Project Report : Housing Price

By: Chirag Patil

FLIPROBO SME: Khushboo Garg

ACKNOWLEDGMENT

I would like to express my special gratitude to "Flip Robo" team, who have provided me with the opportunity to Explore the dataset which in turn helped me tune my analyzation skills. And I want to express my huge gratitude to Ms. Khushboo Garg (SME Flip Robo),who is a great help in any Difficulties,

A huge thanks to "Data trained" who are the reason behind my Internship at Fliprobo.

References use in this project:

1. SCIKIT Learn Library Documentation

2. Blogs from towardsdatascience, Analytics Vidya, Medium

3. Andrew Ng Notes on Machine Learning (GitHub)

4. Data Science Projects with Python Second Edition by Packt

5. Hands on Machine learning with scikit learn and tensor flow by Aurelien Geron

Introduction

1.1 Business Problem Framing

Real Estate Property is not only the basic need of a man but today it also represents the riches and prestige of a person. Investment in real estate generally seems to be profitable because their property values do not decline rapidly. The market demand for housing is always increasing every year due to increase in population and migrating to other cities for their financial purpose. Changes in the real estate price can affect various household investors, bankers, policy makers and many. Investment in Housing seems to be an attractive choice for the investments.

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.

There are many factors that have an impact on house prices, such as the number of bedrooms and bathrooms. House price depends upon its location as well. A house with great accessibility to highways, schools, malls, employment opportunities, would have a greater price as compared to a house with no such accessibility..

Regression is a supervised learning algorithm in machine learning which is used for prediction by learning and forming a relationship between present statistical data and target value i.e., Sale Price in this case. Different factors are taken into consideration while predicting the worth of the house like location, neighbourhood and various amenities like garage space etc. if learning is applied to above parameters with target values for a certain geographical region as different areas differ in price like land price, housing style, material used, availability of public utilities.

1.2 Conceptual Background of the Domain Problem

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 data is provided in the CSV file below. 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?

It is 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.

1.3 Review of Literature

Related Work or Literature survey is the most important step in any kind of research. Before start developing Machine Learning Model, we need to study the previous papers of our domain which we are working and on the basis of study we can predict or generate the drawback and start working with the reference of previous papers. In this section, we briefly review the related work on house price prediction and the techniques used. Predicting house prices manually is a difficult task and generally not very accurate, hence there are many systems developed for house price prediction.

Motivation

The project was the first provided to me by Flip Robo Technologies as a part of the internship programme. The exposure to real world data and the opportunity to deploy my skillset in solving a real time problem has been the primary motivation.

Our main objective of doing this project is to build a model to predict the house prices with the help of other supporting features. In order to improve the selection of customers, the client wants some predictions that could help them in further investment and improvement in selection of customers.

The purpose of this study is to deepen the knowledge in regression methods in machine learning. In addition, the given datasets should be processed to enhance performance, which is accomplished by identifying the necessary features by applying one of the selection methods to eliminate the unwanted variables since each house has its unique features that help to estimate its price. These features may or may not be shared with all houses, which means they do not have the same influence on the house pricing resulting in inaccurate output.

The study answers the following research questions:

- Research question 1: Which machine learning algorithm performs better and has the most accurate result in house price prediction? And why?

- Research question 2: What are the factors that have affected house prices in Australia over the years?

Analytical Problem Framing

1. Data set provided by Flip Robo was in the format of CSV (Comma Separated Values). There are 2 data sets that are given. One is training data and one is testing data. 1) Train file will be used for training the model, i.e., the model will learn from this file. It contains all the independent variables and the target variable. The dimension of data is 1168 rows and 81 columns. 2) Test file contains all the independent variables, but not the target variable. We will apply the model to predict the target variable for the test data. The dimension of data is 292 rows and 80 columns.

   The data types of different features are as shown below:

```
In [57]:
# As we have too many columns let sort columns by their datatype
df.columns.to_series().groupby(df.dtypes).groups

Out[57]: {int64: ['Id', 'MSSubClass', 'LotArea', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd', 'B
smtFinSF1', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF', 'GrLivA
rea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath', 'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'TotRms
AbvGrd', 'Fireplaces', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'OpenPorchSF', 'EnclosedPorch', '3Ss
nPorch', 'ScreenPorch', 'PoolArea', 'MiscVal', 'MoSold', 'YrSold', 'SalePrice'], float64: ['LotFronta
ge', 'MasVnrArea', 'GarageYrBlt'], object: ['MSZoning', 'Street', 'Alley', 'LotShape', 'LandContour',
'Utilities', 'LotConfig', 'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType', 'House
Style', 'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType', 'ExterQual', 'ExterCon
d', 'Foundation', 'BsmtQual', 'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinType2', 'Heating',
'HeatingQC', 'CentralAir', 'Electrical', 'KitchenQual', 'Functional', 'FireplaceQu', 'GarageType', 'G
arageFinish', 'GarageQual', 'GarageCond', 'PavedDrive', 'PoolQC', 'Fence', 'MiscFeature', 'SaleType',
'SaleCondition']}
```

   3.Data Pre-processing The dataset is large and it may contain some data error. In order to reach clean, error free data some data cleaning & data pre-processing performed data.

   • Data Integrity check – No duplicate entries present in dataset.

Data Integrity Check

```
In [59]:
#For train Data
df.duplicated().sum()

Out[59]: 0

In [60]:
#for test Data
df_test.duplicated().sum()

Out[60]: 0
```

- There are no duplicated Values

Check for presense of any whitespaces, '?', 'NA', '-' in dataset

```
In [61]:
#For train Data
df.isin(['NA','N/A','-',' ','?',' ?']).sum().any()

Out[61]: False

In [62]:
#for test Data
df_test.isin(['NA','N/A','-',' ','?',' ?']).sum().any()

Out[62]: False
```

Some features contain missing values as shown below:

In [66]:
```
# Finding what percentage of data is missing from dataset
pd.set_option('display.max_rows',None)
missing_values = df.isnull().sum().sort_values(ascending = False)
percentage_missing_values = (missing_values/len(df))*100
print(pd.concat([missing_values, percentage_missing_values], axis = 1, keys =['Missing Values', '% Missing data']))
```

	Missing Values	% Missing data
PoolQC	1161	99.400685
MiscFeature	1124	96.232877
Alley	1091	93.407534
Fence	931	79.708904
FireplaceQu	551	47.174658
LotFrontage	214	18.321918
GarageCond	64	5.479452
GarageQual	64	5.479452
GarageType	64	5.479452
GarageFinish	64	5.479452
GarageYrBlt	64	5.479452
BsmtExposure	31	2.654110
BsmtFinType2	31	2.654110
BsmtQual	30	2.568493
BsmtCond	30	2.568493
BsmtFinType1	30	2.568493
MasVnrType	7	0.599315
MasVnrArea	7	0.599315
MSSubClass	0	0.000000
Fireplaces	0	0.000000
TotRmsAbvGrd	0	0.000000
KitchenQual	0	0.000000
KitchenAbvGr	0	0.000000
BedroomAbvGr	0	0.000000
HalfBath	0	0.000000
FullBath	0	0.000000
BsmtHalfBath	0	0.000000

Removing the features which contain high amount missing values e.g., Top 5 features with missing value in above list. These Features contain missing values higher than 50 -60 % .Rest of feature are handle based on mean, median or mode imputation depending on outliers & distribution of feature.

Missing Value Imputation in Test Dataset

In [80]:
```
# Removing columns with high missing value percentage
df_test= df_test.drop(['PoolQC','MiscFeature','Alley','Fence','FireplaceQu'],axis=1)
```

In [81]:
```
df_test['LotFrontage'] = df_test['LotFrontage'].fillna(df_test['LotFrontage'].median())
df_test['GarageType'] = df_test['GarageType'].fillna(df_test['GarageType'].mode()[0])
df_test['GarageYrBlt']= df_test['GarageYrBlt'].fillna(df_test['GarageYrBlt'].mode()[0])
df_test['GarageFinish']= df_test['GarageFinish'].fillna(df_test['GarageFinish'].mode()[0])
df_test['GarageQual'] = df_test['GarageQual'].fillna(df_test['GarageQual'].mode()[0])
df_test['GarageCond'] = df_test['GarageCond'].fillna(df_test['GarageCond'].mode()[0])
df_test['BsmtFinType2'] = df_test['BsmtFinType2'].fillna(df_test['BsmtFinType2'].mode()[0])
df_test['BsmtExposure'] = df_test['BsmtExposure'].fillna(df_test['BsmtExposure'].mode()[0])
df_test['BsmtFinType1'] = df_test['BsmtFinType1'].fillna(df_test['BsmtFinType1'].mode()[0])
df_test['BsmtCond'] = df_test['BsmtCond'].fillna(df_test['BsmtCond'].mode()[0])
df_test['BsmtQual'] = df_test['BsmtQual'].fillna(df_test['BsmtQual'].mode()[0])
df_test['MasVnrType'] = df_test['MasVnrType'].fillna(df_test['MasVnrType'].mode()[0])
df_test['MasVnrArea'] = df_test['MasVnrArea'].fillna(df_test['MasVnrArea'].median())
df_test['Electrical'] = df_test['Electrical'].fillna(df_test['Electrical'].mode()[0])
```

▪ Some of the features are remove as they are representing in another similar feature.

In [112]:
```
# Droping unnecessary columns
df.drop('Id','Utilities',axis=1,inplace=True)
df_test.drop('Id','Utilities',axis=1,inplace=True)
```

TotalBsmtSF is sum of above remaining features. We will drop other three feature

In [113]:
```
df.drop(['BsmtFinSF1','BsmtFinSF2','BsmtUnfSF'],axis=1,inplace=True)
df_test.drop(['BsmtFinSF1','BsmtFinSF2','BsmtUnfSF'],axis=1,inplace=True)
```

GrLivArea is sum of above remaining features. We will drop other three features

In [116]:
```
df.drop(['1stFlrSF','2ndFlrSF','LowQualFinSF'],axis=1,inplace=True)
df_test.drop(['1stFlrSF','2ndFlrSF','LowQualFinSF'],axis=1,inplace=True)
```

In [117]:
```
df.drop(['EnclosedPorch','3SsnPorch','ScreenPorch','PoolArea','MiscVal'],axis=1,inplace=True)
df_test.drop(['EnclosedPorch','3SsnPorch','ScreenPorch','PoolArea','MiscVal'],axis=1,inplace=True)
```

▪ Label Encoding of Categorical features: The categorical Variable in training & testing dataset are converted into numerical datatype using label encoder from scikit library.

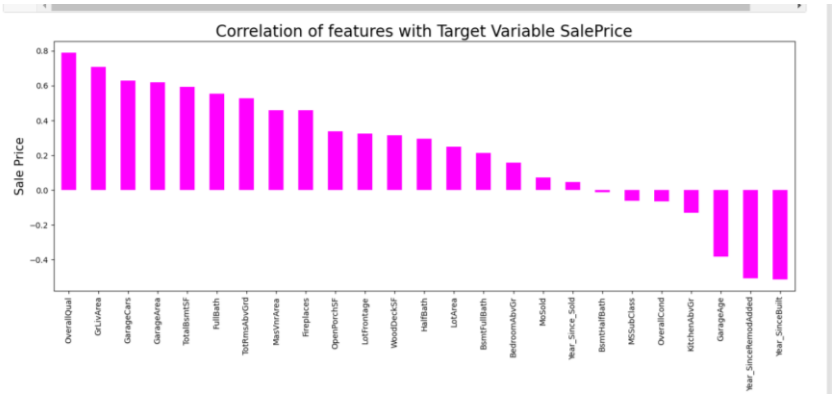
In [122]:
```
# Using Label encoder for transforming Categorical data
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for i in Categorical_features:
    df[i] = le.fit_transform(df[i])
df.head()
```

Out[122]:

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	LotShape	LandContour	LotConfig	LandSlope	Neighborhood	Condi
0	120	3	70.0	4928	1	0	3	4	0	13	
1	20	3	95.0	15865	1	0	3	4	1	12	
2	60	3	92.0	9920	1	0	3	1	0	15	
3	20	3	105.0	11751	1	0	3	4	0	14	
4	20	3	70.0	16635	1	0	3	2	0	14	

In [123]:
```
#For testing Data
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for i in Categorical_features:
    df_test[i] = le.fit_transform(df_test[i])
df_test.head()
```

4.Data Inputs- Logic- Output Relationships Correlation heatmap is plotted to gain understanding of relationship between target features & independent features. We can see that lot of features are highly correlated with target variable Sale Price. To gain insights about relationship between Input & output different types of visualization are plotted which we will see in EDA section of this report.



Libraries Used – General libraries used for data wrangling:

Importing Necessary Libraries

In [53]:
```
import pandas as pd
pd.set_option('display.max_columns',None) # this will enable us to see truncated columns
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

Libraries used for machine learning model building:

Model Training and Selection

In [130]:
```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import ExtraTreesRegressor
```

Models Development & Evaluation

1. IDENTIFICATION OF POSSIBLE PROBLEMSOLVING APPROACHES (METHODS):

Our objective is to predict house price and analyse feature impacting Sale price. This problem can be solve using regression-based machine learning algorithm like linear regression. For that purpose, first task is to convert categorical variable into numerical features. Once data encoding is done then data is scaled using standard scalar. Final model is built over this scaled data. For building ML model before implementing regression algorithm, data is split in training & test data using train_test_split from model_selection module of sklearn library

2. Testing of Identified Approaches (Algorithms):

The different regression algorithm used in this project to build ML model are as below:

- ❖ Linear Regression
- ❖ Random Forest Regressor
- ❖ Decision Tree Regressor
- ❖ Extra Tree Regressor

3. KEY METRICS FOR SUCCESS IN SOLVING PROBLEM UNDER CONSIDERATION:

Following metrics used for evaluation:

1. Mean absolute error which gives magnitude of difference between the prediction of an observation and the true value of that observation.
2. Root mean square error is one of the most commonly used measures for evaluating the quality of predictions.
3. R2 score which tells us how accurate our model predict result, is going to important evaluation criteria along with Cross validation score.
4. Cross Validation Score

4.RUN AND EVALUATE SELECTED MODELS

1. Linear Regression:

Linear Regression

In [134]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, random_state= 266, test_size=0.20)
lin_reg= LinearRegression()
lin_reg.fit(X_train, y_train)
y_pred = lin_reg.predict(X_test)
print('\033[1m'+ 'Error :'+ '\033[0m')
print('Mean absolute error :', mean_absolute_error(y_test,y_pred))
print('Mean squared error :', mean_squared_error(y_test, y_pred))
print('Root Mean squared error :', np.sqrt(mean_squared_error(y_test, y_pred)))
print('\033[1m'+' R2 Score :'+'\033[0m')
print(r2_score(y_test,y_pred)*100)
```

```
Error :
Mean absolute error : 16755.119080357068
Mean squared error : 468632928.0569503
Root Mean squared error : 21647.931265064344
 R2 Score :
89.69343733155823
```

Cross Validation of Linear Regression

In [136]:
```
from sklearn.model_selection import cross_val_score
score = cross_val_score(lin_reg, X, y, cv=5)
print('\033[1m'+'Cross Validation Score :',lin_reg,":"+'\033[0m\n')
print("Mean CV Score :",score.mean())
print('Difference in R2 & CV Score:',(r2_score(y_test,y_pred)*100)-(score.mean()*100))
```

```
Cross Validation Score : LinearRegression() :

Mean CV Score : 0.7662072724233103
Difference in R2 & CV Score: 13.0727100892272
```

2. Decision Tree Regressor:

Decision Tree Regressor

In [140]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, random_state= 266, test_size=0.20)
dtc = DecisionTreeRegressor()
dtc.fit(X_train, y_train)
y_pred = dtc.predict(X_test)
print('\033[1m'+ 'Error of Decision Tree Regressor:'+ '\033[0m')
print('Mean absolute error :', mean_absolute_error(y_test,y_pred))
print('Mean squared error :', mean_squared_error(y_test, y_pred))
print('Root Mean squared error :', np.sqrt(mean_squared_error(y_test, y_pred)))
print('\033[1m'+'R2 Score of Decision Tree Regressor :'+'\033[0m')
print(r2_score(y_test,y_pred)*100)
```

```
Error of Decision Tree Regressor:
Mean absolute error : 30015.23076923077
Mean squared error : 2151291356.786325
Root Mean squared error : 46382.01544549703
R2 Score of Decision Tree Regressor :
71.74674786725194
```

Cross Vaidation

In [141]:
```
from sklearn.model_selection import cross_val_score
score = cross_val_score(dtc, X, y, cv=5)
print('\033[1m'+'Cross Validation Score :',dtc,":"+'\033[0m\n')
print("Mean CV Score :",score.mean())
print('Difference in R2 & CV Score:',(r2_score(y_test,y_pred)*100)-(score.mean()*100))
```

```
Cross Validation Score : DecisionTreeRegressor() :

Mean CV Score : 0.6814966900182493
Difference in R2 & CV Score: 3.597078865427008
```

3.RandomForest Regressor:

Random Forest Regressor

In [143]:
```
X_train, X_test, y_train,y_test = train_test_split(X, y, random_state= 266, test_size=0.20)
rfc = RandomForestRegressor()
rfc.fit(X_train, y_train)
y_pred = rfc.predict(X_test)
print('\033[1m'+ 'Error of Random Forest Regressor:'+ '\033[0m')
print('Mean absolute error :', mean_absolute_error(y_test,y_pred))
print('Mean squared error :', mean_squared_error(y_test, y_pred))
print('Root Mean squared error :', np.sqrt(mean_squared_error(y_test, y_pred)))
print('\033[1m'+'R2 Score of Random Forest Regressor :'+'\033[0m')
print(r2_score(y_test,y_pred)*100)
```

```
Error of Random Forest Regressor:
Mean absolute error : 17725.927487046632
Mean squared error : 669356405.9270023
Root Mean squared error : 25871.92311999636
R2 Score of Random Forest Regressor :
90.3829722756163
```

Cross Validation of Random Forest Regressor

In [144]:
```
from sklearn.model_selection import cross_val_score
score = cross_val_score(rfc,X,y, cv=5)
print('\033[1m'+'Cross Validation Score :',rfc,":"+'\033[0m\n')
print("Mean CV Score :",score.mean())
print('Difference in R2 & CV Score:',(r2_score(y_test,y_pred)*100)-(score.mean()*100))
```

```
Cross Validation Score : RandomForestRegressor() :

Mean CV Score : 0.8312567397795227
Difference in R2 & CV Score: 7.257298297664022
```

4. Extra Trees Regressor:

Extra Tree Regressor

In [146]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state= 266, test_size=0.20)
etc = ExtraTreesRegressor()
etc.fit(X_train, y_train)
y_pred = etc.predict(X_test)
print('\033[1m'+ 'Error of Extra Tree Regressor:'+ '\033[0m')
print('Mean absolute error :', mean_absolute_error(y_test,y_pred))
print('Mean squared error :', mean_squared_error(y_test, y_pred))
print('Root Mean squared error :', np.sqrt(mean_squared_error(y_test, y_pred)))
print('\033[1m'+'R2 Score of Extra Tree Regressor :'+'\033[0m')
print(r2_score(y_test,y_pred)*100)
```

```
Error of Extra Tree Regressor:
Mean absolute error : 14411.390042735044
Mean squared error : 378291300.9373927
Root Mean squared error : 19449.712104229016
R2 Score of Extra Tree Regressor :
91.68030506050187
```

cross Validation of Exta Tree Regressor

In [147]:
```
from sklearn.model_selection import cross_val_score
score = cross_val_score(etc, X,y, cv=5)
print('\033[1m'+'Cross Validation Score :',etc,":"+'\033[0m\n')
print("Mean CV Score :",score.mean())
print('Difference in R2 & CV Score:',(r2_score(y_test,y_pred)*100)-(score.mean()*100))
```

```
Cross Validation Score : ExtraTreesRegressor() :

Mean CV Score : 0.8315782868043463
Difference in R2 & CV Score: 8.522476380067232
```

5-Fold cross validation performed over all models. We can see that Random Forest Regressor gives maximum R2 score of 90.38 and with cross validation score of 83.10 %. Among all model we will select Random Forest Regressor as final model and we will perform hyper parameter tuning over this model to enhance its R2 Score.

Hyper Parameter Tuning : GridSearchCV

In [149]:
```
from sklearn.model_selection import GridSearchCV
```

In [150]:
```
print(rfc.get_params())
```

```
{'bootstrap': True, 'ccp_alpha': 0.0, 'criterion': 'squared_error', 'max_depth': None, 'max_features': 1.0, 'max_leaf_nodes': N
one, 'max_samples': None, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 1, 'min_samples_split': 2, 'min_weight_fraction_lea
f': 0.0, 'n_estimators': 100, 'n_jobs': None, 'oob_score': False, 'random_state': None, 'verbose': 0, 'warm_start': False}
```

In [151]:
```
parameter = {
 'bootstrap': [True, False],
 'max_features': ['auto'],
 'min_samples_leaf': [1, 2, 4 ],
 'n_estimators': [100, 500,1000,1500,2000]}
```

In [154]:
```
GCV = GridSearchCV(RandomForestRegressor(),parameter,verbose =10)
GCV.fit(X_train,y_train)
```

Saving of the Final Model :

Final Model

In [155]:
```
Final_mod=RandomForestRegressor(max_depth = None, bootstrap = True, max_features='sqrt', min_samples_leaf = 1,
                                n_estimators = 100 )
Final_mod.fit(X_train,y_train)
pred=Final_mod.predict(X_test)
print('R2_Score:',r2_score(y_test,pred)*100)
print('mean_squared_error:',mean_squared_error(y_test,pred))
print('mean_absolute_error:',mean_absolute_error(y_test,pred))
print("RMSE value:",np.sqrt(mean_squared_error(y_test, pred)))
```

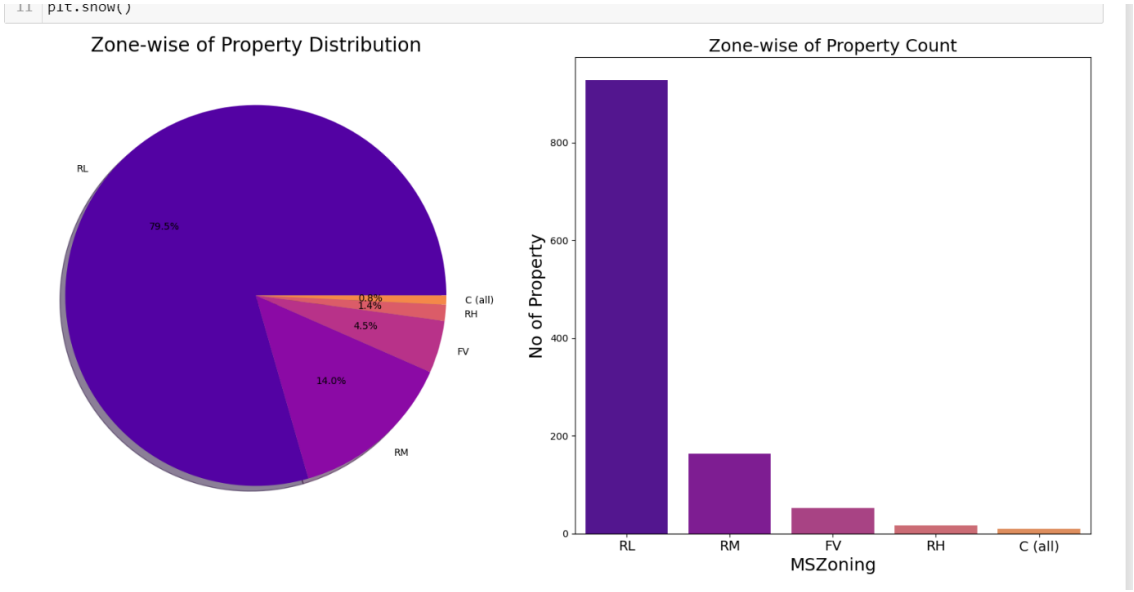
```
R2_Score: 90.74498594610705
mean_squared_error: 420819673.32953846
mean_absolute_error: 14692.97452991453
RMSE value: 20513.889765949763
```

Saving the Model

In [156]:
```
# Saving the model using .pkl
import joblib
joblib.dump(Final_mod,"Surprise_Housing_Price_Prediction.pkl")
```

Out[156]:
```
['Surprise_Housing_Price_Prediction.pkl']
```

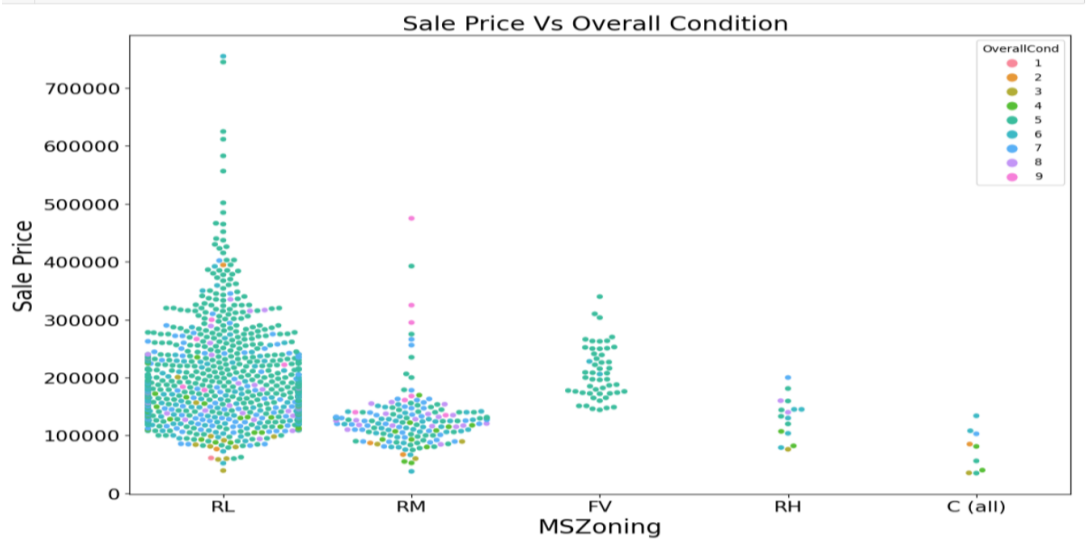
VISUALIZATIONS:



Obsesrvation:

▪ 79.5% of House properties belongs to Low Density Residential Area followed by 14 % of properties belong to Medium Density Residential Area.

▪ Very Few properties (0.8%) belong to Commercial zone.

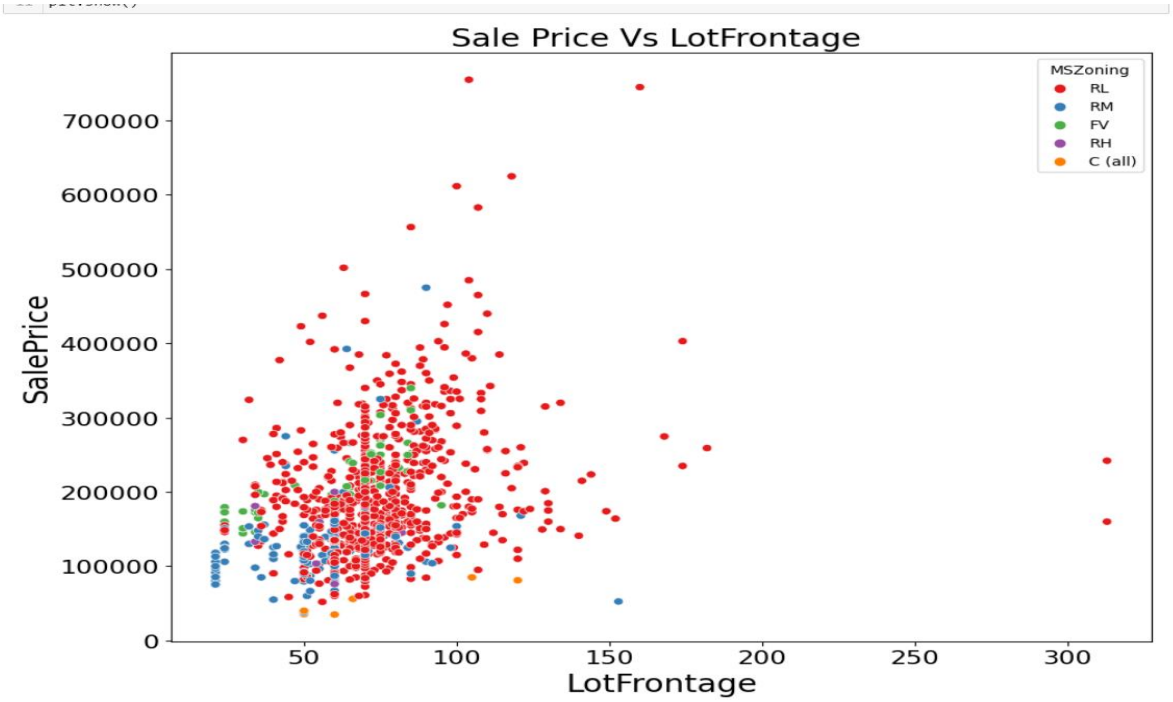


Observation:

• Most of property for sale have overall condition rating of either 5 or 6.

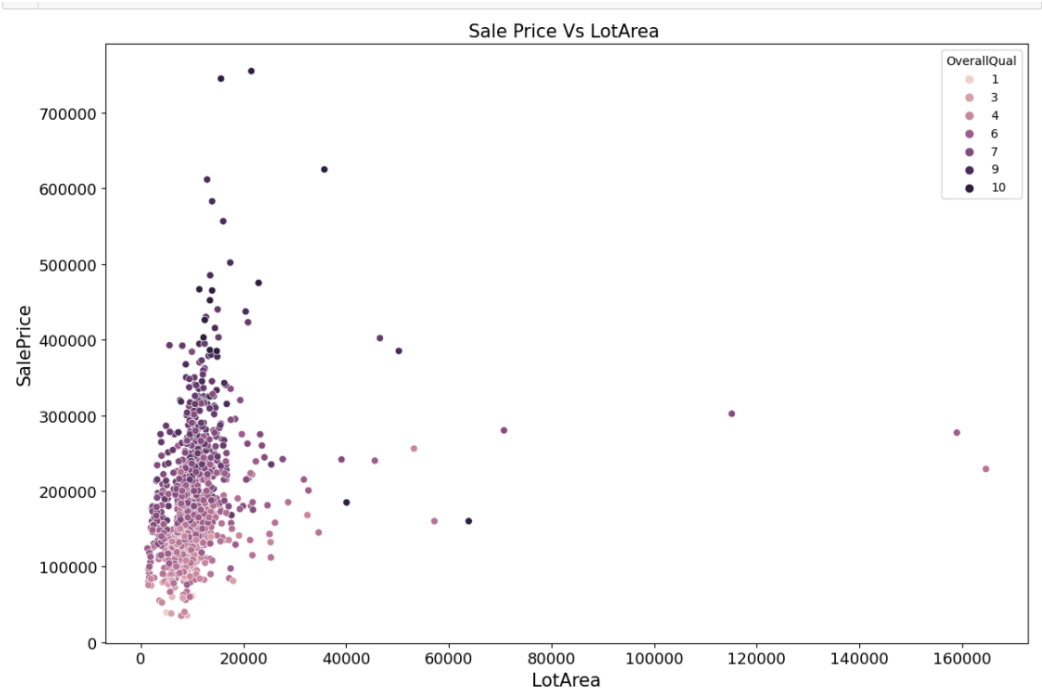
• 80% of housing data belongs to Low density Residential Area and Now in Swarm Plot that Sale Price inside RL Zone is much higher than another remaining zone.

• Cheapest properties are available in Commercial zone



Obseravtion:

With Exception of Commercial zone, As Lot Frontage area increase (which indicate Size of street connected to property) the Sale Price increases.

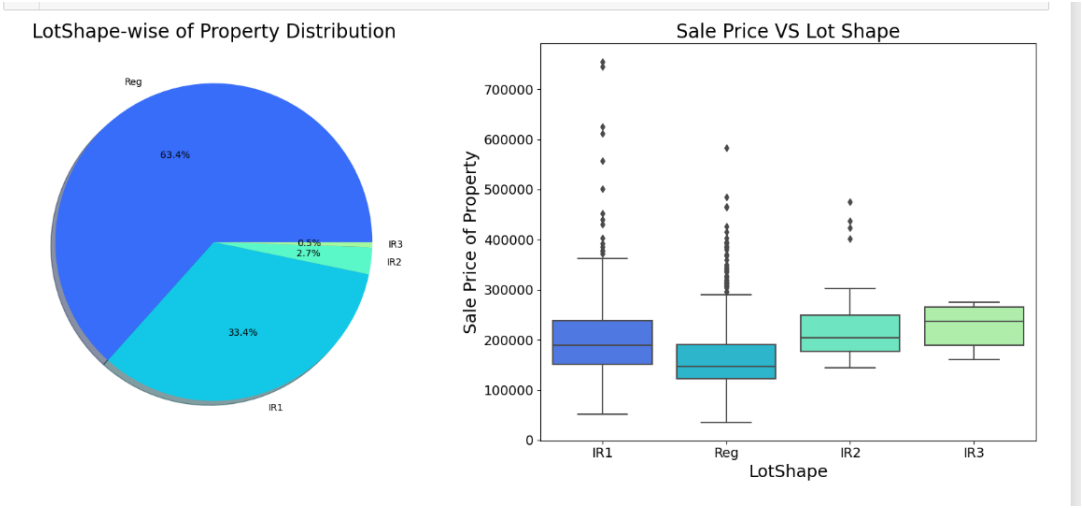


Observation:

▪ There is seems to be no Significant relationship found between Sale price & Lot area.

Observation:

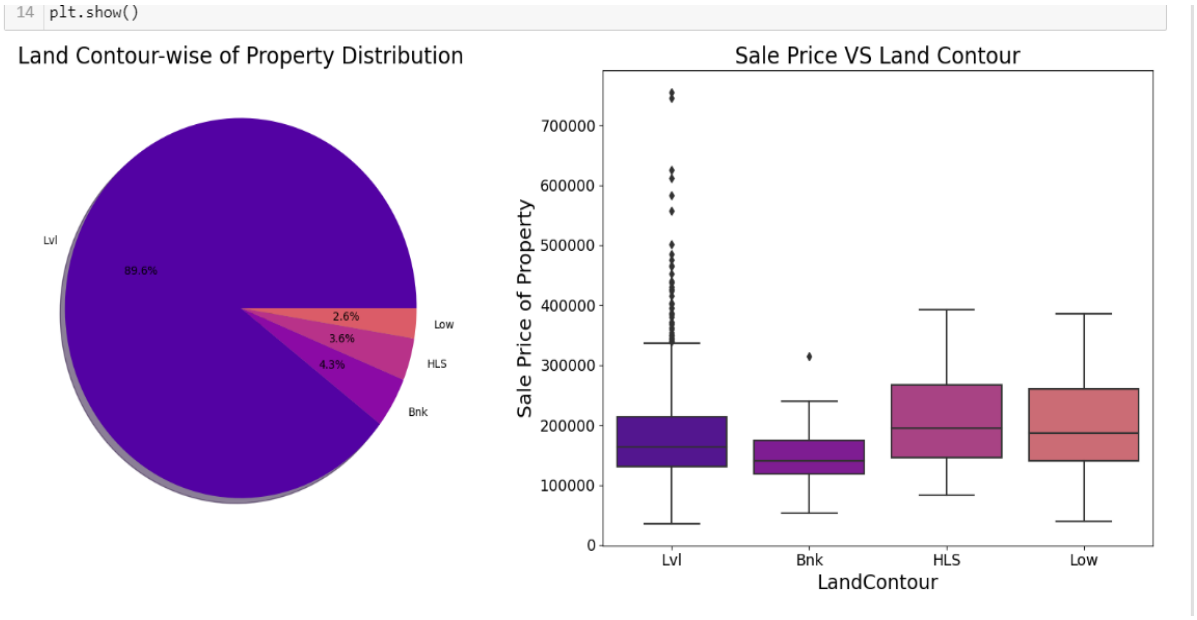
• In terms of Average Sale price house properties belonging to Floating Village Residential Zone are costlier than rest.



Observation :

▪ 63.4% house properties are regular in shape.

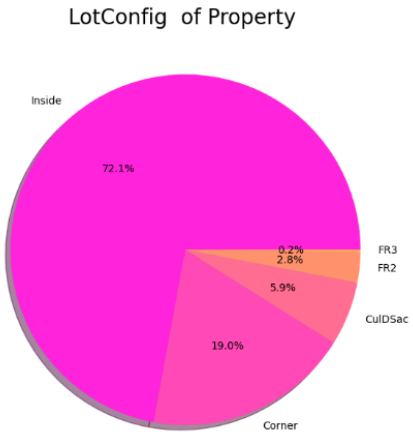
▪ Sale Price of property with slight irregular shape is higher than regular shape.

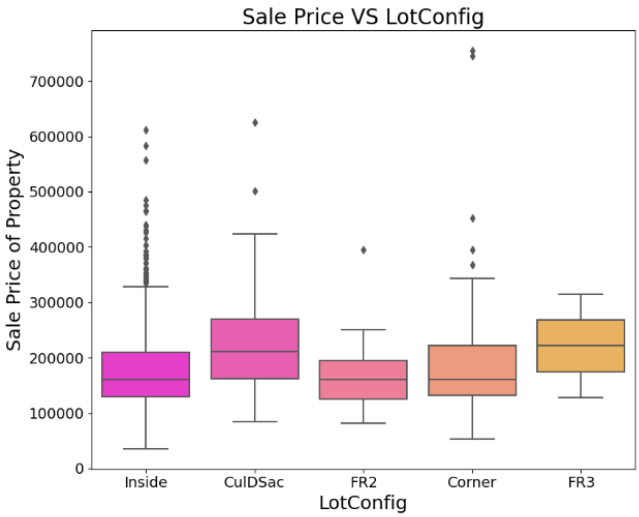


Observation:

• 89.6% of House properties are near flat level surface

•Price for Flat level surface house is much higher than other land contour types.



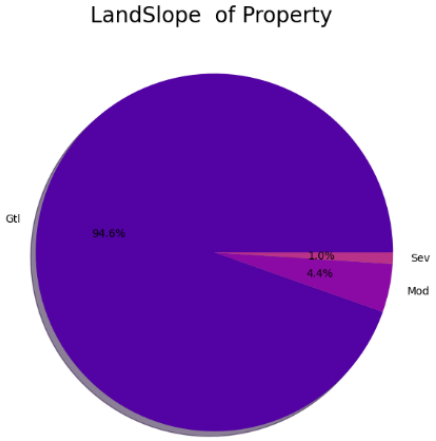


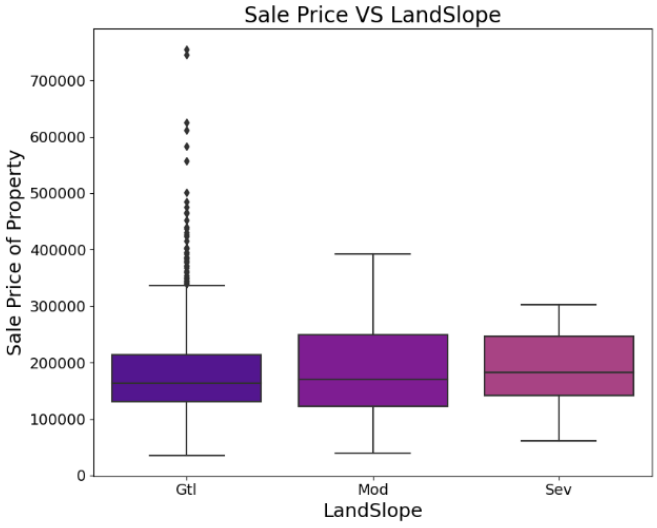
Observation: -

• Around 72 % of house comes with inside Lot configuration.

• Cul-de-sac has maximum Mean Sale Price among all lot configuration.

• Cheapest Houses belong to Inside lot configuration while Costlier houses belongs to Corner Lot Configuration.

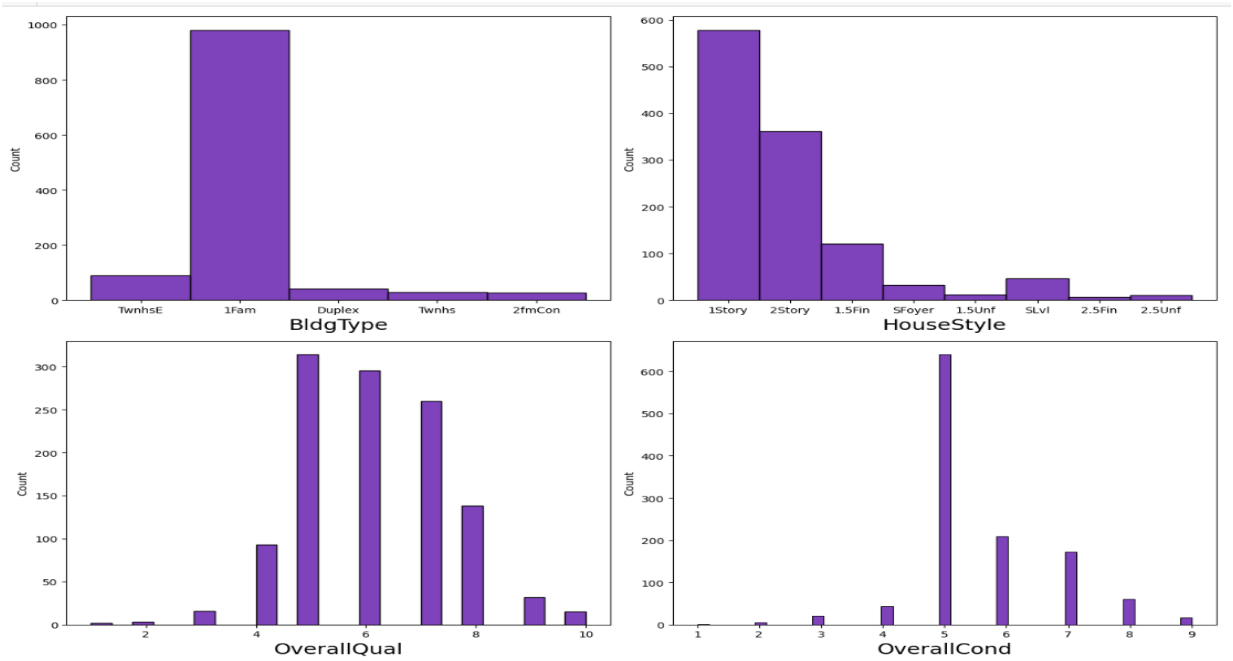




Observation:

• Clearly, we can see in boxplot that as Land slope increases the Sale price of house decreases

• 1% properties come with severe slope and they come with low price compare to Gentle Slope properties.

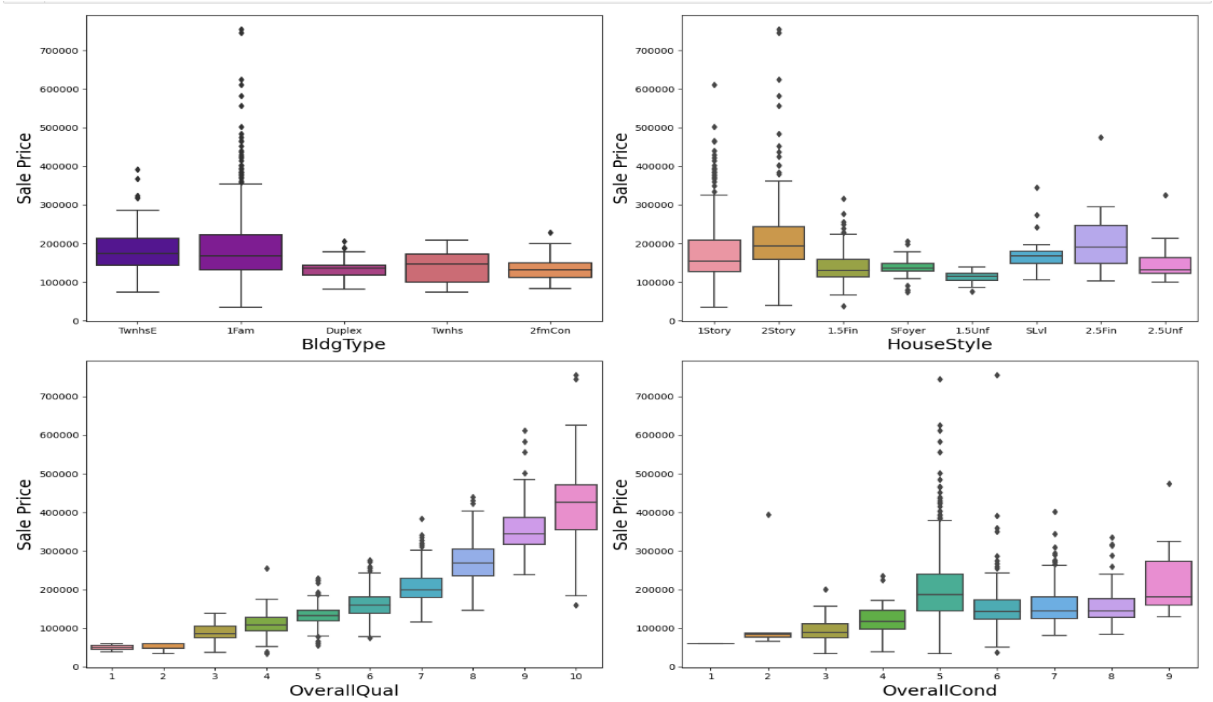


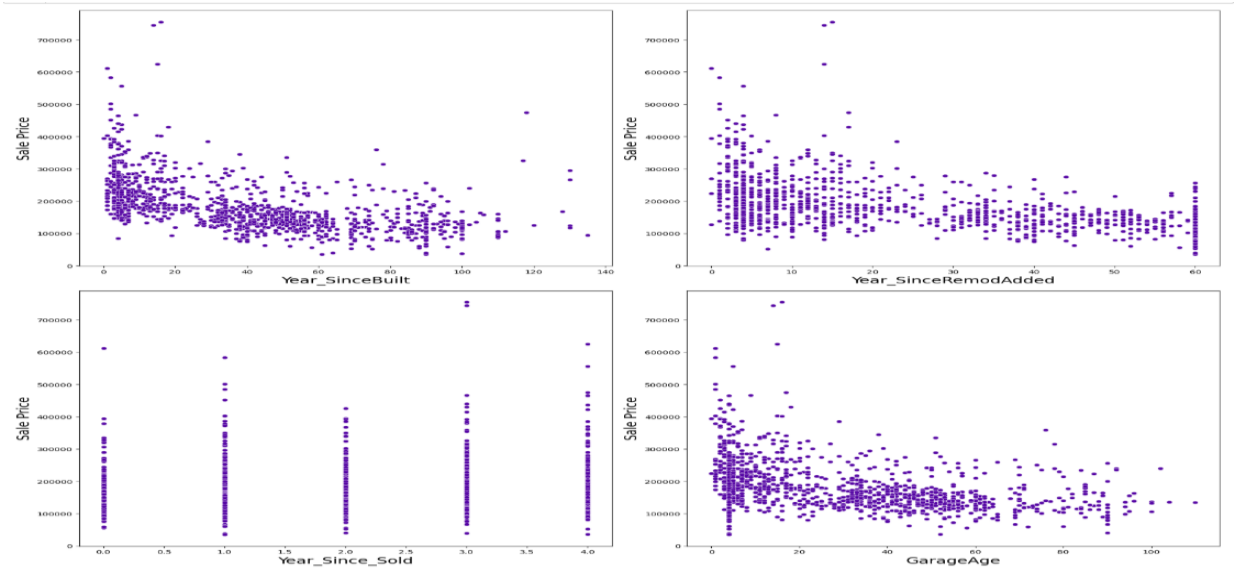
Observation:

• More than 950 house properties are with building type Singlefamily Detached.

• More than 50% of house properties comes with Overall Condition Rating of 5.

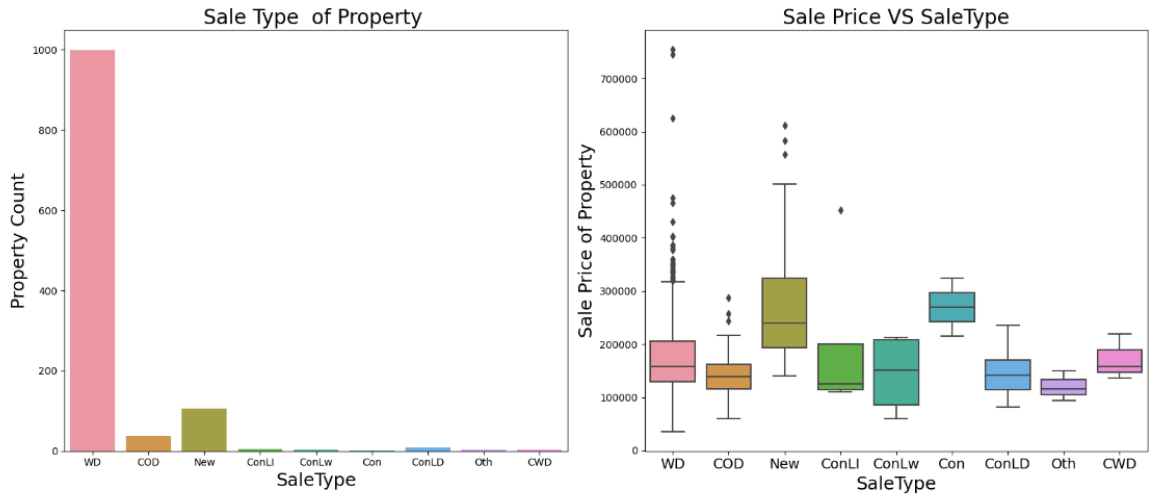
• More than 75% of house properties come with overall Quality Rating varies between 5 to 6.



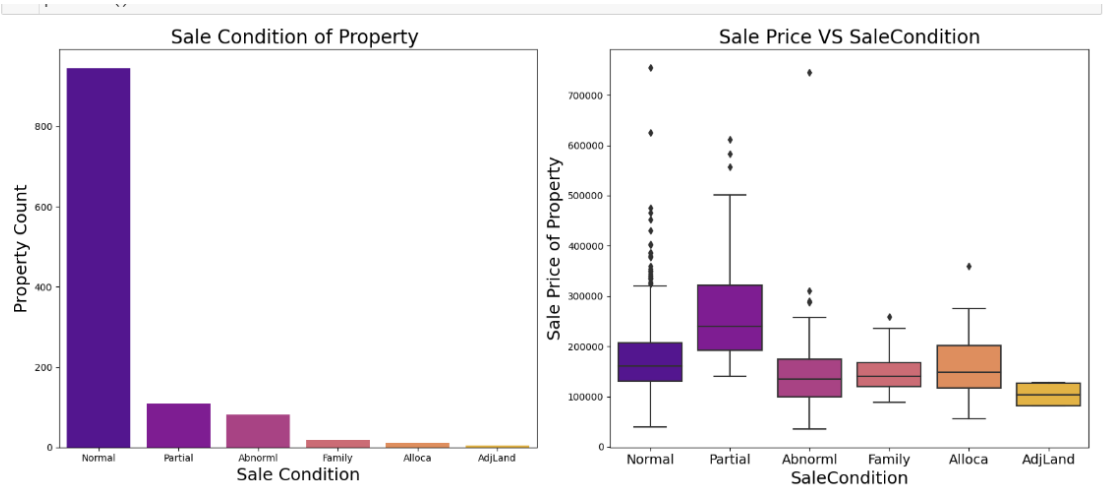


Observation: -

- We can see that as Property get older with time its sale Price get deprecates.
- 20 years after Remodelling Price of properties start decreases.
- Older the age of garage lesser the price of Property

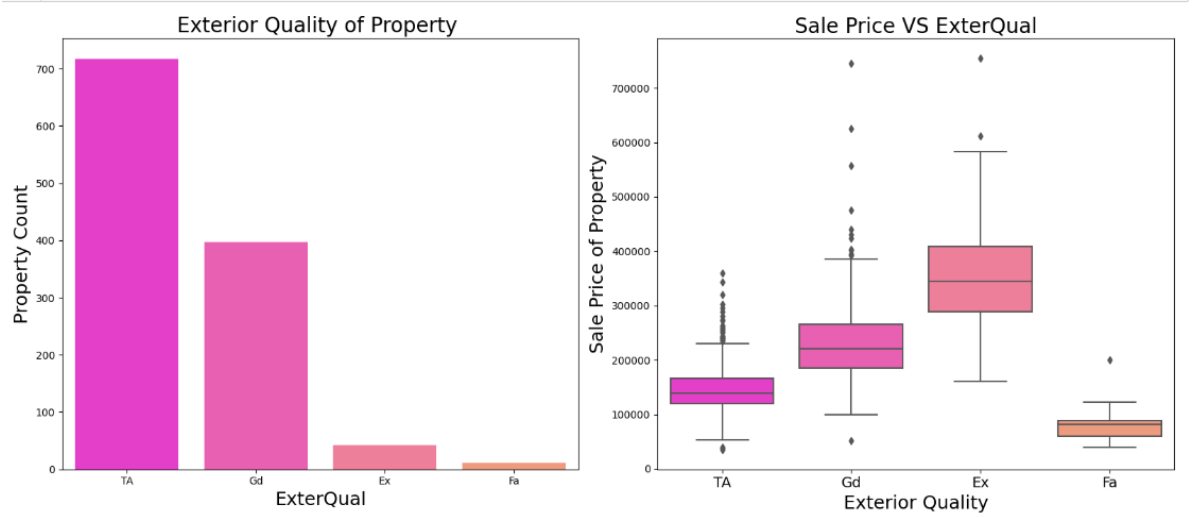


- Around 1000 sales happen by Conventional Warranty Deed. • House constructed and sold category are exceptionally much costlier than anyone else.



Observation: - • Sale with condition like Abnormal, Family, Allocate, AdjLand are below the price of 300000.

• Maximum Base Price for House comes from Partial categoryHome was not completed when last assessed (associated with New Homes) is higher than rest.

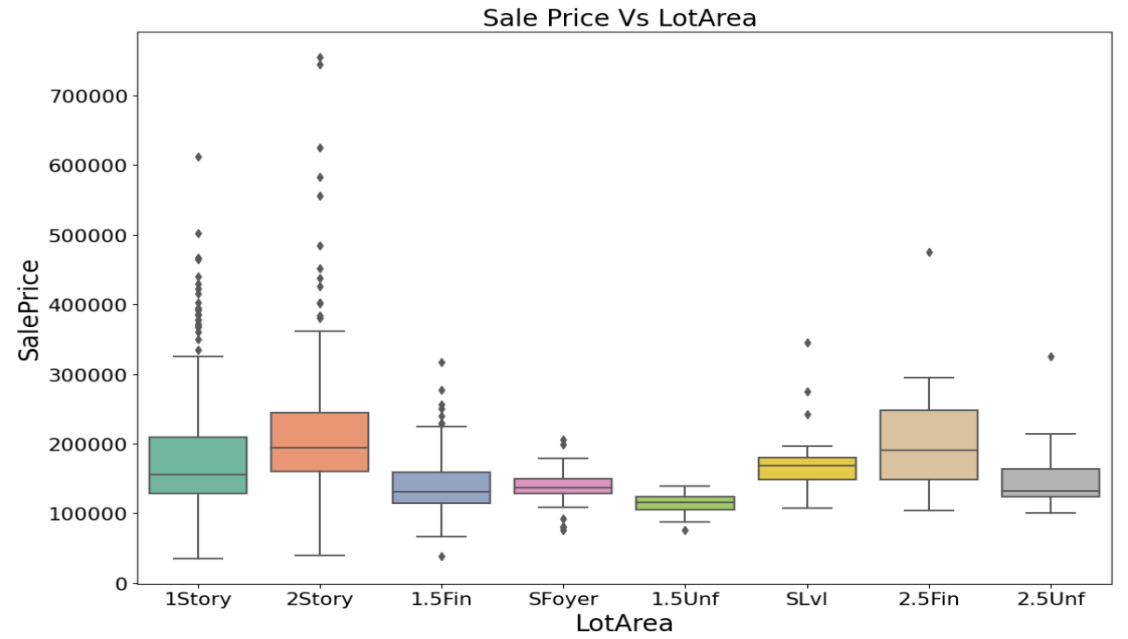


Observation: -

▪ Around 60% of house properties come with Average Exterior quality and all of them below 400000.

▪ Very few House Properties comes with Excellent Exterior Quality.

▪ Costlier house properties come with Good & Excellent exterior quality



Observation: -

- As Basement Quality increase in relation to it, sale Price also get increases.
- Two Story Building are costlier than remaining.