study, a machine learning model is proposed to predict a house price based on data related to the house (its size, the year it was built in, etc.).
- During the development and evaluation of our model, we will show the code used for each step followed by its output. This will facilitate the reproducibility of our work

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. 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 project endeavours to extensive data analysis and implementation of different machine learning techniques in python for having the best model with most important features of a house on insight of both business value and realistic perspective.

- Review of Literature

The relationship between house prices and the economy is an important motivating factor for predicting house prices (Pow, Janulewicz, & Liu, 2014). There is no accurate measure of house prices (Pow, Janulewicz, & Liu, 2014).

Pow states that Real Estate property prices are linked with economy (Pow, Janulewicz, & Liu, 2014). He also states there is no accurate measure of house prices. A property's value is important in real estate transactions. Pow tries to predict the sold and asking prices of real estate values without bias to help both buyers and sellers make their decisions.

A property's value is important in real estate transactions. Housing market is important for economic activities (Khamis & Kamarudin, 2014). Traditional housing price prediction is based on cost and sale price comparison. So, there is a need for building a model to efficiently predict the house price.

Motivation for the Problem Undertaken

I have gone thorough many projects before, but this project has given me an idea to handle large number of attributes. By doing this project I have got an idea about how to deal with data exploration where I have used all my analyzation skills to predict the house price using ML models. The model will be a good way for both buyers and sellers to understand the pricing dynamic of a new market.

The main objectives of this study are as follows:

• To apply data pre-processing and preparation techniques in order to obtain clean data.
• To build machine learning models able to predict house price based on house features.
• To analyse and compare models' performance in order to choose the best model.
By processing the above objects, I will be able to find which variables are important to predict the price of house?
And how do these variables describe the price of the house?
The relation between house prices and the economy is an important factor for predicting house prices.

Analytical Problem Framing

- Mathematical/ Analytical Modelling of the Problem

The house price model is based on a demand function for housing services and a standard life-cycle model of utility for a representative household.

This is a common approach in academic research into house prices. The study is to predict the sale price of the house and analysing which features are important and how they contribute in the prediction. There are two datasets. One is train dataset which is supervised and another one is test dataset which is unsupervised. The target variable is "SalePrice" and it is a regression type problem.
I have used train dataset to build machine learning models and then by using this model I made prediction for the test dataset.
I have observed some columns having more than 85% of zero entries and 70% of null values so, I decided to drop those columns. I have performed both univariate and bivariate analysis to analyse the sale price of the house. I have analysed the categorical and

numerical features using categorical plots and numerical plots respectively to get better insights from the data. In this project I have done various mathematical and statistical analysis such as describing the statistical summary of the columns, feature engineering, treating null values, removing outliers, skewness, encoding the data etc. Checked for correlation between the features and visualized it using heat map. Also, I built many regression algorithms while building machine learning models, used hyper tuning method for best model and saved the best model. Finally, I predicted the sale price of the house using the saved trained model.

- Data Sources and their formats

A US-based housing company named Surprise Housing has collected the dataset from the sale of houses in Australia and the data is provided by Flip Robo Company and it is in csv format.
There are 2 data sets:
1. Train dataset
2. Test dataset

Train dataset will be used for training the machine learning models. The dataset contains 1168 rows and 81 columns, out of 81 columns, 80 are independent variables and remaining 1 is dependent variable (SalePrice).

Test dataset contains all the independent variables, but not the target variable. We will apply the trained model to predict the target variable for the test data. The dataset contains 292 rows and 80 columns.

The dataset contains both numerical and categorical data. Numerical data contains both continuous and discrete variables and categorical data contains both nominal and ordinal variables.

• I concatenate both train and test data, and decided to process both the data together.

- Data Pre-processing

Firstly, I have imported the necessary libraries and imported both train and test datasets which were in csv format. And process both datasets simultaneously.

I have done some statistical analysis like checking shape, column names, data types of the features, info about the features, value counts etc for both train and test data.

I have dropped the columns “Id” and “Utilities” from both the datasets. Since Id is the unique identifier which contains unique value throughout the data also all the entries in Utilities column were unique. They had no significance impact on the prediction.

While looking into the value count function I found some of the columns having more than 85% of zero values so, I dropped those columns from both the datasets as they might create skewness which will impact my model.

Also, I have done some feature extraction as the datasets contained sometime variables like YearBuilt, YearRemodAdd, GarageYrBlt and YrSold. Converting them into age seem more meaningful as they offer more information about the longevity of the features. So, I have extracted age information from the datetime variables by taking the difference in year between the year the house was built and year the house was sold and dropped the year columns.

I checked the null values and found them in some of the columns. So, I imputed null values present in categorical and numerical columns using mode and mean methods respectively. I found some columns having more than 80% of null values so, I dropped those columns to overcome with the skewness.

Described statistical summary of both train and test datasets.

Visualized each feature using seaborn and matplotlib libraries by plotting several categorical and numerical plots.

Identified outliers using box plots in both datasets. I tried to remove them using both Zscore and IQR method and got huge data loss of around 19%and 35% respectively, so removed outliers using percentile method by setting data loss to 2%.

Checked for skewness and removed skewness in numerical columns using power transformation method (yeo-johnson). Also dropped KitchenAbvGr columns as it contains zero values throughout the data after using power transformation.

Encoded both train and test data frames using Ordinal Encoder. Also replaced some categorical columns having ratings by numbers based on specific condition.

Used Pearson's correlation coefficient to check the correlation between label and features.

Scaled the datasets using Standard Scalar method and used regression algorithms to build ML models.
All these steps were performed to both train and test datasets together.

- Data Inputs- Logic- Output Relationships

To analyse the relation between features and target I have done EDA where I analysed the relation using many plots like bar plot, reg plot, scatter plot, line plot, swarm plot, strip plot, violin plot etc. And found some of the columns like OverallQual, TotalRmsAbvGrd, FullBath, GarageCars etc have strong positive linear relation with the label.

I have checked the correlation between the target and features using heatmap and bar plot. Where I got the positive and negative correlation between the label and feature

Features having high Positive correlation with label
- OverallQual
- GrLivArea
- ExterQual
- BsmtQual
- GarageCars
- GarageArea
- TotalBsmtSF
- 1stFlrSF
- FullBath
- TotRmsAbvGrd
- KitchenQual

Features having high Negative correlation with label
- Heating
- MSZoning
- LotShape
- BsmtExposure
- GarageType
- AgeRemod
- AgeGarage
- AgeBuilt
- GarageFinish

- Hardware and Software Requirements and Tools Used

To build the machine learning projects it is important to have the following hardware and software requirements and tools.

❖ Hardware required:

- Processor: core i5 or above
- RAM: 8 GB or above
- ROM/SSD: 250 GB or above

❖ Software required:

- Anaconda 3- language used Python 3

import numpy as np
It is defined as a Python package used for performing the various numerical computations and processing of the multidimensional and single dimensional array elements. The calculations using numpy arrays are faster than the normal Python array

Import pandas as pd:
Pandas is a Python library that is used for faster data analysis, data cleaning and data pre-processing. The data-frame term is coming from Pandas only.

import matplotlib.pyplot as plt:
Matplotlib and Seaborn acts as the backbone of data visualization through Python. Matplotlib: It is a Python library used for plotting graphs with the help of other libraries like Numpy and Pandas. It is a powerful tool for visualizing data in Python. It is used for creating statical interferences and plotting 2D graphs of arrays

```
from scipy.stats import zscore
from sklearn.preprocessing import PowerTransformer
from sklearn.preprocessing import LabelEncoder
from  sklearn.preprocessing import StandardScaler
```

With the above sufficient library, we can perform pre-processing and data cleaning. For building my ML models Below libraries are required

In [1]:
```
#importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
```

```
from sklearn import metrics
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
```

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

1.I have used imputation methods to treat the null values.

2.Used zscore method to remove outliers.

3.Removed skewness using power transformation (yeo-Johnson) method.

4.I have used correlation coefficient method to check the correlation between the dependent and independent variables.

5. I have scaled the data using Standard Scalar method to overcome with the data biasness.

6.Used many machine Learning models to predict the sale price of the house.

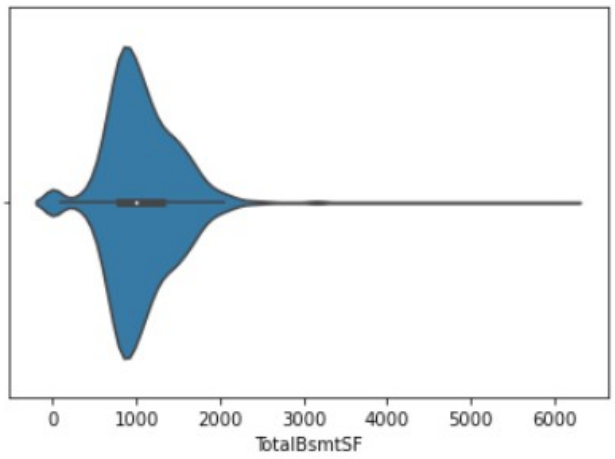
Visualizations:

I have analysed the data using both univariate and bivariate analysis to visualize the data. In univariate analysis I have used pie plots, count plots and distribution plot and in bivariate analysis I have used bar plots for categorical columns and used reg plots, scatter plots, strip plots, swarm plots, violin plots and line plot to visualize numerical columns. These plots have given good pattern. Here I will be showing only bivariate analysis to get the better insights of relation between label and the features

```
df=df.drop('BsmtFinSF2',axis=1)
```

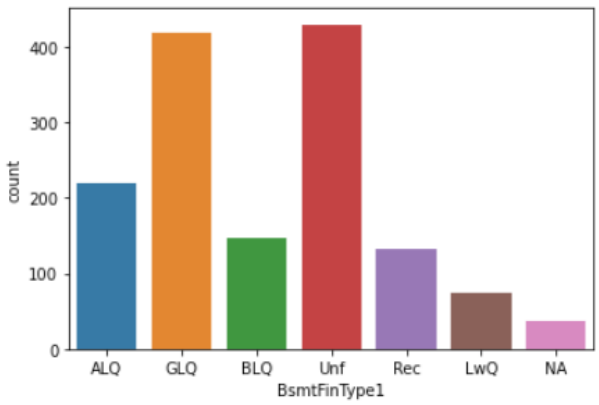
```
sns.violinplot(df['TotalBsmtSF'])
```

```
<AxesSubplot:xlabel='TotalBsmtSF'>
```



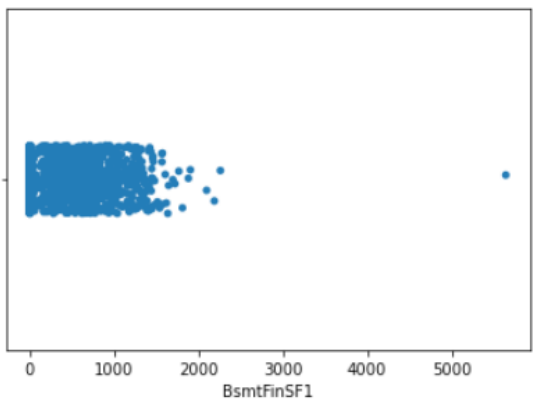

```
sns.countplot(df['BsmtFinType1'])
```

```
<AxesSubplot:xlabel='BsmtFinType1', ylabel='count'>
```



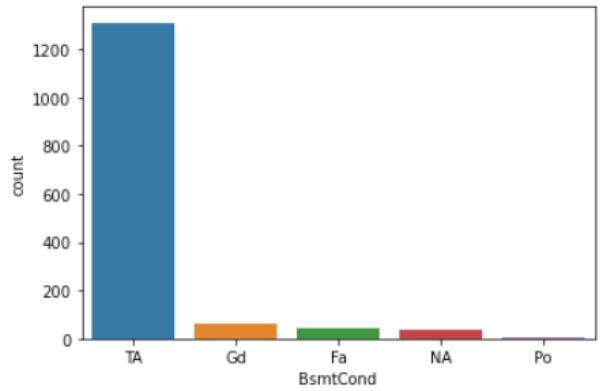

```
sns.stripplot(df['BsmtFinSF1'])
```

```
<AxesSubplot:xlabel='BsmtFinSF1'>
```



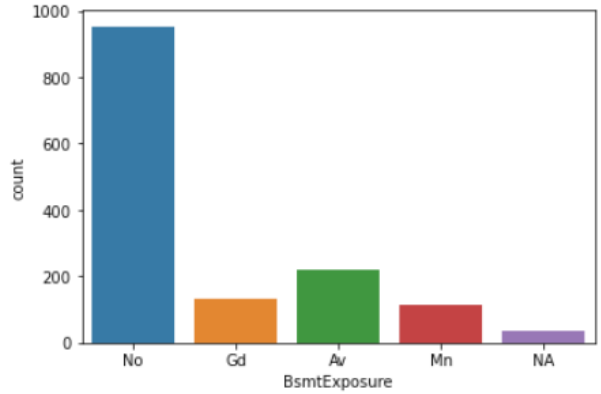
```
sns.countplot(df['BsmtCond'])
```

```
<AxesSubplot:xlabel='BsmtCond', ylabel='count'>
```



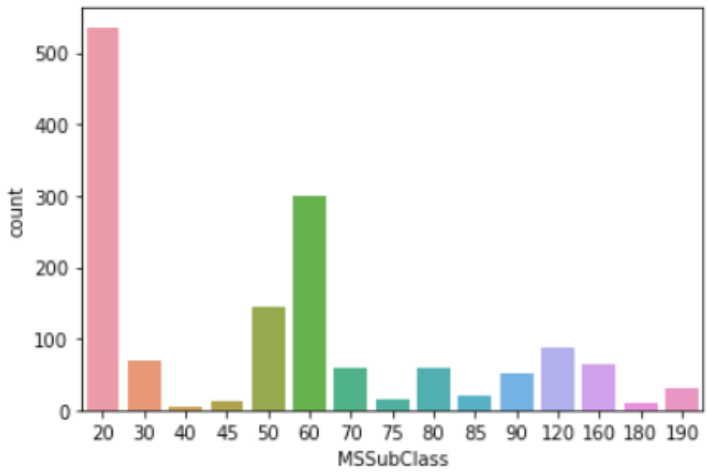

```
sns.countplot(df['BsmtExposure'])
```

```
<AxesSubplot:xlabel='BsmtExposure', ylabel='count'>
```



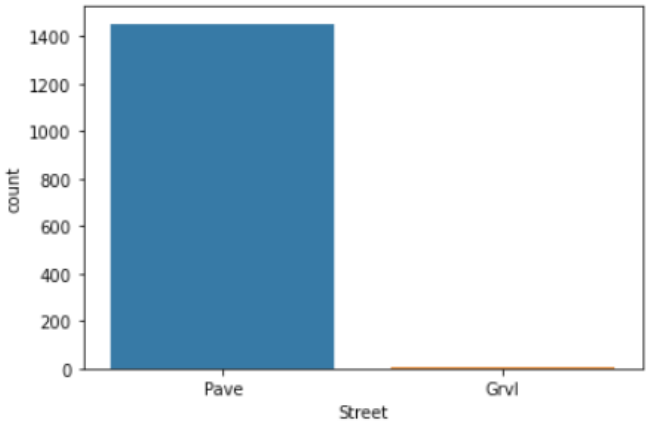

```
sns.countplot(df['MSSubClass'])
```

```
<AxesSubplot:xlabel='MSSubClass', ylabel='count'>
```



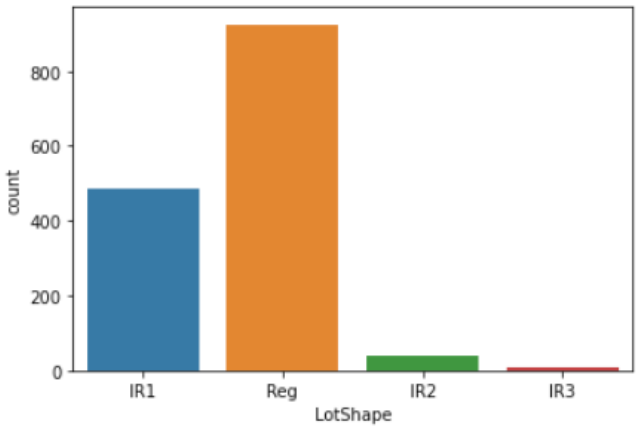
```
sns.countplot(df['Street'])
```

```
<AxesSubplot:xlabel='Street', ylabel='count'>
```




```
sns.countplot(df['LotShape'])
```

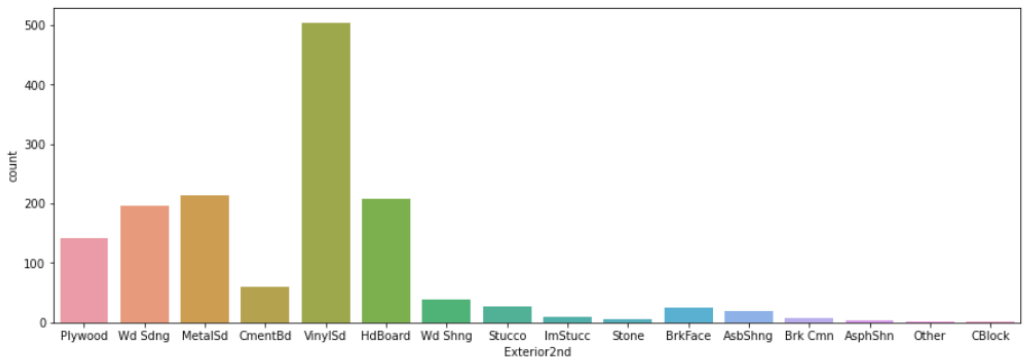
```
<AxesSubplot:xlabel='LotShape', ylabel='count'>
```



Exterior1st

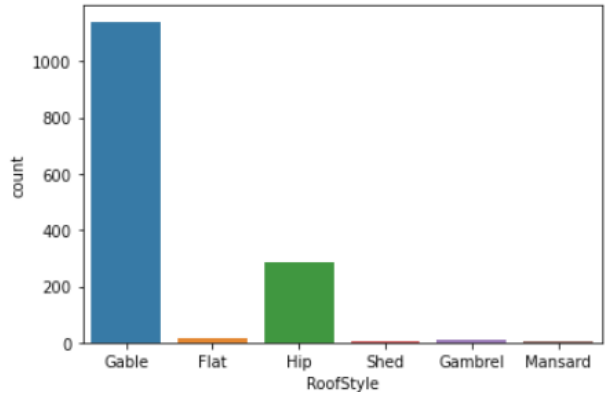
```
plt.figure(figsize=(15,5))
sns.countplot(df['Exterior2nd'])
```

```
<AxesSubplot:xlabel='Exterior2nd', ylabel='count'>
```



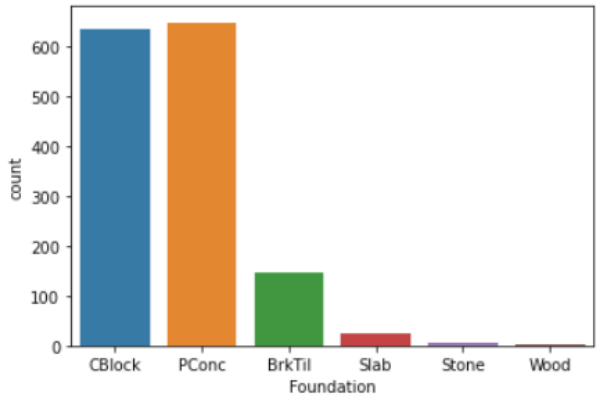
```
sns.countplot(df['RoofStyle'])
```

```
<AxesSubplot:xlabel='RoofStyle', ylabel='count'>
```



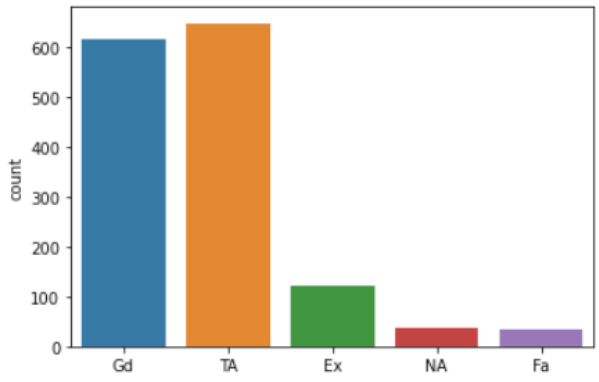

```
sns.countplot(df['Foundation'])
```

```
<AxesSubplot:xlabel='Foundation', ylabel='count'>
```



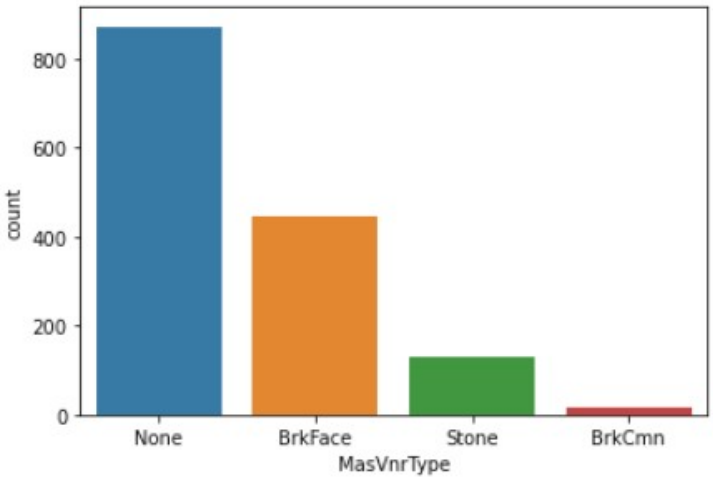

```
sns.countplot(df['BsmtQual'])
```

```
<AxesSubplot:xlabel='BsmtQual', ylabel='count'>
```



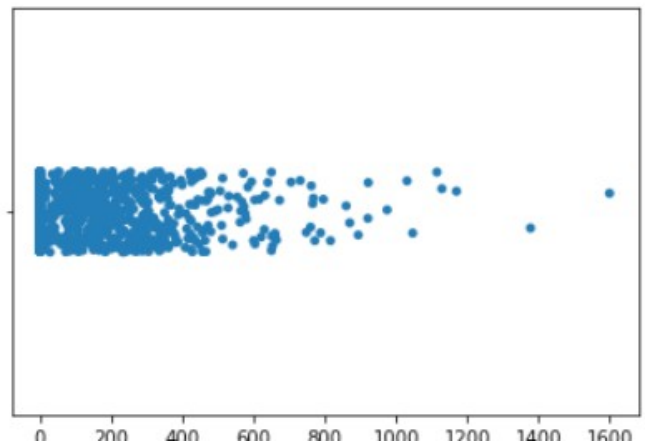
```
sns.countplot(df['MasVnrType'])
```

```
<AxesSubplot:xlabel='MasVnrType', ylabel='count'>
```



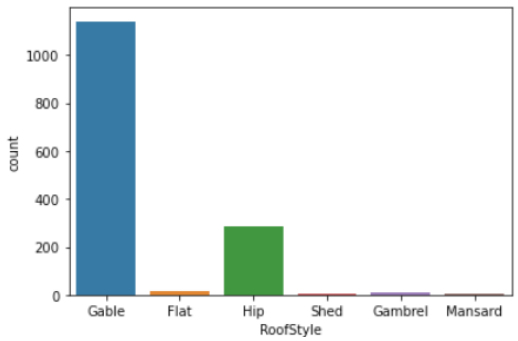

```
sns.stripplot(df['MasVnrArea'])
```

```
<AxesSubplot:xlabel='MasVnrArea'>
```




```
sns.countplot(df['RoofStyle'])
```

```
<AxesSubplot:xlabel='RoofStyle', ylabel='count'>
```



Testing of Identified Approaches (Algorithms)

In this problem SalePrice is my target variable which is continuous in nature, from this I can conclude that it is a regression type problem hence I have used following regression algorithms to predict the sale price of the house. After the pre-processing and data cleaning I left with 67 columns including target and I used these features for prediction.

1. Linear Regression
2. Lasso Regressor
3. DecisionTree Regressor
4. Random Forest Regressor

Linear Regression:

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables. Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable

Created linear regression model and getting 90.2% R2 score using this model. From the reg plot I can observe the sales price of the house

```
]: lr.score(x_test,y_test)
```

```
]: 0.9025712673438843
```

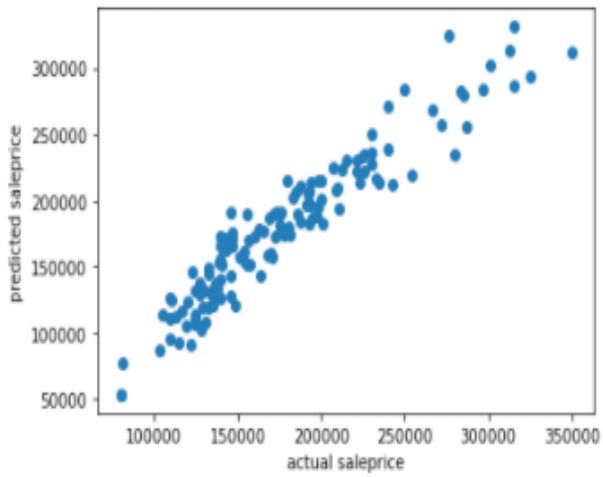
```
]: y_pred_lr=lr.predict(x_test)
```

```
]: # comparing actual values with predicted values

actual_vs_pred = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred_lr})
actual_vs_pred.sample(10)
```

]:

	Actual	Predicted
1088	187500.0	210724.422769
55	230000.0	236682.672769
687	173000.0	191254.422769
1141	193000.0	206188.672769
737	172500.0	172931.422769
785	143000.0	162459.172769
1161	225000.0	233709.672769
427	208500.0	208839.672769
261	156932.0	169381.172769
470	147000.0	175666.922769

```
]: plt.scatter(y_test,y_pred_lr)
plt.xlabel('actual saleprice')
plt.ylabel('predicted saleprice')
plt.show()
```



2.Lasso Regression

It performs L1 regularization and it is used over regression methods for a more accurate prediction. This model uses shrinkage. Shrinkage is where data values are shrunk towards a central point as the mean. This particular type of regression is well-suited for models showing high levels of multicollinearity. It is used when we have greater number of features because it automatically performs feature selection

Lasso regressor model and getting 90% R2 score using this model

```
parameters={'alpha':[0.0001,0.001,0.01,0.1,1,10],'random_state':list(range(0,10))}
```

```
ls=Lasso()
```

```
clf=GridSearchCV(ls,parameters)
clf.fit(x_train,y_train)
clf.best_params_
```

```
{'alpha': 10, 'random_state': 0}
```

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

```
ls.fit(x_train,y_train)
```

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

```
ls.score(x_train,y_train)
```

```
0.9151387909982901
```

```
pred_ls=ls.predict(x_test)
```

```
lss=r2_score(y_test,pred_ls)
```

```
lss
```

```
0.9022326894752659
```

3.DecisionTree Regressor

```
#using Decisiontree regressor
```

```
dt.fit(x_train,y_train)
```

```
DecisionTreeRegressor()
```

```
dt.score(x_test,y_test)
```

```
0.7949216744261004
```

```
y_pred_dt=dt.predict(x_test)
```

```
y_predict_dt=dt.predict(x_test)
```

```
print('MAE:', metrics.mean_absolute_error(y_test,y_predict_dt))
print('MSE:', metrics.mean_squared_error(y_test, y_predict_dt))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test,y_predict_dt)))
```

```
MAE: 18156.97014925373
MSE: 646354715.4626865
RMSE: 25423.507143246123
```

By using DecisionTree regressor model getting 80% R2 score

4. Random Forest Regressor:

Random forest is an ensemble technique capable of performing both regression and classification tasks with use of multiple decision trees and a technique called Bootstrap Aggregation. It improves the predictive accuracy and control over-fitting

```
]: #using random forest regressor

3]: rfc.fit(x_train,y_train)

3]: RandomForestRegressor()

4]: rfc.score(x_test,y_test)

4]: 0.9213154430689763

5]: y_predict_rfc=rfc.predict(x_test)

6]: print('MAE:', metrics.mean_absolute_error(y_test,y_predict_rfc))
    print('MSE:', metrics.mean_squared_error(y_test, y_predict_rfc))
    print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test,y_predict_rfc)))

    MAE: 11627.061716417911
    MSE: 247993708.08268437
    RMSE: 15747.815978182001
```

Created Random Forest Regressor model and getting 92% R2 score using this model.

Run and Evaluate selected models

From the difference between R2 score and Cross Validation score I can conclude that Random forest Regressor as my best fitting model as it is giving less difference compare to other models. Let's perform Hyperparameter tuning to increase the model accuracy

```
]: #using random forest regressor
```

```
3]: rfc.fit(x_train,y_train)
```

```
3]: RandomForestRegressor()
```

```
4]: rfc.score(x_test,y_test)
```

```
4]: 0.9213154430689763
```

```
5]: y_predict_rfc=rfc.predict(x_test)
```

```
6]: print('MAE:', metrics.mean_absolute_error(y_test,y_predict_rfc))
    print('MSE:', metrics.mean_squared_error(y_test, y_predict_rfc))
    print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test,y_predict_rfc)))
```

```
MAE: 11627.061716417911
MSE: 247993708.08268437
RMSE: 15747.815978182001
```

```
from sklearn.model_selection import cross_val_score
```

```
cross_val_score(lr,x_scaled,y,cv=5)
```

```
array([0.88539436, 0.90822606, 0.84058405, 0.91803464, 0.8610659 ])
```

```
cross_val_score(lr,x_scaled,y,cv=5).mean()
```

```
0.8826610023244953
```

```
cross_val_score(dt,x_scaled,y,cv=5)
```

```
array([0.76423263, 0.77025496, 0.74786052, 0.48163939, 0.64808254])
```

```
cross_val_score(dt,x_scaled,y,cv=5).mean()
```

```
0.6804935879283119
```

```
cross_val_score(rfc,x_scaled,y,cv=5)
```

```
array([0.90475661, 0.89242036, 0.87022446, 0.84742015, 0.84480996])
```

```
cross_val_score(rfc,x_scaled,y,cv=5).mean()
```

```
0.8728881952617273
```

Hyper Parameter Tuning

```
            "max_features" :['auto','sqrt','log2'],
            }
```

```
rfc=RandomForestRegressor()
```

```
grid_search=GridSearchCV(rfc,parameters)
```

```
grid_search.fit(x_train,y_train)
```

```
GridSearchCV(estimator=RandomForestRegressor(),
             param_grid={'criterion': ['mse', 'mae'],
                         'max_features': ['auto', 'sqrt', 'log2']})
```

```
best_parameters=grid_search.best_params_
```

```
best_parameters
```

```
{'criterion': 'mse', 'max_features': 'auto'}
```

```
clf=RandomForestRegressor(criterion='mse' ,max_features='auto')
```

```
clf.fit(x_train,y_train)
```

```
RandomForestRegressor(criterion='mse')
```

```
y_pred=clf.predict(x_test)
```

```
plt.scatter(y_test,y_pred)
```

```
<matplotlib.collections.PathCollection at 0x1dceeef1850>
```

I have used GridSearchCV to get the best parameters of Random Forest Regressor. And used all the obtained parameters to get the accuracy of final model

Saving the final model and predicting the saved model.

```
# Saving model
import joblib
joblib.dump(best_parameters,'housing_projects.pkl')
```

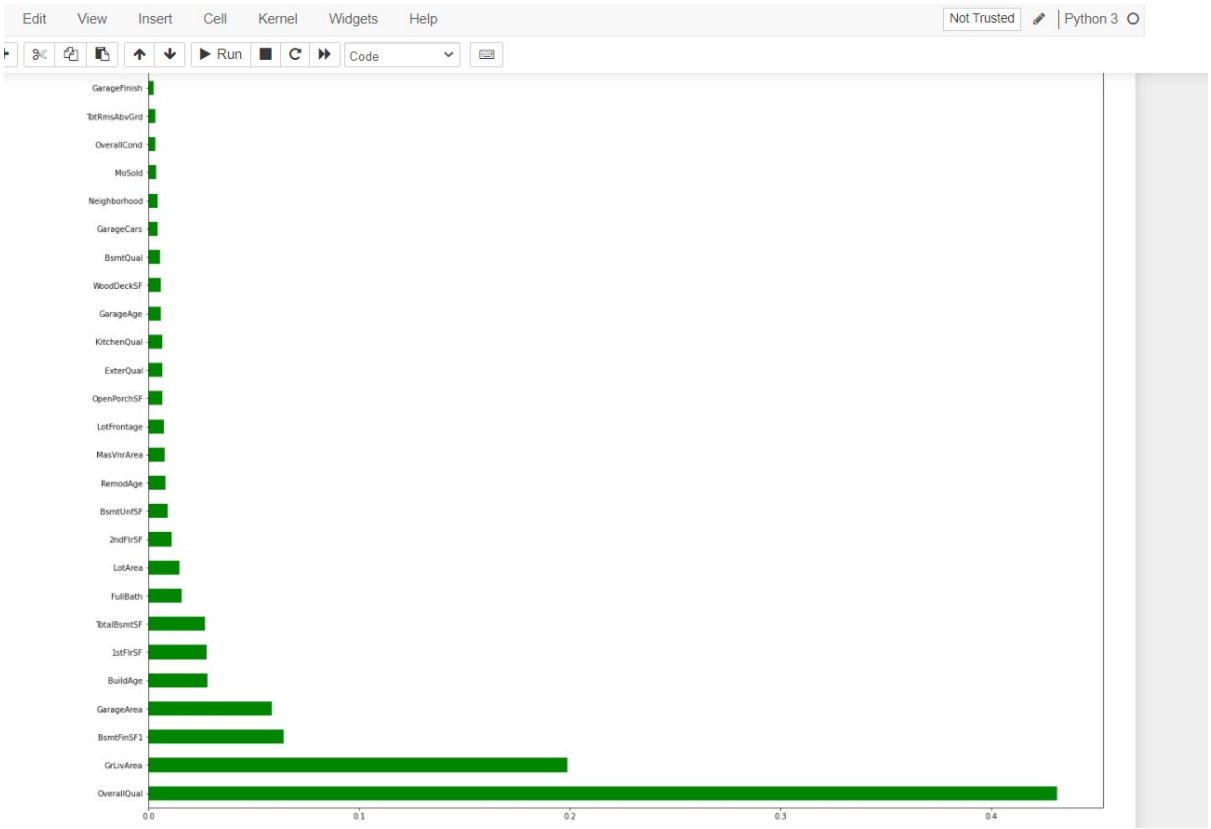
```
['housing_projects.pkl']
```

```
Conclusion:

Which variables are important to predict the price of house?

Overall Quality is the most contributing and highest positive impacting feature. Also, the features like GarageArea, LotArea,
1stFlrSF, TotalBsmtSF etc have some what linear relation with the price variable.
```

In [278]:
```
# Which variables are important to predict the price of variable?
# Feature Importance
# Lets chkeck the feature importance using Random Forest Regressor

RFR = RandomForestRegressor()
RFR.fit(x_train, y_train)
importances = pd.DataFrame({'Features':x.columns, 'Importance':np.round(RFR.feature_importances_,3)})
importances = importances.sort_values('Importance', ascending=False).set_index('Features')
importances
```

Out[278]:

Features	Importance
OverallQual	0.431
GrLivArea	0.199
BsmtFinSF1	0.064
GarageArea	0.059
BuildAge	0.028
1stFlrSF	0.028
TotalBsmtSF	0.027
FullBath	0.016
LotArea	0.015
2ndFlrSF	0.011
BsmtUnfSF	0.009



Key Metrics for success in solving problem under consideration

The essential step in any machine learning model is to evaluate the accuracy and determine the metrics error of the model.
I have used the following metrics for my model evaluation:

Mean absolute error (MAE): MAE is a popular error metric for regression problems which give magnitude of absolute difference between actual and predicted values

Mean Squared Error (MSE): MSE is a most used and very simple metric with a little bit of change in mean absolute error. Mean squared error states that finding the squared difference between actual and predicted value. We perform squared to avoid the cancellation of negative terms and it is the benefit of MSE.

Root Mean Squared Error (RMSE): RMSE is an extension of the mean squared error. The square root of the error is calculated, which means that the units of the RMSE are the same as the original units of the target value that is being predicted.

R2 Score: I have used R2 score which gives the accurate value for the models used. On the basis of R2 score I have created final model.

- Visualizations

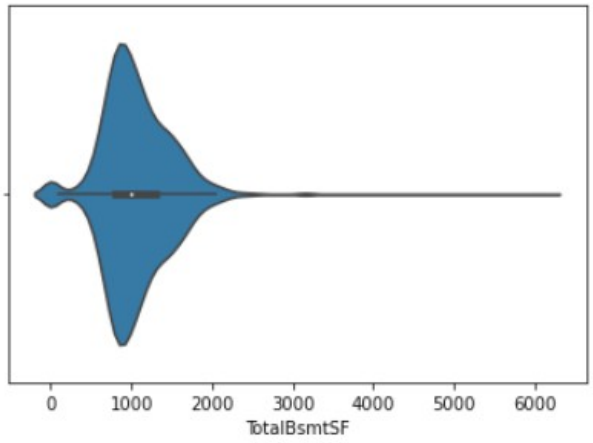
Mention all the plots made along with their pictures and what were the inferences and observations obtained from those. Describe them in detail.

If different platforms were used, mention that as well.

```
df=df.drop('BsmtFinSF2',axis=1)
```

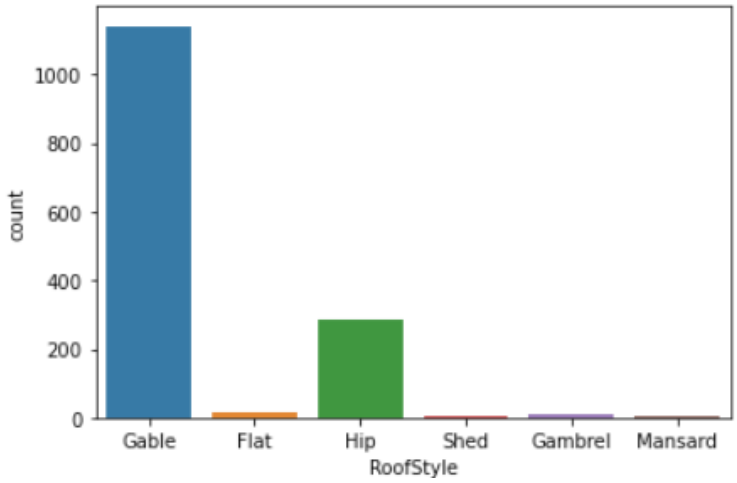
```
sns.violinplot(df['TotalBsmtSF'])
```

```
<AxesSubplot:xlabel='TotalBsmtSF'>
```



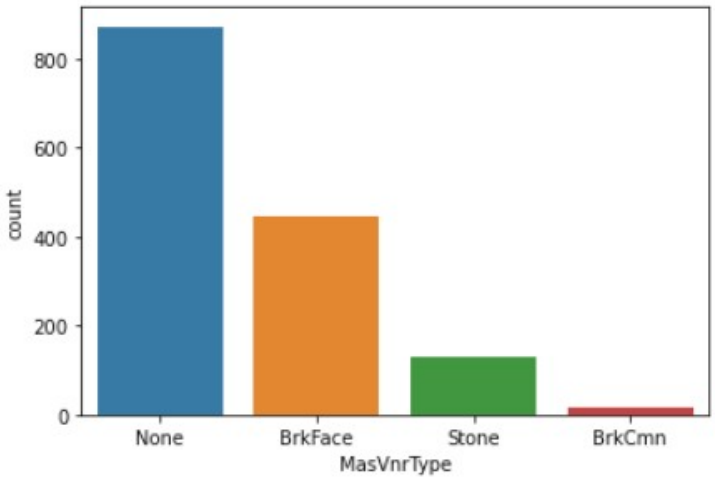

```
sns.countplot(df['RoofStyle'])
```

```
<AxesSubplot:xlabel='RoofStyle', ylabel='count'>
```



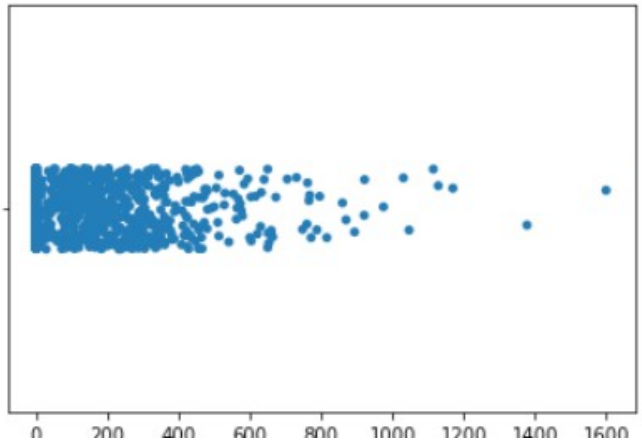
```
sns.countplot(df['MasVnrType'])
```

```
<AxesSubplot:xlabel='MasVnrType', ylabel='count'>
```



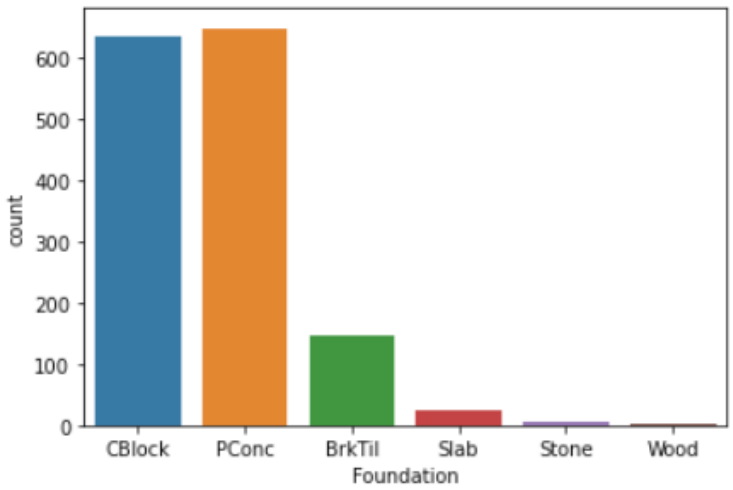

```
sns.stripplot(df['MasVnrArea'])
```

```
<AxesSubplot:xlabel='MasVnrArea'>
```



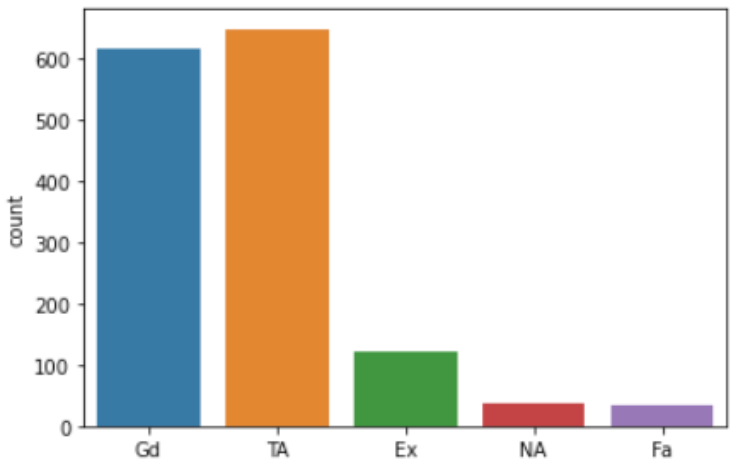
```
sns.countplot(df['Foundation'])
```

```
<AxesSubplot:xlabel='Foundation', ylabel='count'>
```



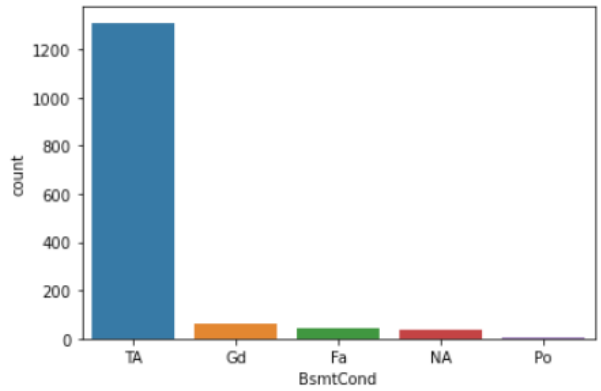

```
sns.countplot(df['BsmtQual'])
```

```
<AxesSubplot:xlabel='BsmtQual', ylabel='count'>
```



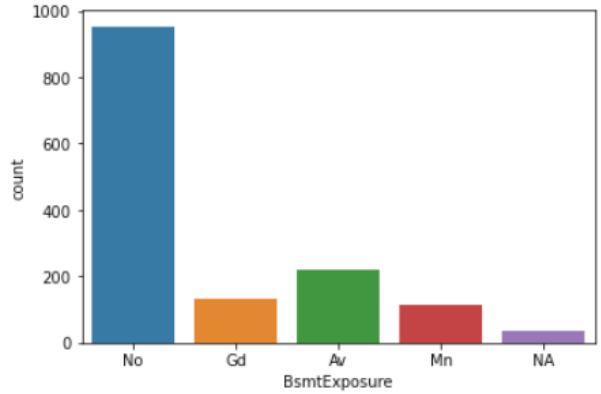
```
sns.countplot(df['BsmtCond'])
```

```
<AxesSubplot:xlabel='BsmtCond', ylabel='count'>
```

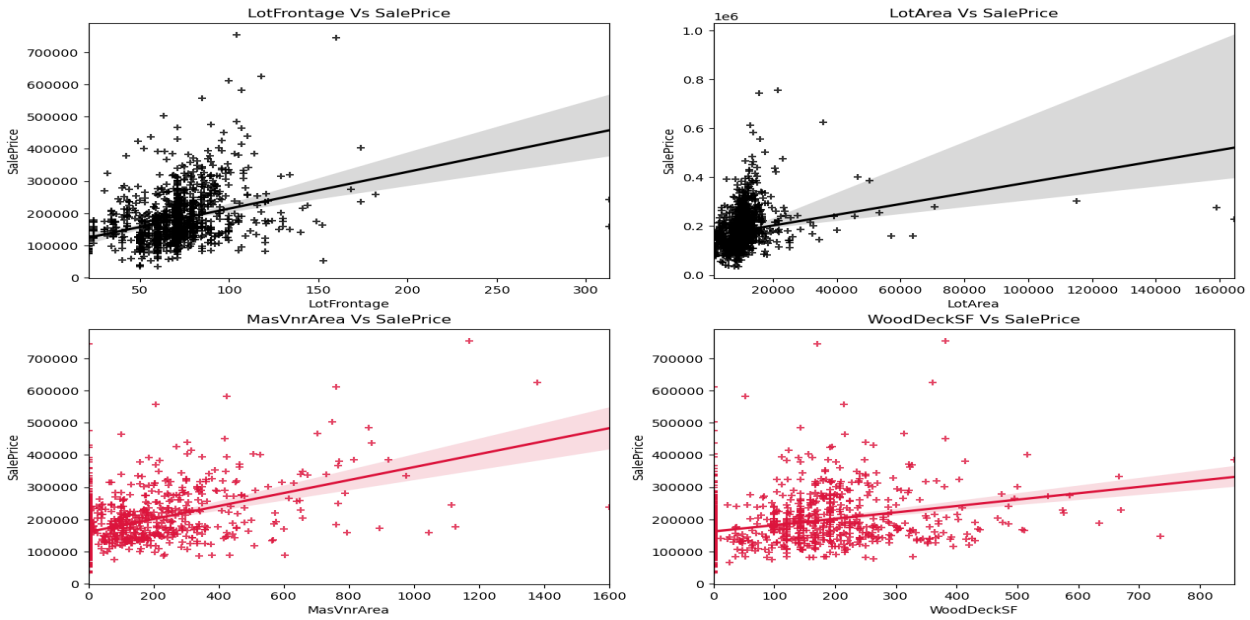



```
sns.countplot(df['BsmtExposure'])
```

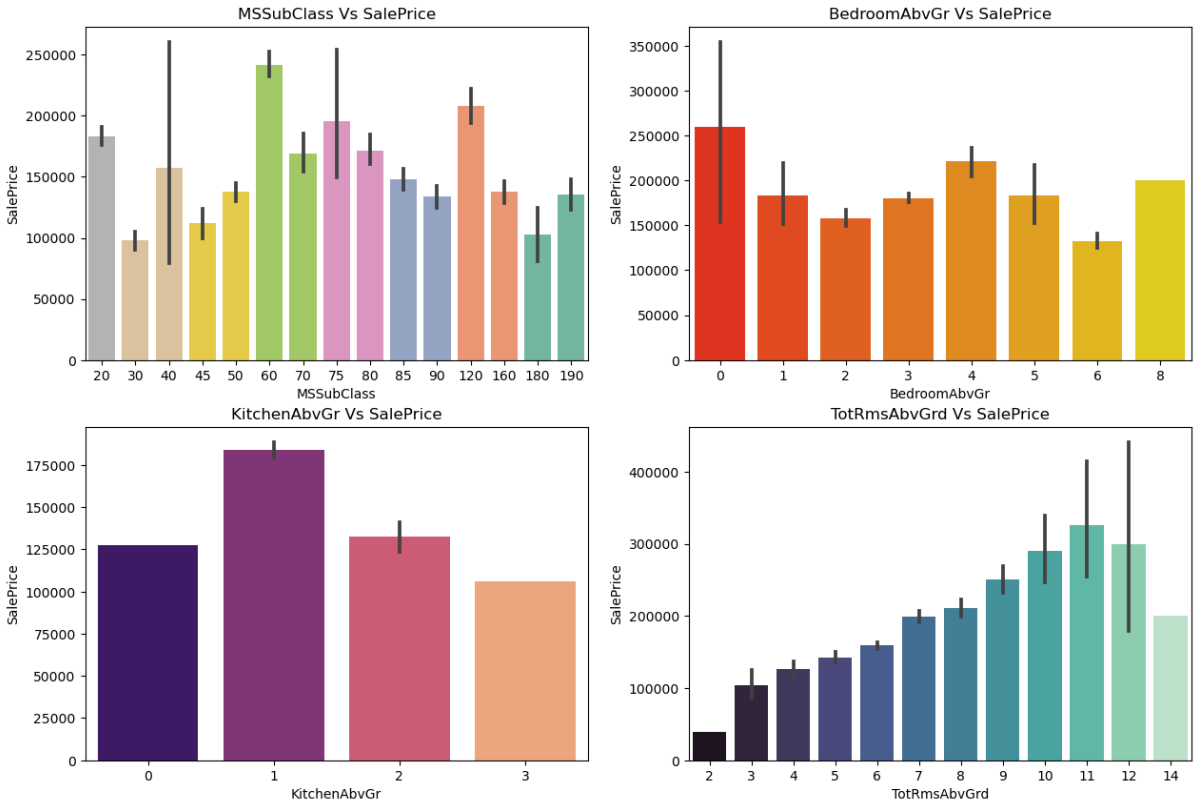
```
<AxesSubplot:xlabel='BsmtExposure', ylabel='count'>
```



Continuous variables Vs SalePrice

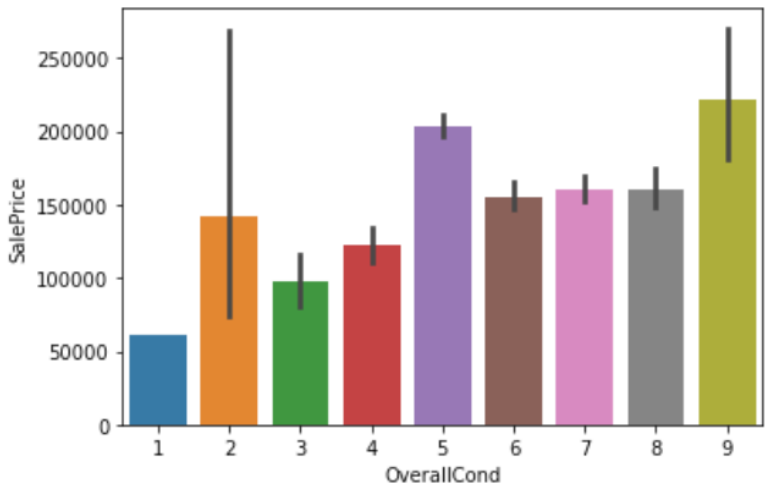


Discrete variables Vs SalePrice



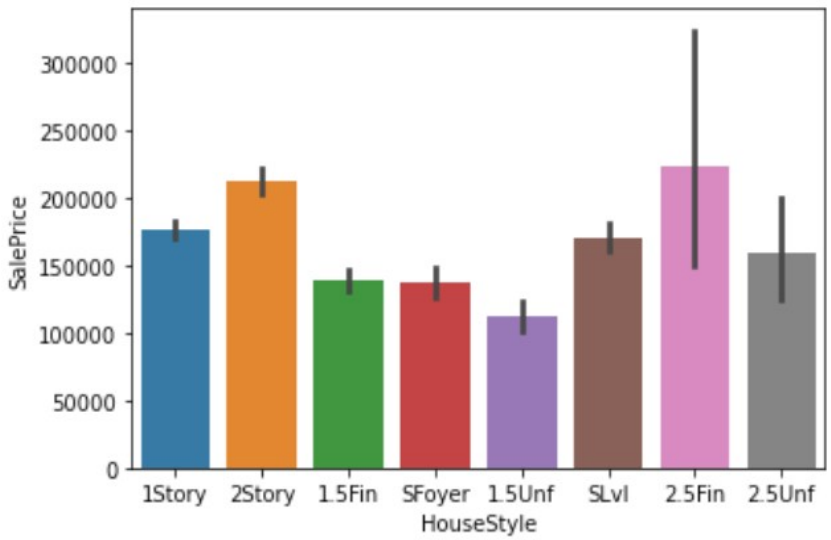

```
: sns.barplot(x='OverallCond',y='SalePrice',data=df)
```

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



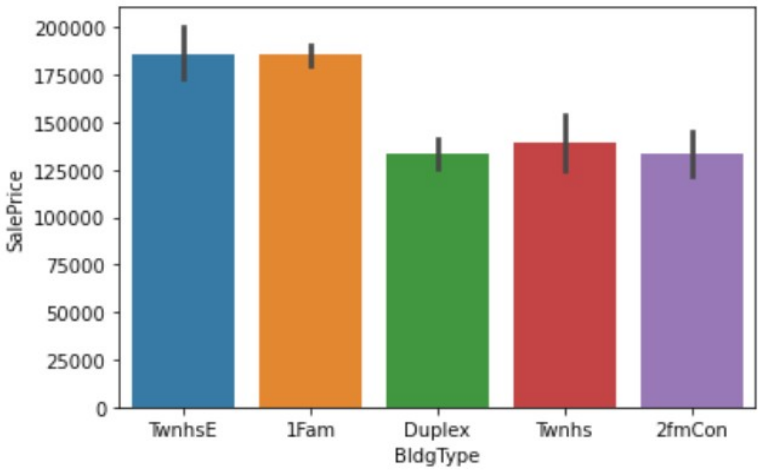
```
sns.barplot(x='HouseStyle',y='SalePrice',data=df)
```

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




```
sns.barplot(x='BldgType',y='SalePrice',data=df)
```

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



Observation:

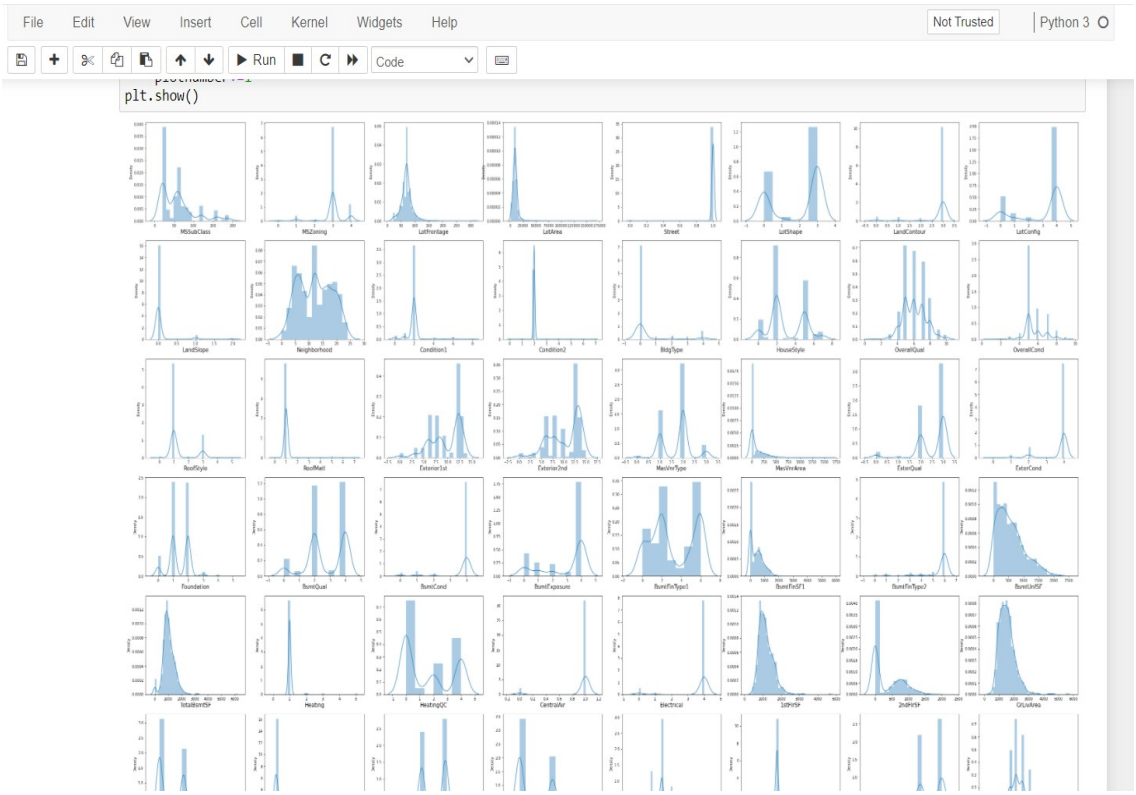
- ✓ **SalePrice vs MSZoning :** Most of the houses are belongs to Floating Village Residential followed by Residential Low Density. The houses from this zone are have high sale price compared to other zones.
- ✓ **SalePrice vs Street:** By observing the bar plot, it is obvious that the property of house with Paved type of road have high SalePrice and the the houses in gravel roads have very less sale price.
- ✓ **SalePrice vs LotShape:** Most of the houses having moderately irregular and irregular shape of property have high sale price and houses with regullar type of property have less sale peice compared to others.
- ✓ **SalePrice vs LandContour:** The houses having the hillside and depression property flatness have high sale price compared to others.
- ✓ **SalePrice vs LotConfig:** Most of the houses with Frontage on 3 sides of property have high sale price compared to others.
- ✓ **SalePrice vs LandSlope:** There is no significance difference between the slope of the property. As we can observe the houses having Gentle slope, Moderate Slope and Severe Slope have same sale price.
- ✓ **SalePrice vs Neighborhood:** The houses which are located near Northridge have high sale price compared to others.
- ✓ **SalePrice vs Condition1:** The houses having the conditions adjacent to postive off-site feature and houses within 200' of North-South Railroad have high sale price compared to others.
- ✓ **SalePrice vs Condition2:** The houses having the conditions near positive off-site feature park, greenbelt, etc and adjacent to postive off-site feature have high sale price.
- ✓ **SalePrice vs BldgType:** Most of the houses are Single-family Detached and Townhouse End Unit and they have higher sale price compared to other categories.
- ✓ **SalePrice vs HouseStyle:** Houses which are having style of dwelling 2nd level finished and Two story have high sale price compared to other types.

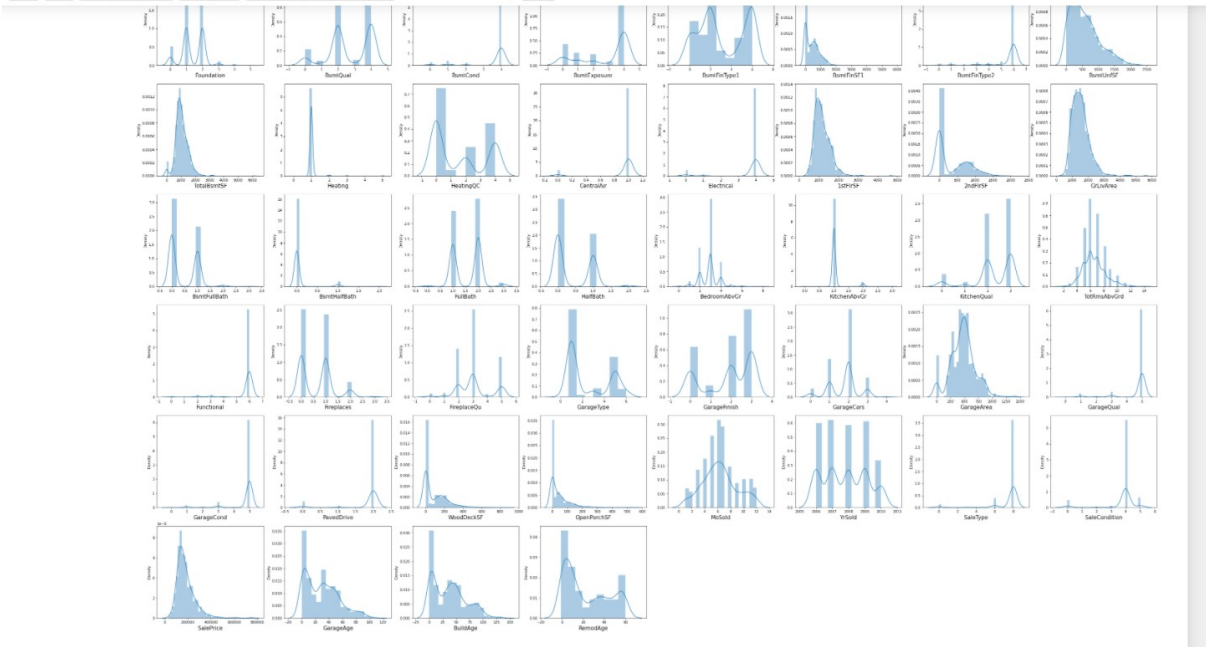
- ✓ **SalePrice vs RoofStyle:** The houses having the roof style Flat, Hip and Shed have high sale price and the houses having gabrel roof style have less sale price.

- ✓ **SalePrice vs RoofMatl**: Houses with Wood Shingles root materials have high sale prices.
- ✓ **SalePrice vs Exterior1st:** Houses having Imitation Stucco, Stone and Cement Board as 1st exterior cover have high sale price.

- ✓ **SalePrice vs Exterior2nd:** Houses having Imitation Stucco and other as 2nd cover have high sale price.

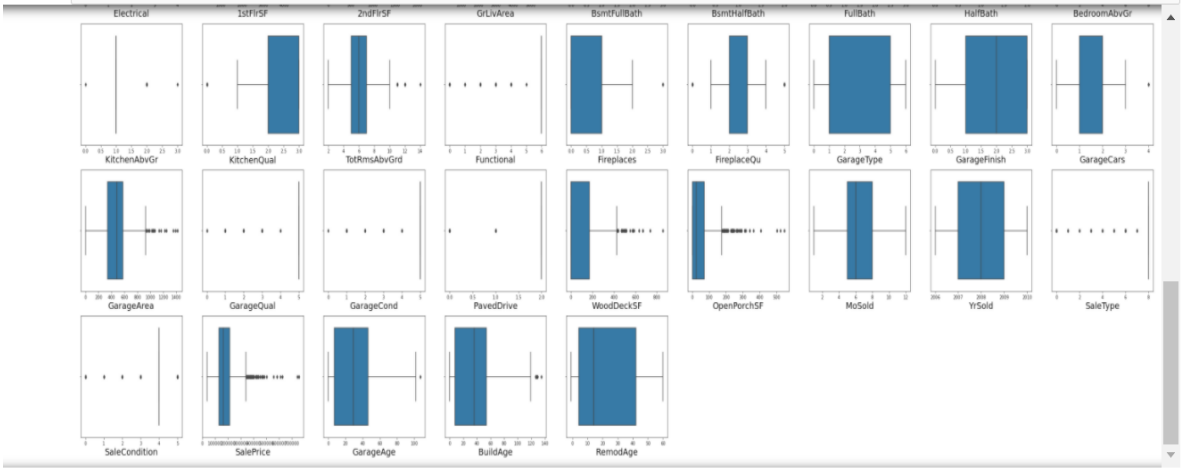
Skewness:

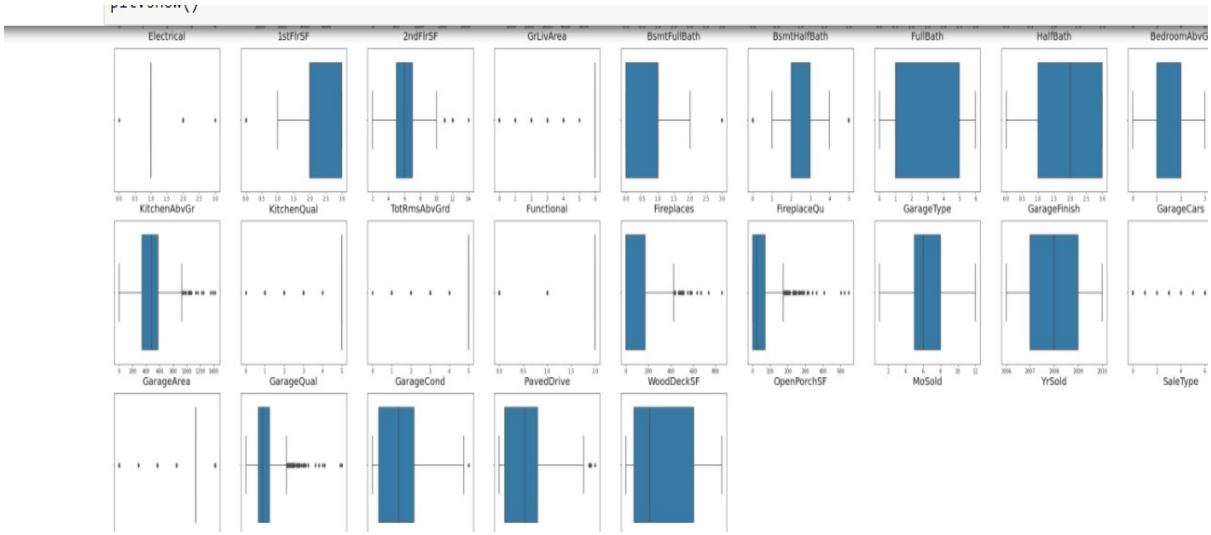




Outlier:

```
for column in df_train:
    if plotnumber<=81:
        ax=plt.subplot(9,9,plotnumber)
        sns.boxplot(df_train[column])
        plt.xlabel(column,fontsize=20)
    plotnumber+=1
plt.show()
```





- Interpretation of the Results

Interpretation of the results Visualizations:
Visualisation:

I have used distribution plot to visualize the target variable SalePrice, which was almost normally distributed. From the scatter plot we noticed most of the features like OverallQual, TotalRmsAbvGrd, FullBath, GarageCars etc had some strong linear relation with target as we observed as the quality or area increased, the sale price also tends to increase. The heat map helped to understand the correlation between target and features. Also, with the help of heat map I found multicollinearity problem and I have done feature selection to overcome with the issue. Detected outliers and skewness using box plots and distribution plots. I got to know the count of each column using count plots.

Pre-processing:

The dataset should be cleaned and scaled to build the ML models to get good predictions. I have performed many processing steps which I have already mentioned in the pre-processing step.

Modelling:

After cleaning and processing both train and test data, I performed train test split to build the model. I have built multiple regression models to get the accurate R2 score, and evaluation metrics. I got Random Forest regressor as best model which gives 90% R2 score. I checked the cross-validation score also.
After tuning the best model Random Forest regressor I got 92% R2 score and the value of MAE, MSE and RMSE values. And finally, I saved my final model and got the good predictions results for test dataset.

CONCLUSION

- **Key Findings and Conclusions of the Study**

In this study, we have used multiple machine learning models to predict the house sale price. We have gone through the data analysis by performing feature engineering, finding the relation between features and label through visualizations. And got the important feature. We have got good prediction results. After using hyper parameter tuning, the best model's R2score was 92% also the errors decreased which means no over-fitting issue.

Which variables are important to predict the price of variable?

Overall Quality is the most contributing and highest positive impactingfeature for prediction. Also, the features like GarageArea, LotArea, 1stFlrSF, TotalBsmtSF etc have somewhat linear relation with the price variable.
The houses which have very excellent overall quality like material and finish of the house have high sale price. Also we have observed from the plot that as the overall quality of the house increases, the sale price also increases.

That is there is good linear relation between SalePrice and OverallQual. So, if the seller builds the house according to these types of qualities that will increase the sale price of the house. There is a linear relation between the SalePrice and 1stFlrSF. As we have seen as the 1st floor area increases, sales price also increases moderately. So, people like to live in the houses where there are only 1-2 floors and the cost of the house also increases in this case.

Also, we have seen the positive linear relation between the SalePrice and GarageArea. As size of garage area increases, sale price also increases. There is positive linear relation between sale price and TotalBsmtSF. As total basement area increases, sale price also increases.

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

  While working on this project I learned many things about the real-estate market and how the machine learning models helped to predict the price of houses which indeed helps the sellers and buyers to understand the future price of the house. Here I found that the dataset contains several types of data. I used several types of plotting to visualize the relation between target and features. This graphical representation helped me to understand which features are important and how these features describe the sale price.

  Data cleaning was one of the important and crucial things in this project where I replaced all the null values with imputation methods and dealt with features having zero values. Finally, our aim is achieved by predicting the house price for the test data, I hope this will be further helps for sellers and buyers to understand the house marketing. The machine learning models and data analytic techniques will have an important role to play in this type of problems. It helps the customers to know the future price of the houses

- Limitations of this work and Scope for Future Work

Limitation:

The dataset contains some irrelevant columns, zero values, null values, so it is needed to increase the dataset size by filling these values.

The dataset has many limitations, the main limitation is that we have no information potential buyers and environment of the sale. The factors such as auctions can have an influence on the price of the house.

The dataset does not capture many economic factors. Collecting more accurate and important details about the houses from the buyers will help to analyse the data more clearly.

Scope of Future Works:

One of the major future scopes is adding estate database for more cities which will provide the user to explore more estates and reach an accurate decision.

As a recommendation, I advise to use this model by the people who want to buy a house in the area covered by the dataset to have an idea about the actual price.

The model can be used also with datasets that cover different cities and areas provided that they contain the same features. I also suggest that people take into consideration the features that were deemed as most important as seen in this study might help them estimate the house price better