NAME OF THE PROJECT

Housing Price Prediction and Analysis Project

Submitted by:

Amruta Dongare

ACKNOWLEDGMENT

I would like to express my special gratitude to “Flip Robo” team, who has given me this opportunity to deal with a dataset and it has helped me to improve my analyzation skills. And I want to express my gratitude to Ms. Khushboo Garg (SME Flip Robo)

who has helped me while doing the project.

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. Notes provided by the Data Trained Institute

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

6. Hybrid Regression Technique for House Prices Prediction”, @2017

7. Zhen Peng, Qiang Huang, Yincheng Han, “Model Research on Forecast of Second-Hand House Price in Chengdu Based on XGboost

Algorithm

INTRODUCTION

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 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. In general, purchasing and investing in any real estate project will involve various transactions between different parties. Thus, it could be a vital decision for both households and enterprises. How to construct a realistic model to precisely predict the price of real estate has been a challenging topic with great potential for further research. 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.

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.

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 based on 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 difficult task and generally not very accurate, hence there are many systems developed for house price prediction.

Sifei Lu, Zengxiang Li, Zheng Qin, Xulei Yang, Rick Siow Mong Goh [6] had proposed an advanced house prediction system using linear regression. This system's aim was to make a model that can give us a good house price prediction based on other variables. This paper proposed on Hybrid Regression technique for housing Prices Prediction focused on the use of creative feature engineering to find the optimal features and their correlation with Sales Prices. Feature engineering improved the data normality and linearity of data. Their system showed that working on the Ames Housing dataset was convenient and showed that the use of Hybrid algorithms (65% Lasso and 35% Gradient Boost) provided results in predicting the house prices rather than using one from lasso, ridge or gradient boost.

CH. Raga Madhuri, Anuradha G et.al [7], estimated house price by the analysis of fare ranges, foregoing merchandise and forewarns of developments. The author discussed diverse regression techniques such as Gradient boosting and AdaBoost Regression, Ridge, Elastic Net, Multiple linear, LASSO to locate the most excellent. The performance measures used are [MSE] Mean Square Error and [RMSE] Root Mean Square Error.

Zhongyun, Jiang, Guoxin, Shen [8], develops 6-layer BP neural network with Kera's deep learning. For backend Tens flow or Theano is used. The test results give the predict real value of house with accuracy of 95.59%. The Gaussian noise model is favourable for the house price forecast.

Zhen Peng, Qiang Huang, Yincheng Han [9], precisely study the cost of recycled houses, and examined 35417 bits of information caught by the Chengdu HOME LINK organization. First and foremost, the caught information was cleaned, and the attributes were chosen. The test results show that the precision of XGboost expectation is the most elevated, and the forecast exactness score arrives at 0.9251. Dissimilarity with linear regression, decision tree model, the XGboost algorithm gives improved speculation capacity and strength in information forecast, and, furthermore, data prediction over-fitting aspect, establishing a strong framework for the ensuing recycled house value expectation.

Liu et al. [10] have constructed a statistical model based on the fuzzy neural network prediction model, which incorporates the hedonic theory and a great database with relevant characteristics affecting the price of properties based on recently sold projects. The experimental outcome and analysis have shown that the fuzzy neural network prediction model has a promising ability for real estate price prediction given reliable input data with high quality.

According to Kusan et al. [11], factors affecting housing sale price can be classified into three types: house factors, environmental factors, and transportation factors. The most influential type is residential factors, including residence, usability, and number of rooms. When people consider purchasing a house for living purposes, the factors above are the main determinants for the living quality. Buyers with family members would typically attach more importance to the essential feature of the house, like the living area and number of rooms, which have a significant impact on the overall living quality and experience in the house. Besides, the intangible features, like the view of residence and usability, also have a rather considerable influence on the housing price, through affecting buyers' experience on the house and willingness to pay. The other influential types are the main factors related to building properties and floor factors. Building properties are mainly about hardware and basic facilities in the building, such as the elevator, generator, and garage. The rising trend of numbers of vehicles per household possessing generates a necessary demand for the quality and capacity of a garage in a house. Other affiliated facilities to the house like the swimming pool and backyard have also played an essential role in determining the housing price, as the demand for leisure and relaxation has been arising with the economic progress. On the other hand, floor factors, like the number of floors, have also impacted the housing price significantly. A family with children and elders tends to prefer a house with multi floor construction, which offers different family members separate living areas with appropriate privacy while living together. Environmental factors mainly consist of two parts: regional environment and nearby pollution. Regional Scientific Pro environment refers to the overall living conditions in the surrounding community. Sanitation, as a significant indicator of the living quality, has been given more importance in the recent decades community

with comprehensive sanitation services tends to attract buyers to pay a higher price.

According to the paper proposed by Aayush Varma et. al. [12] suggested that the use of neural networks along with linear and boosted algorithms improved prediction accuracy. Three algorithms were used namely Linear Regression, Forest Regression and Boosted Regression. The dataset was tested on all three and the results of all the above algorithms were fed as an input to the neural network. Neural networks were used mainly to compare all the predictions and display the most accurate result. A neural network along with Boosted Regression was used to increase the accuracy of the result. Another research study showed that there exist relationships between the visual appearance and non-visual attributes such as crime statistics, housing prices, population density, etc. of a city. For instance, “City Forensics: Using Visual Elements to Predict Non-Visual City Attributes” [13], uses visual attributes to predict the sale price of the property.

1. Motivation for the Problem Undertaken

The project is 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 No Free Lunch Theorem state that algorithms perform differently when they are used under the same circumstances.

This study aims to analyse & predicting house prices when using Multiple linear, Lasso, Ridge, XGBoost, Random Forest regression and Extra Tree Regressor algorithms. Thus, 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?

2. Analytical Problem Framing

1. Mathematical/ Analytical Modelling of the Problem:

Our objective is to predict House price which can be resolve by use of regression-based algorithm. In this project we are going to use different types of algorithms which uses their own mathematical equation on background. This project comes with two separate data set for training & testing model. Initially data cleaning & pre-processing perform over data. Feature engineering is performed to remove unnecessary feature & for

dimensionality reduction. In model building Final model is select based on evaluation benchmark among different models with different algorithms. Further Hyperparameter tuning performed to build more accurate model out of best model.

2. Data Sources and their formats:

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.

```
In [6]: # Checking number of rows and columns of dataset
        df.shape

Out[6]: (1168, 81)
```

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.

```
In [94]: print('No. of Rows :',dft.shape[0])
         print('No. of Columns :', dft.shape[1])
         pd.set_option('display.max_columns',None) # this will enable us to see truncated columns
         dft.head()

         No. of Rows : 292
         No. of Columns : 80
```

The data types of different features are as shown below:

```
In [8]:  # Sort column by their datatype
         df.columns.to_series().groupby(df.dtypes).groups

Out[8]: {int64: ['Id', 'MSSubClass', 'LotArea', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd', 'BsmtFinSF1', 'BsmtFinSF
        2', 'BsmtUnfSF', 'TotalBsmtSF', '1stFlrSF', '2ndFlrSF', 'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBa
        th', 'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'TotRmsAbvGrd', 'Fireplaces', 'GarageCars', 'GarageArea', 'WoodDeckSF', 'Op
        enPorchSF', 'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'MiscVal', 'MoSold', 'YrSold', 'SalePrice'], float64:
        ['LotFrontage', 'MasVnrArea', 'GarageYrBlt'], object: ['MSZoning', 'Street', 'Alley', 'LotShape', 'LandContour', 'Utilitie
        s', 'LotConfig', 'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType', 'HouseStyle', 'RoofStyle', 'RoofMatl',
        'Exterior1st', 'Exterior2nd', 'MasVnrType', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual', 'BsmtCond', 'BsmtExposure',
        'BsmtFinType1', 'BsmtFinType2', 'Heating', 'HeatingQC', 'CentralAir', 'Electrical', 'KitchenQual', 'Functional', 'FireplaceQ
        u', 'GarageType', 'GarageFinish', 'GarageQual', 'GarageCond', 'PavedDrive', 'PoolQC', 'Fence', 'MiscFeature', 'SaleType', 'S
        aleCondition']}
```

3. Data Pre-processing Done:

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.

```
In [9]:  # Checking for duplicate
         df.duplicated().sum()

Out[9]: 0
```

Some features contain missing values as shown below.

```
In [13]: # 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
GarageYrBlt	64	5.479452
GarageFinish	64	5.479452
GarageType	64	5.479452
GarageQual	64	5.479452
GarageCond	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

We have removed feature which contain high amount missing values e.g., Top 5 features with missing value in above list.

Rest of feature are handle based on mean, median or mode imputation depending on outliers &distribution of feature.

In [100]:
```
dft['LotFrontage'] = dft['LotFrontage'].fillna(dft['LotFrontage'].median())
dft['GarageType'] = dft['GarageType'].fillna(dft['GarageType'].mode()[0])
dft['GarageYrBlt']= dft['GarageYrBlt'].fillna(dft['GarageYrBlt'].mode()[0])
dft['GarageFinish']= dft['GarageFinish'].fillna(dft['GarageFinish'].mode()[0])
dft['GarageQual'] = dft['GarageQual'].fillna(dft['GarageQual'].mode()[0])
dft['GarageCond'] = dft['GarageCond'].fillna(dft['GarageCond'].mode()[0])
dft['BsmtFinType2'] = dft['BsmtFinType2'].fillna(dft['BsmtFinType2'].mode()[0])
dft['BsmtExposure'] = dft['BsmtExposure'].fillna(dft['BsmtExposure'].mode()[0])
dft['BsmtFinType1'] = dft['BsmtFinType1'].fillna(dft['BsmtFinType1'].mode()[0])
dft['BsmtCond'] = dft['BsmtCond'].fillna(dft['BsmtCond'].mode()[0])
dft['BsmtQual'] = dft['BsmtQual'].fillna(dft['BsmtQual'].mode()[0])
dft['MasVnrType'] = dft['MasVnrType'].fillna(dft['MasVnrType'].mode()[0])
dft['MasVnrArea'] = dft['MasVnrArea'].fillna(dft['MasVnrArea'].median())
dft['Electrical'] = dft['Electrical'].fillna(dft['Electrical'].mode()[0])
```

- Feature extraction for age related feature

In [88]:
```
# Converting years column to age column
df['Year_SinceBuilt'] = df['YearBuilt'].max() - df['YearBuilt']
df['Year_SinceRemodAdded'] = df['YearRemodAdd'].max() - df['YearRemodAdd']
df['Year_Since'] = df['YrSold'].max() - df['YrSold']
df['GarageAge'] = df['GarageYrBlt'].max() - df['GarageYrBlt']
```

In [89]:
```
# Dropping old columns in train dataset
df.drop(['YearBuilt','YearRemodAdd','YrSold','GarageYrBlt'], axis=1, inplace = True)
```

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

```
BsmtFinSF1: Type 1 finished square feet
BsmtFinSF2: Type 2 finished square feet
BsmtUnfSF: Unfinished square feet of basement area
TotalBsmtSF: Total square feet of basement area
TotalBsmtSF is sum of above remaining features. We will drop other three features for modelling.
```

In [173]:
```
df.drop(['BsmtFinSF1','BsmtFinSF2','BsmtUnfSF'],axis=1,inplace=True)
dft.drop(['BsmtFinSF1','BsmtFinSF2','BsmtUnfSF'],axis=1,inplace=True)
```

```
1stFlrSF: First Floor square feet
2ndFlrSF: Second floor square feet
LowQualFinSF: Low quality finished square feet (all floors)
GrLivArea: Above grade (ground) living area square feet
GrLivArea is sum of above remaining features. We will drop other three features for modelling.
```

In [174]:
```
df.drop(['1stFlrSF','2ndFlrSF','LowQualFinSF'],axis=1,inplace=True)
dft.drop(['1stFlrSF','2ndFlrSF','LowQualFinSF'],axis=1,inplace=True)
```

```
Some of the features contain lot of zeros (above 90 %), it will meaningless to keep them while model building.
1.PoolArea - 1161 zeros out of 1168 entries
2.MiscVal - 1126 zeros out of 1168 entries
3.3SsnPorch - 1146 zeros out of 1168 entries
4.EnclosedPorch - 999 zeros out of 1168 entries
5.ScreenPorch - 1073 zeros out of 1168 entries
We will drop these columns from dataset.
```

n [175]:
```
df.drop(['EnclosedPorch','3SsnPorch','ScreenPorch','PoolArea','MiscVal'],axis=1,inplace=True)
dft.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 [184]:
```
# 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()
```

- Standard Scaling:

n [181]:
```
# Splitting data in target and dependent feature
X = df.drop(['SalePrice'], axis =1)
Y = df['SalePrice']
```

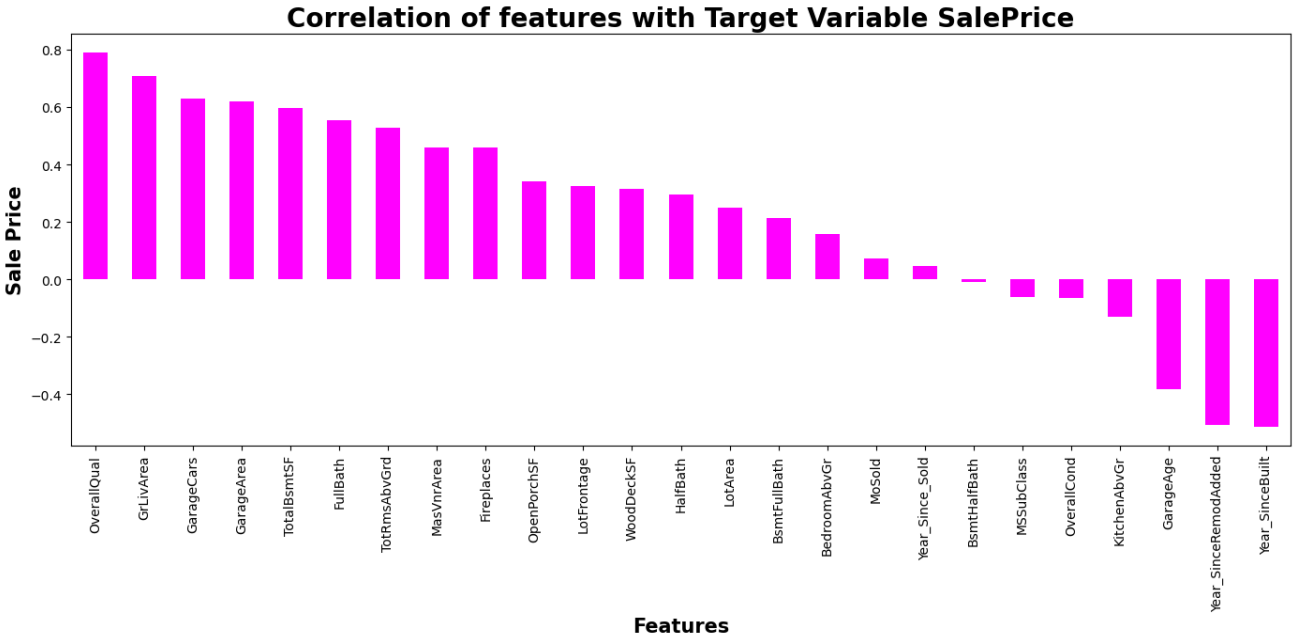
n [182]:
```
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
X_scale=scaler.fit_transform(X)
```

n [183]:
```
# Standard Scaling for Test Dataset
```

n [184]:
```
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
X_scale_test=scaler.fit_transform(dft)
```

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.



5. Hardware and Software Requirements and Tools Used

Software utilised -

1. Anaconda – Jupyter Notebook
2. Google Collab – for Hyper parameter tuning

Libraries Used – General libraries used for data wrangling.

In [1]:
```
# Importing libraries
import pandas as pd
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.

In [186]:
```
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
from xgboost import XGBRegressor
```

MODEL DEVELOPMENT AND EVALUTION

1. Identification of possible problem-solving 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.

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model. After that model is train with various regression algorithm and 5-fold cross validation is performed. Further Hyperparameter tuning performed to build more accurate model out of best model.

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

- ❖ Ridge Regression
- ❖ XGB Regressor

3. Run and Evaluate selected models.

1. Linear Regression:

In [190]:
```
# Linear Regression
```

In [191]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 135, test_size=0.33)
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 : 20097.055793096148
Mean squared error : 868718000.5387535
Root Mean squared error : 29474.022469604544
 R2 Score :
87.51862980338238
```

In [192]:
```
from sklearn.model_selection import cross_val_score
score = cross_val_score(lin_reg, X_scale, 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 : 0.7662072724233104
Difference in R2 & CV Score: 10.897902561051339
```

2. Random Forest Regressor:

In [193]:
```
# Random Forest Regressor
```

In [194]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 135, test_size=0.33)
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 : 17831.9396373057
Mean squared error : 678723275.4505689
Root Mean squared error : 26052.318043709063
R2 Score of Random Forest Regressor :
90.24839308417036
```

In [195]:
```
from sklearn.model_selection import cross_val_score
score = cross_val_score(rfc, X_scale, 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 : 0.8351501531423645
Difference in R2 & CV Score: 6.7333777699339095
```

3. Decision Tree Regressor:

In [196]:
```
# Decision Tree Regressor
```

In [197]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 135, test_size=0.33)
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 : 31273.818652849743
Mean squared error : 2855971940.8549223
Root Mean squared error : 53441.29434112653
R2 Score of Decision Tree Regressor :
58.966611670466264
```

In [198]:
```
from sklearn.model_selection import cross_val_score
score = cross_val_score(dtc, X_scale, 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 : 0.6782612075646521
Difference in R2 & CV Score: -8.859509085998951
```

4. Extra Trees Regressor:

In [199]:
```
# Extra Trees Regressor
```

In [200]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 135, test_size=0.33)
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 : 18152.370518134714
Mean squared error : 739709943.3651228
Root Mean squared error : 27197.60914795863
R2 Score of Extra Tree Regressor :
89.37216261717455
```

In [201]:
```
from sklearn.model_selection import cross_val_score
score = cross_val_score(etc, X_scale, 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 : 0.8326753384322483
Difference in R2 & CV Score: 6.104628773949727
```

5. Ridge Regression:

In [202]:
```
# Ridge Regression
```

In [203]:
```
from sklearn.linear_model import Ridge
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 135, test_size=0.33)
rd = Ridge()
rd.fit(X_train, Y_train)
y_pred = rd.predict(X_test)
print('\033[1m'+ 'Error of XGB 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 XGB Regressor :'+'\033[0m')
print(r2_score(Y_test,y_pred)*100)
```

```
Error of XGB Regressor:
Mean absolute error : 20090.34214217832
Mean squared error : 868587960.7881672
Root Mean squared error : 29471.816380877633
R2 Score of XGB Regressor :
87.52049815912768
```

In [204]:
```
from sklearn.model_selection import cross_val_score
score = cross_val_score(rd, X_scale, Y, cv=5)
print('\033[1m'+'Cross Validation Score :',rd,":"+'\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 : Ridge() :

Mean CV Score : 0.7666037684644635
Difference in R2 & CV Score: 10.86012131268133
```

6. XGB Regressor:

In [205]:
```
# XGB Regressor
```

In [206]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state= 135, test_size=0.33)
xgb = XGBRegressor()
xgb.fit(X_train, Y_train)
y_pred = xgb.predict(X_test)
print('\033[1m'+ 'Error of XGB 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 XGB Regressor :'+'\033[0m')
print(r2_score(Y_test,y_pred)*100)
```

```
Error of XGB Regressor:
Mean absolute error : 19112.426550356216
Mean squared error : 927210436.1906452
Root Mean squared error : 30450.130314838476
R2 Score of XGB Regressor :
86.67823540310482
```

5-Fold cross validation performed over all models. We can see that Random Forest Regressor gives maximum R2 score of 89.57 and with cross validation score of 82.05 %. Among all model we will select Random Forest Regressor as final model

and we will perform hyperparameter tuning over this model to enhance its R2 Score.

```
In [208]: from sklearn.model_selection import GridSearchCV

In [209]: X_train, X_test, Y_train, Y_test = train_test_split(X_scale, Y, random_state=135, test_size=0.33)

In [210]: print(rfc.get_params())

{'bootstrap': True, 'ccp_alpha': 0.0, 'criterion': 'squared_error', 'max_depth': None, 'max_features': 'auto', 'max_leaf_nod
es': None, 'max_samples': None, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 1, 'min_samples_split': 2, 'min_weight_fra
ction_leaf': 0.0, 'n_estimators': 100, 'n_jobs': None, 'oob_score': False, 'random_state': None, 'verbose': 0, 'warm_start':
False}

In [211]: parameter = {
           'bootstrap': [True, False],
           'max_features': ['auto'],
           'min_samples_leaf': [1, 2, 4 ],
           'n_estimators': [100, 500,1000,1500,2000]}

In [212]: GCV = GridSearchCV(RandomForestRegressor(),parameter,verbose =10)

In [213]: GCV.fit(X_train,Y_train)

Fitting 5 folds for each of 30 candidates, totalling 150 fits
```

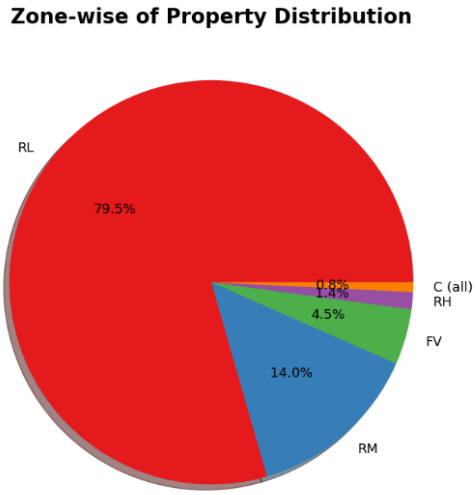
Final model is built with best params got in hyper parameter tuning.

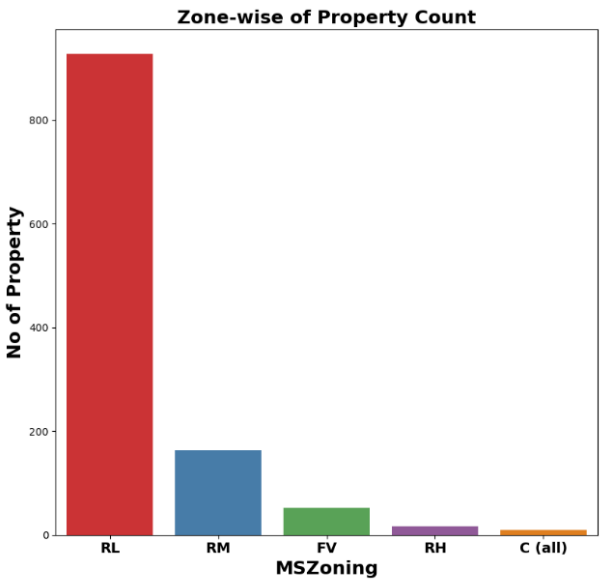
```
In [215]: # Final Model

In [216]: 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: 89.57580398396529
mean_squared_error: 725536265.459682
mean_absolute_error: 17560.851113989636
RMSE value: 26935.780394480535
```

We can see that hyper parameter tuning leading to increase in R2 Score slightly from default model.

4. Visualizations

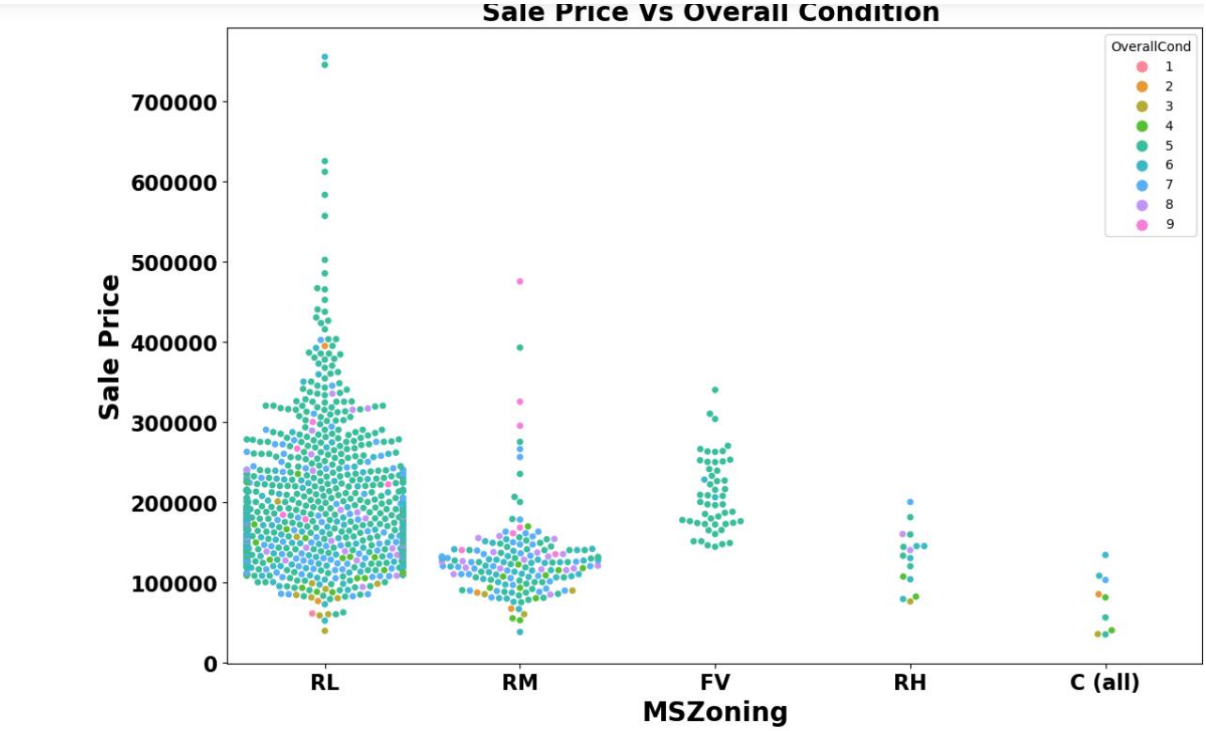
Let see key result from EDA, start with zone wise distribution of property.





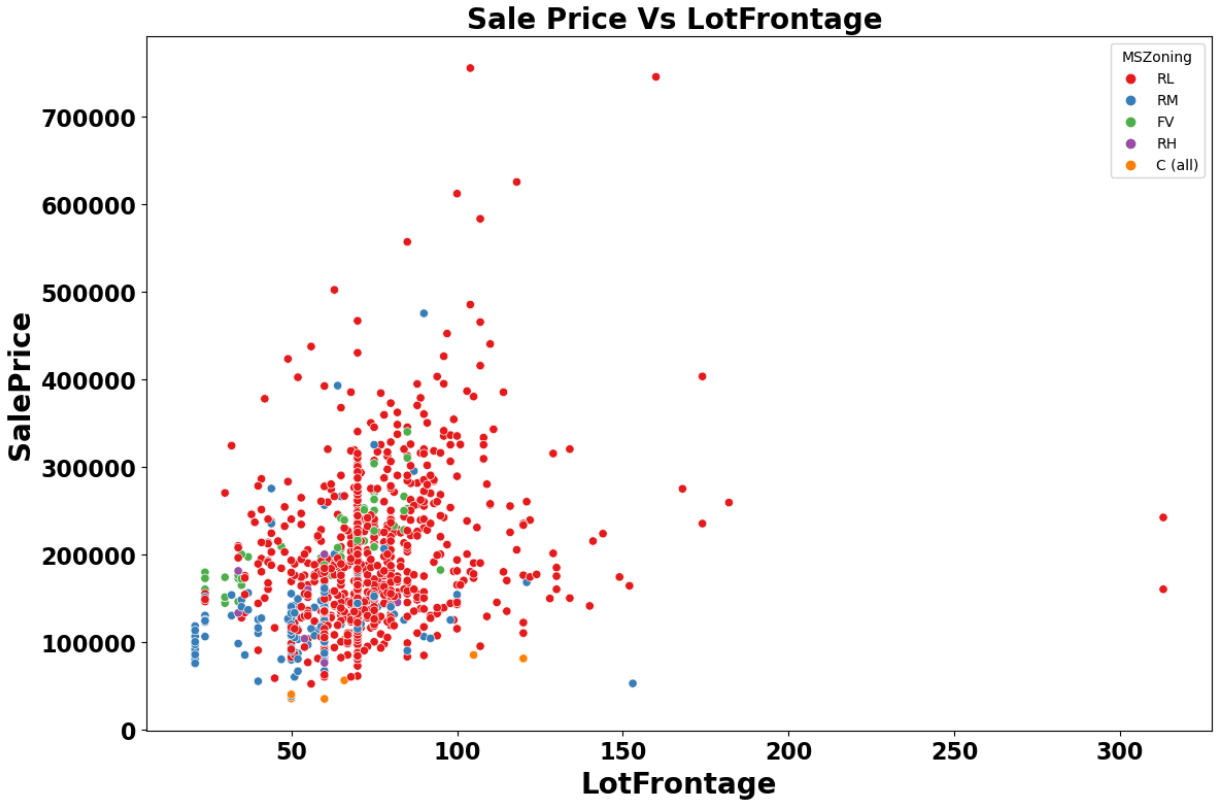
Observation:

- 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.

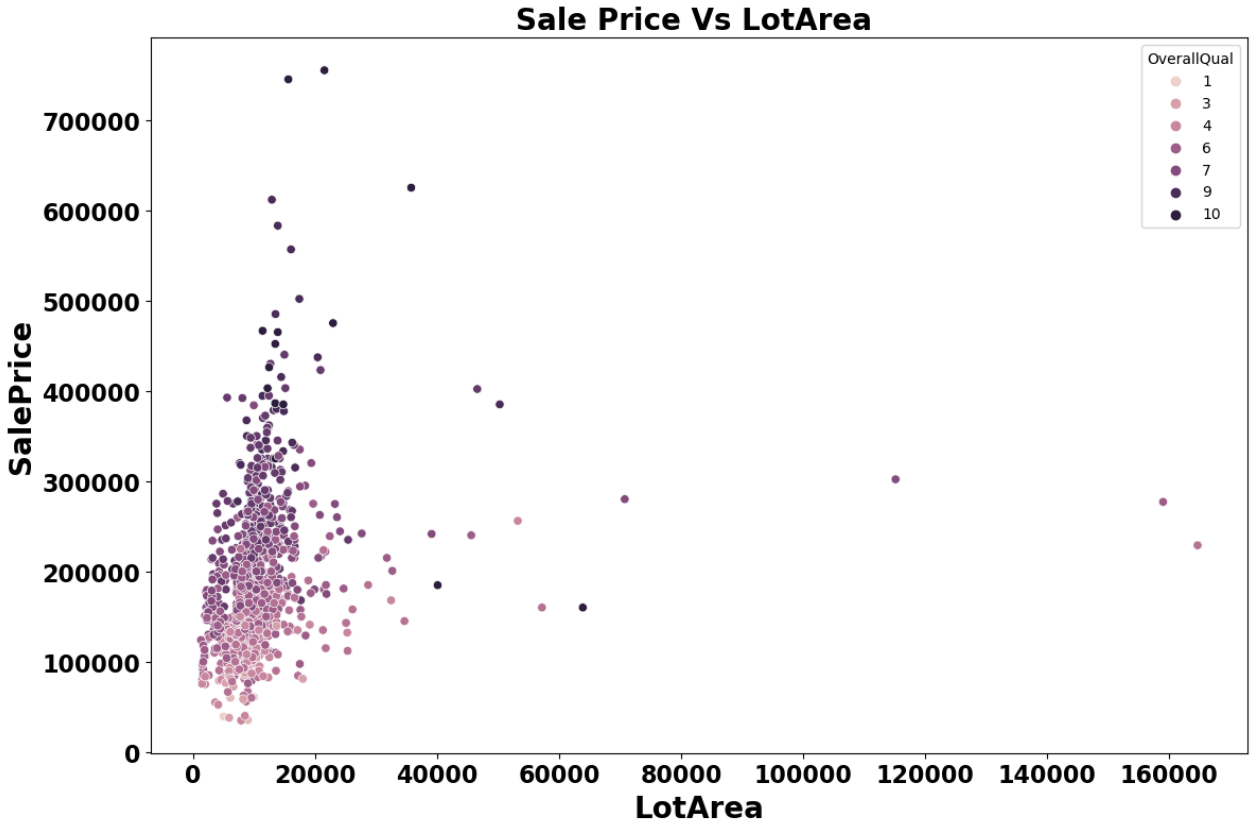


Observation:

- Most of property for sale have overall condition rating of either 5 or 6.
- We already know of 80% of housing data belongs to Low density Residential Area and Now we can see in Swarm Plot that Sale Price inside RL Zone is much higher than another remaining zone.
- Cheapest properties are available in Commercial zone.
- Another interesting observation we get here is for some house properties having Overall condition Rating of 8 & 9 have low price compared to others. This indicate that Overall Condition Rating is Not significant factor in determination of Sale price. Overall Condition Rating may be helpful to buyer in taking decision of Buying property but not in determination of House Price.



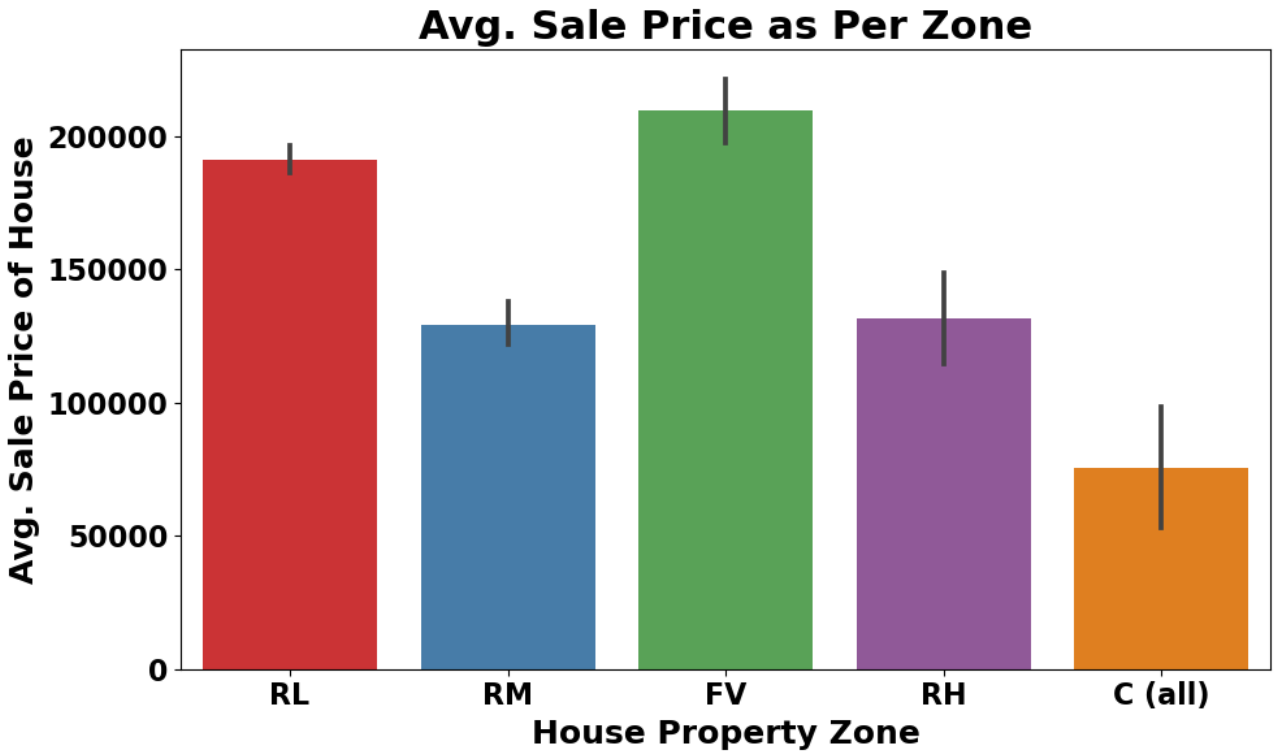
With Exception of Commercial zone, As Lot Frontage area increase (which indicate Size of street connected to property) the Sale Price increases.



Observation:

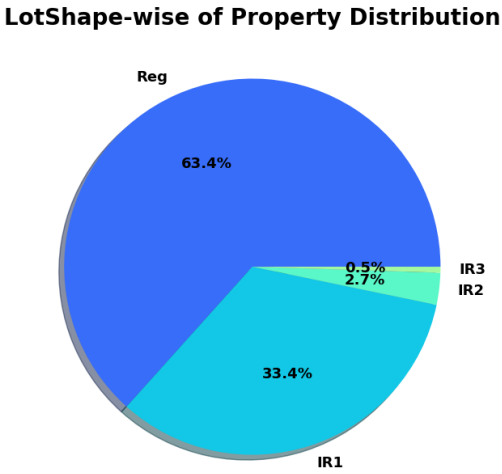
- There is No Significant relationship found between Sale price & Lot area.

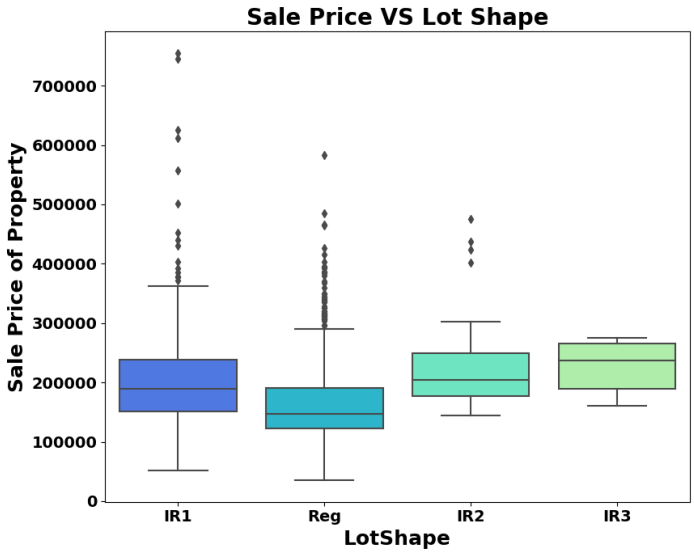
Here we get Important Observation that -
As Overall Quality of House Increase the Sale Price of House also Increase.



Observation:

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

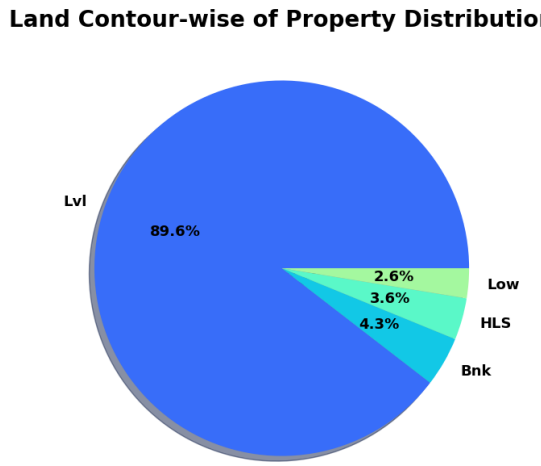


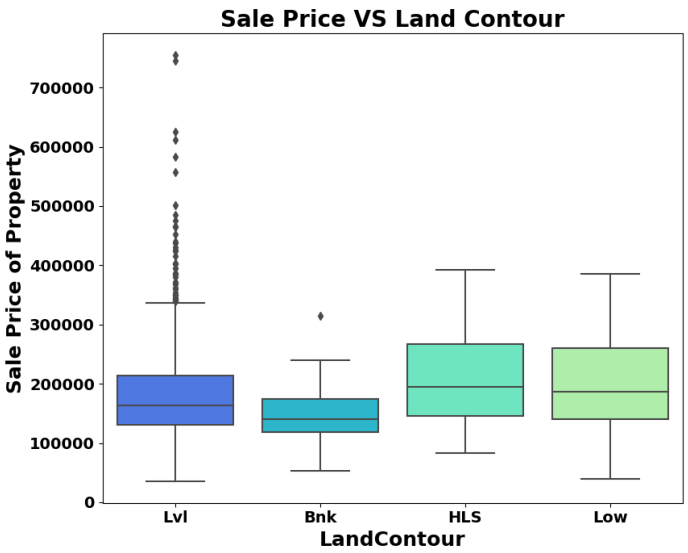


Observation:

- 63.4% house properties are regular in shape.

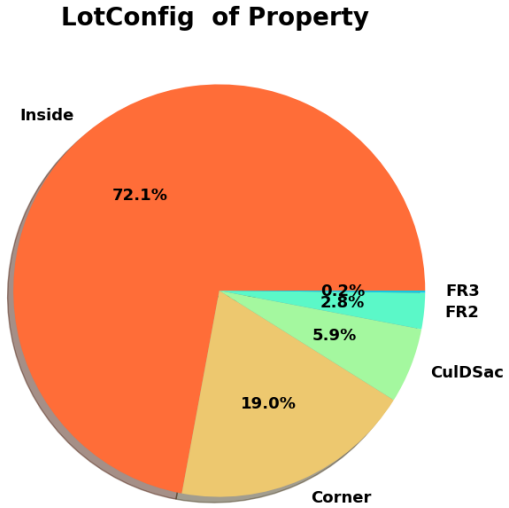
- Sale Price of property with slight irregular shape is higher than regular shape.

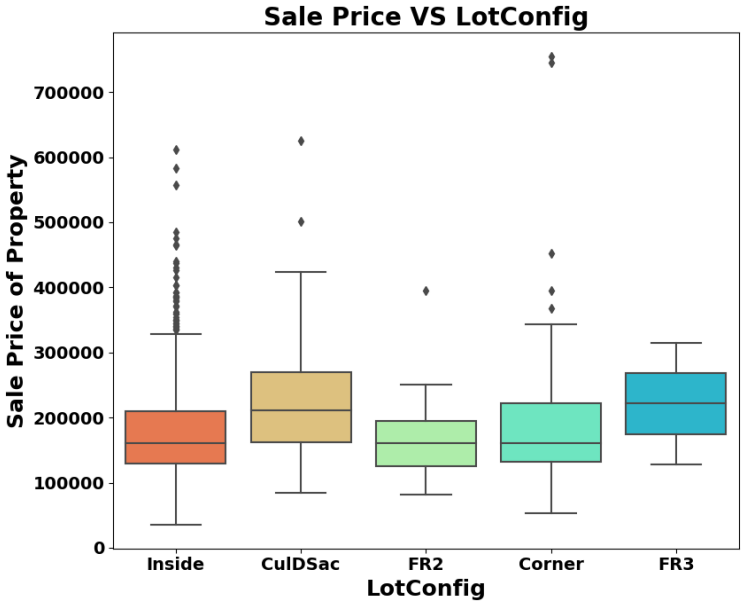




Observation:

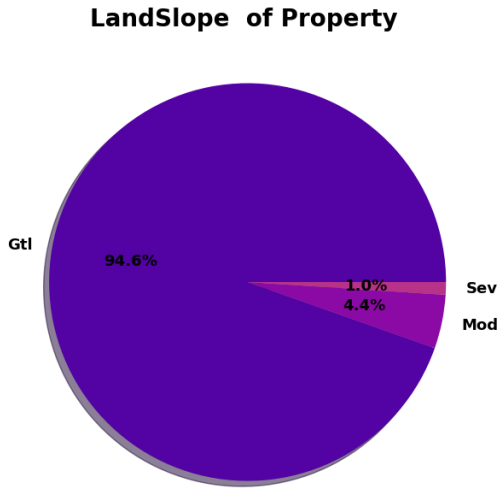
- 89.6% of House properties are near flat level surface.
- Also, price for Flat level surface house is much higher than other land contour.

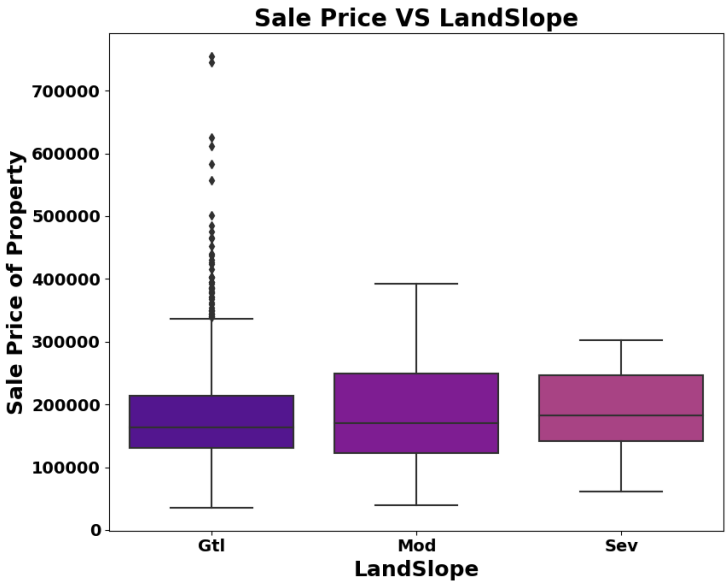




Observation: -

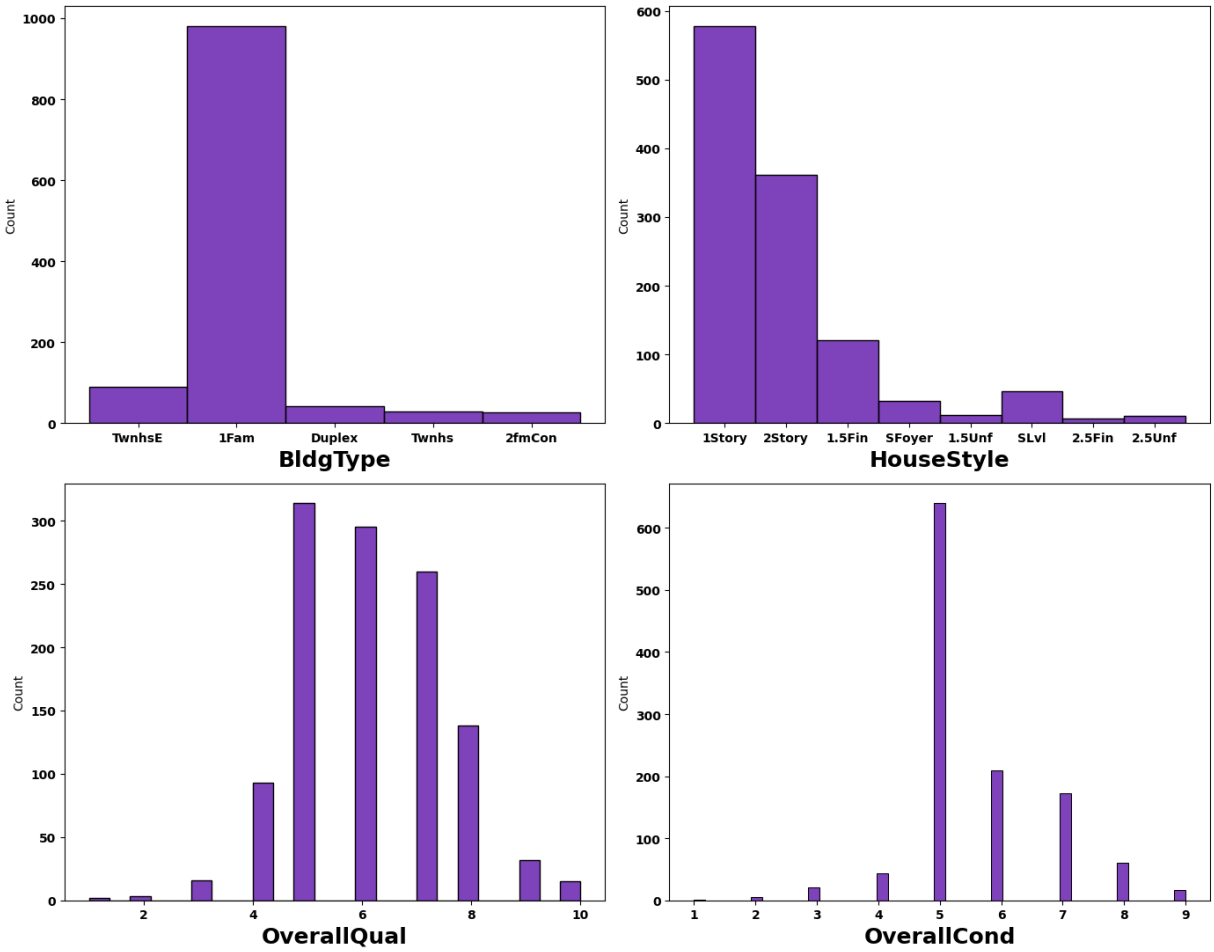
- Around 72 % of house comes with inside Lot configuration.
- Cul-de-sac has maximum Mean Sale Price among all lot configurations.
- Cheapest Houses belong to Inside Lot Configuration while Costlier houses belongs to Corner Lot Configuration.





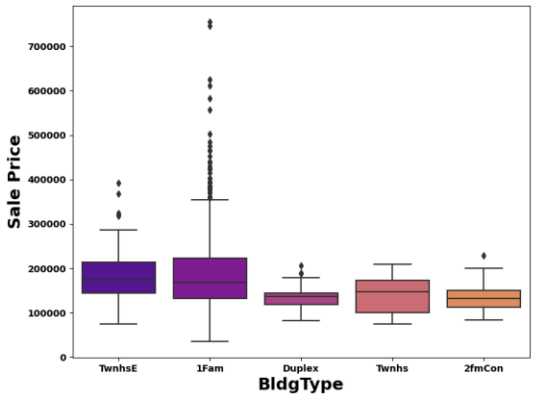
Observation:

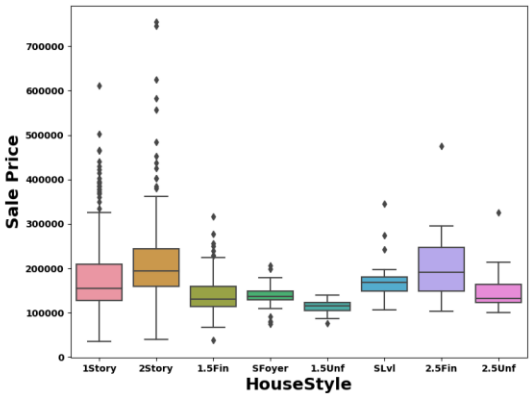
- Clearly, we can see in boxplot that as Land slope increases the Sale price of house decreases.
- 1% properties come with severe slope, and they come with low price compared to Gentle Slope properties.

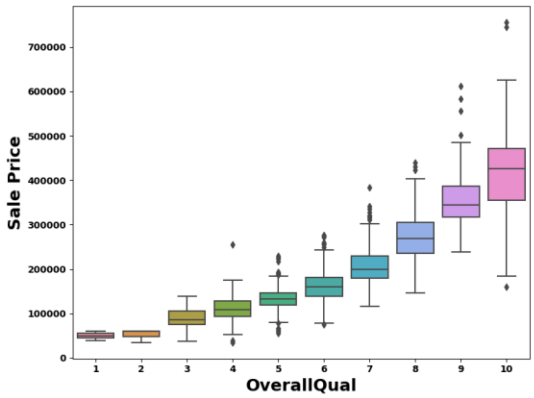


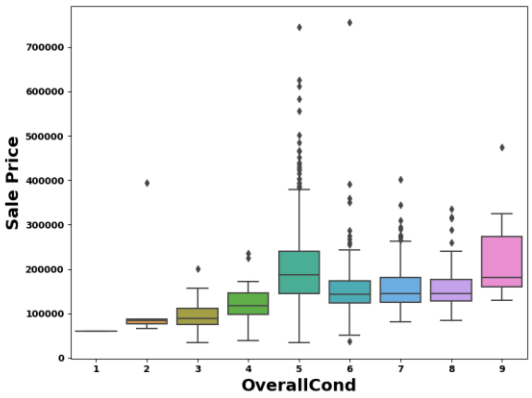
Observation:

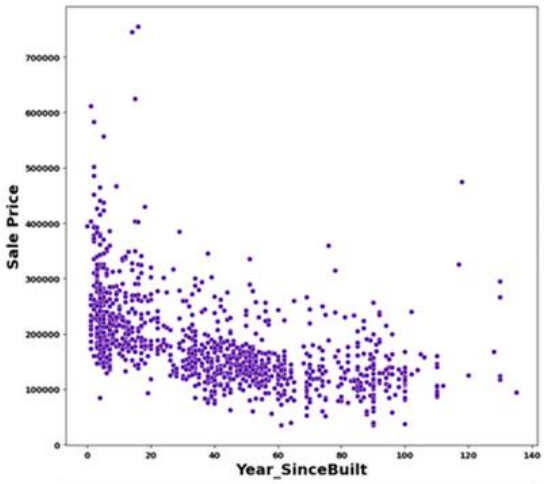
- More than 950 house properties are with building type Single family Detached.
- More than 50% of house properties comes with Overall Condition Rating of 5.
- More than 75% of house properties come with overall Quality Rating varies between 5 to 6.
- More than 500 House Properties comes with one story dwelling.

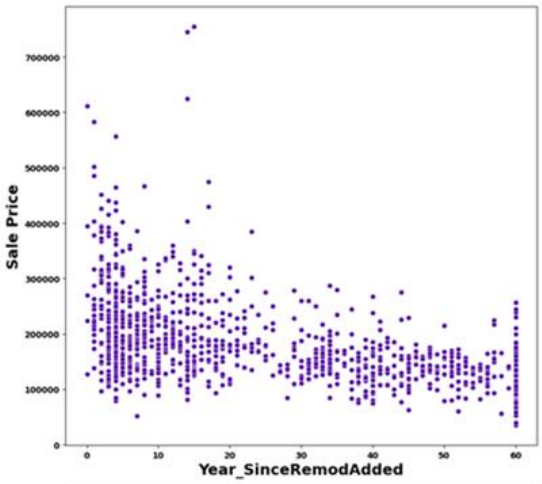

Sale Price
0
100000
200000
300000
400000
500000
600000
700000
TwnhsE
1Fam
Duplex
Twnhs
2fmCon
BldgType

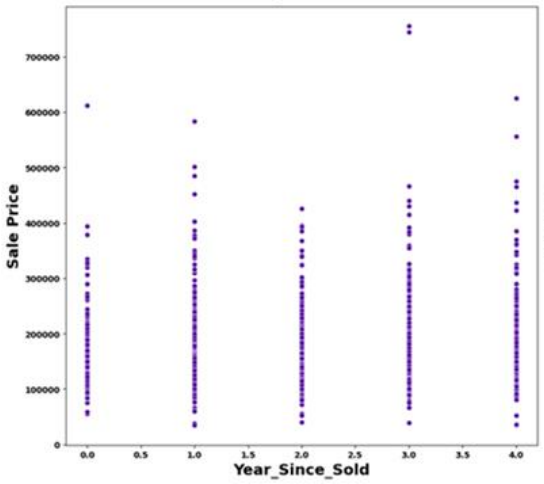

Sale Price
0
100000
200000
300000
400000
500000
600000
700000
1Story
2Story
1.5Fin
SFoyer
1.5Unf
SLvl
2.5Fin
2.5Unf
HouseStyle

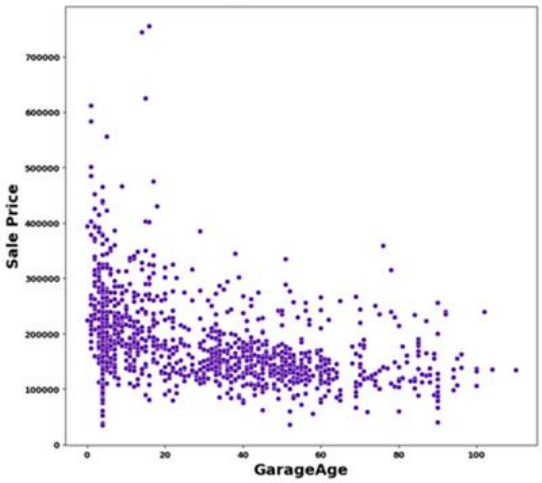

Sale Price
0
100000
200000
300000
400000
500000
600000
700000
1
2
3
4
5
6
7
8
9
10
OverallQual


Sale Price
0
100000
200000
300000
400000
500000
600000
700000
1
2
3
4
5
6
7
8
9
OverallCond


Sale Price
0
100000
200000
300000
400000
500000
600000
700000
0
20
40
60
80
100
120
140
Year_SinceBuilt

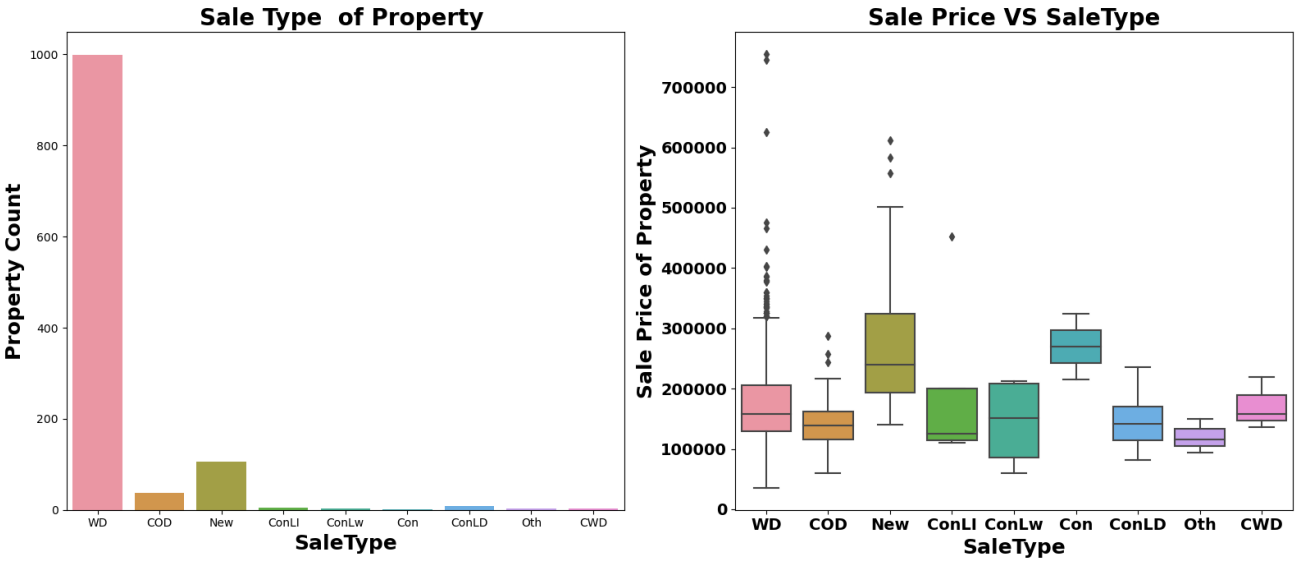

Sale Price
0
100000
200000
300000
400000
500000
600000
700000
0
10
20
30
40
50
60
Year_SinceRemodAdded


Sale Price
0
100000
200000
300000
400000
500000
600000
700000
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Year_Since_Sold


Sale Price
0
100000
200000
300000
400000
500000
600000
700000
0
20
40
60
80
100
GarageAge

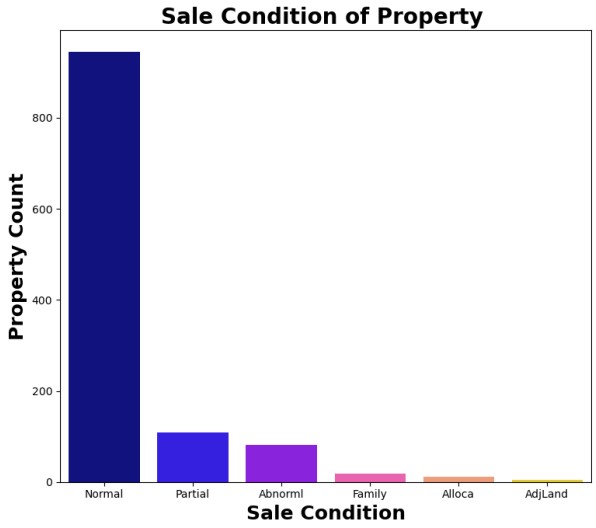
Observation: -

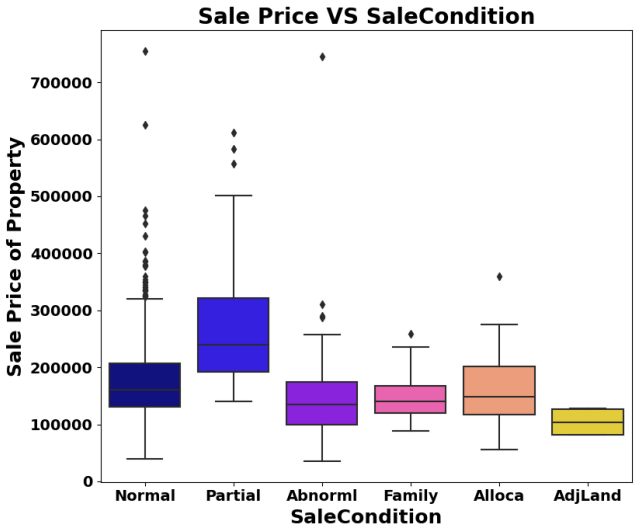
- We can see that as Property get older with time its sale Price get deprecates.
- 20 years after Remodelling Price of properties start decreases.
- Older the age of garage lesser the price of Property.



Observation: -

- Around 1000 sales happen by Conventional Warranty Deed.
- Home just constructed and sold category are exceptionally much costlier than anyone else.
- All loan-based sale is below 300000.



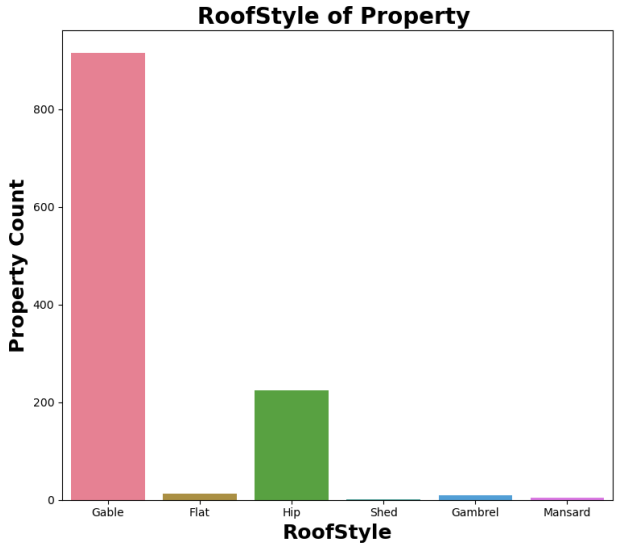


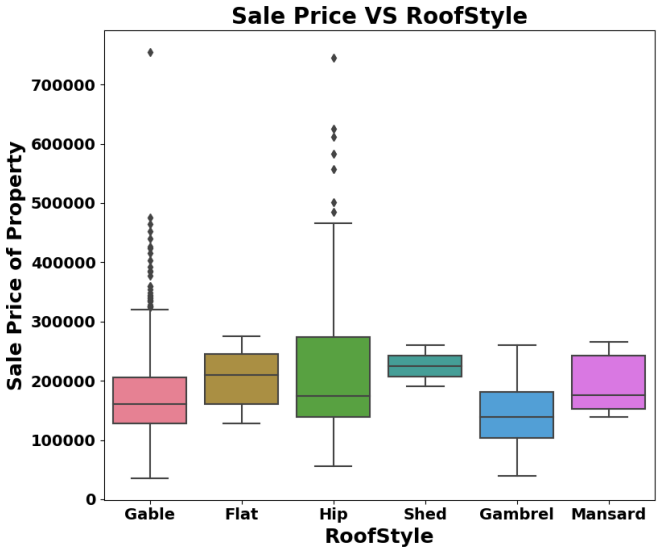
Observation: -

- We can see that Sale with condition like Abnorml, Family, Alloca, AdjLand are below the price of 300000.
- Maximum Base Price for House comes from Partial category Home was not completed when last assessed (associated with New Homes) is higher than rest.
- Minimum base price comes from Normal condition sale and also highest sale price comes from this category

Observation: -

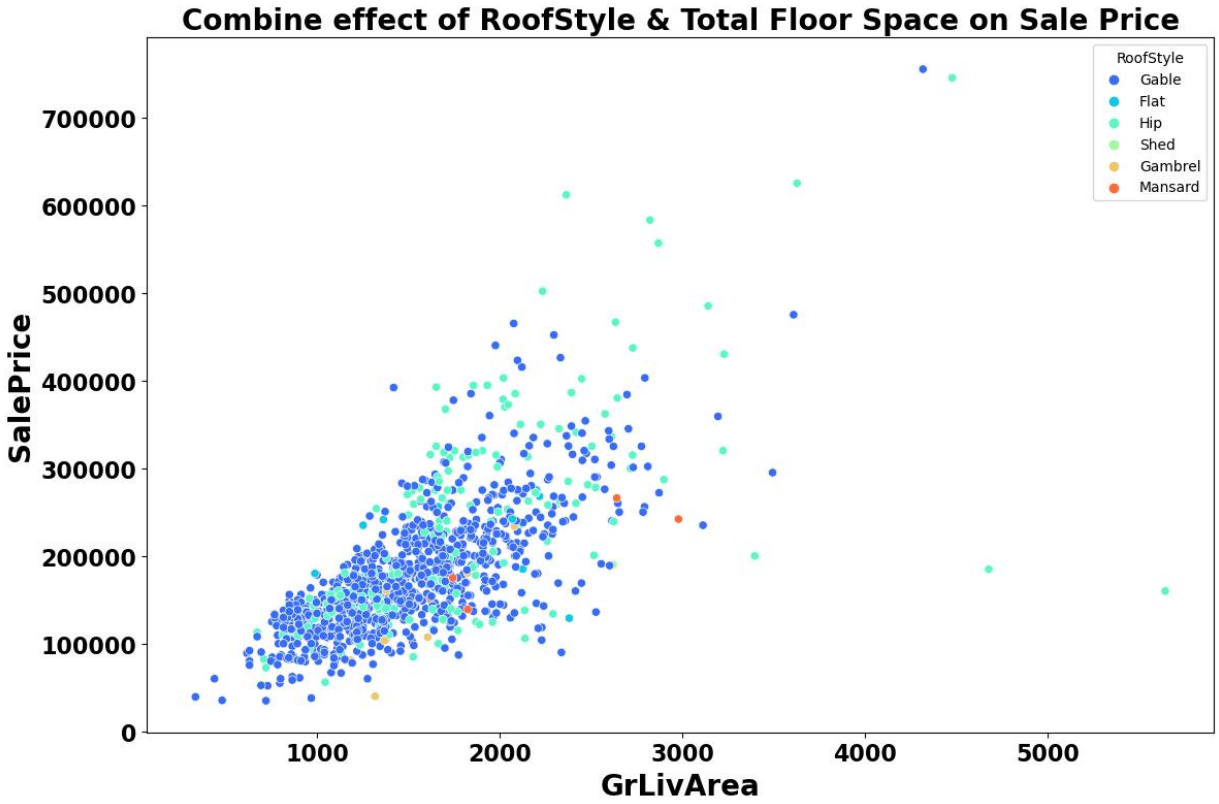
- In above plot we can clearly see relation between all three feature very clearly. As total floor area increases the sale price also get increase corresponding the overall quality of House.





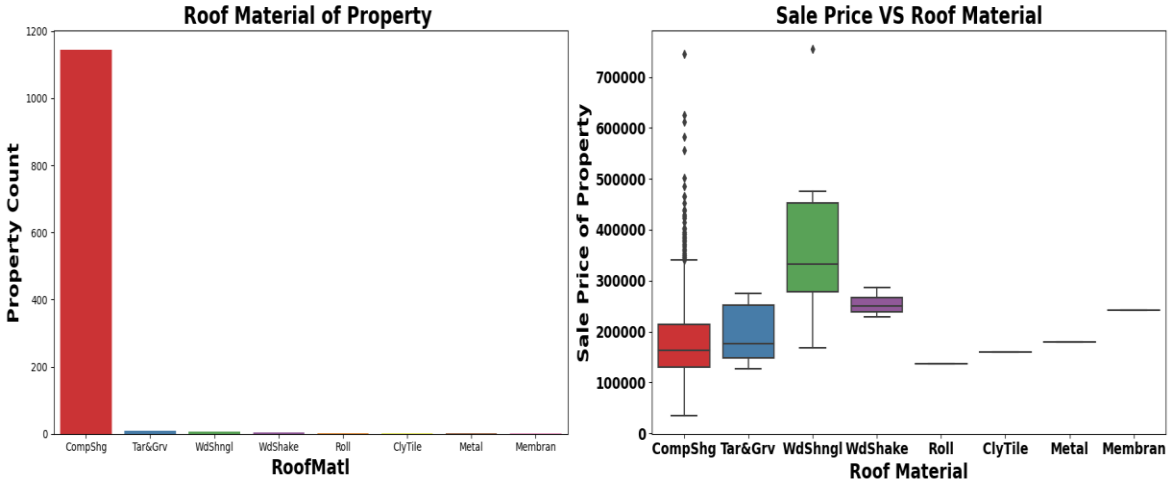
Observation: -

- More than 75% House properties come with Gable Roof Style followed by around 15 % house properties with Hip Style.
- From Boxplot we can see that Hip style Roof are much costlier than remaining roof style.



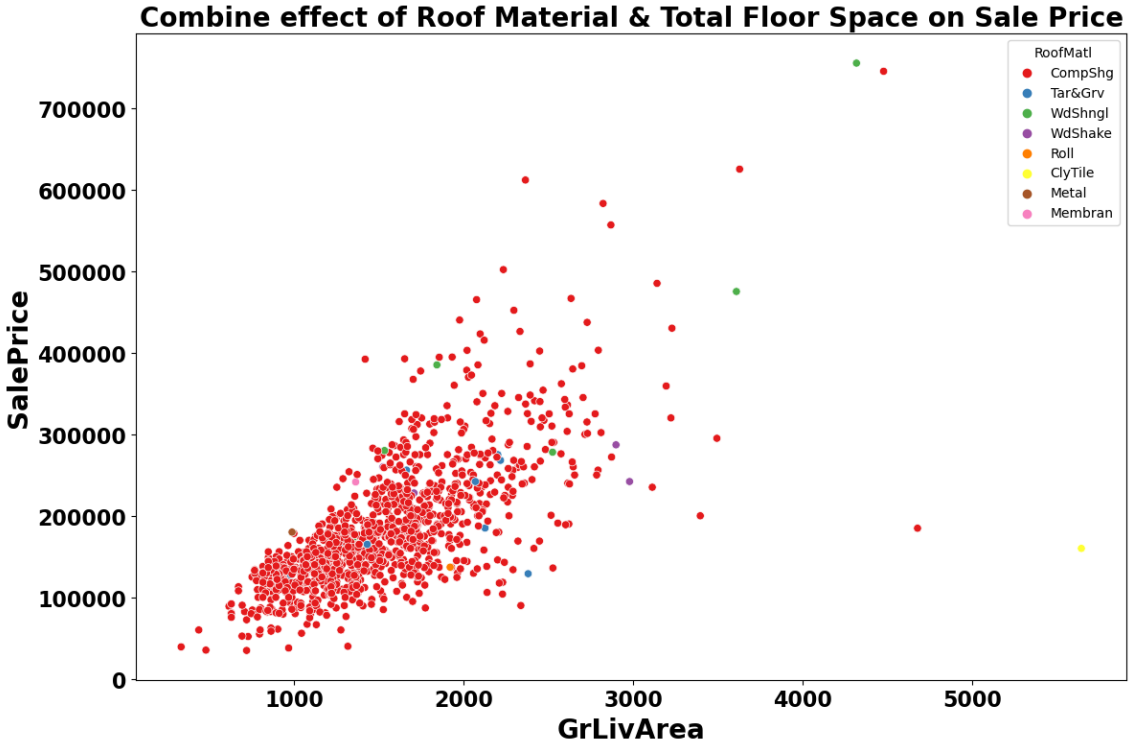
Observation: -

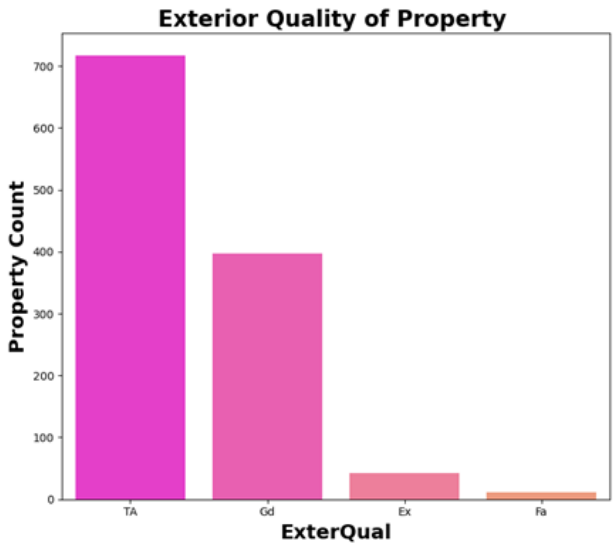
- For High floor area construction mainly Hip style Roof is used and invariably high-cost properties mostly comes up with Hip Style Roof.

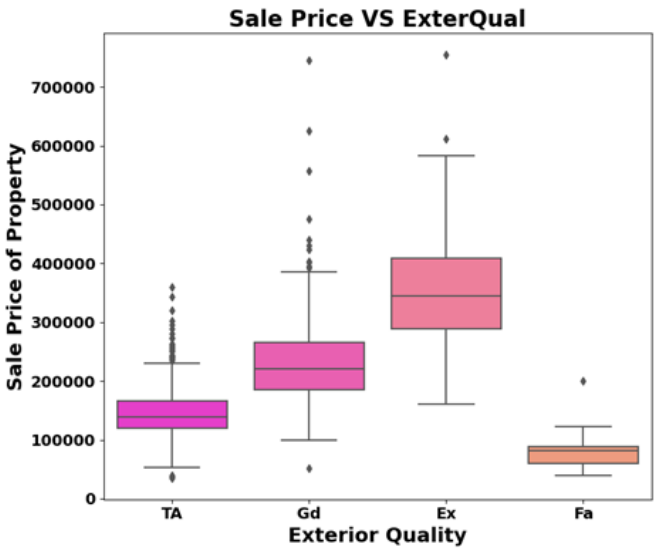


Observation: -

- More than 90% Properties in Data set made with roof material of Standard (Composite) Shingle.
- Wood Shingles is Costlier Material compared to rest.
-

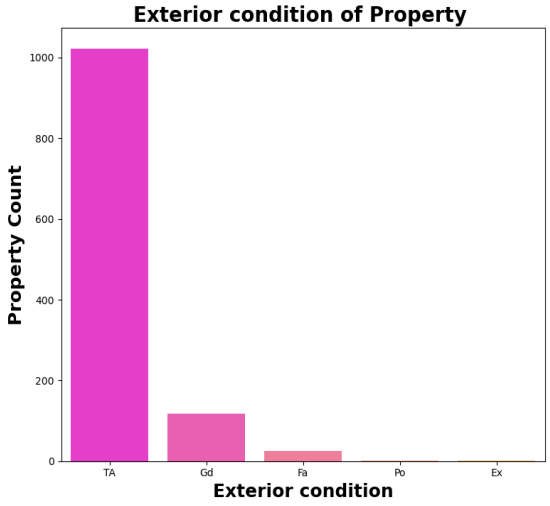


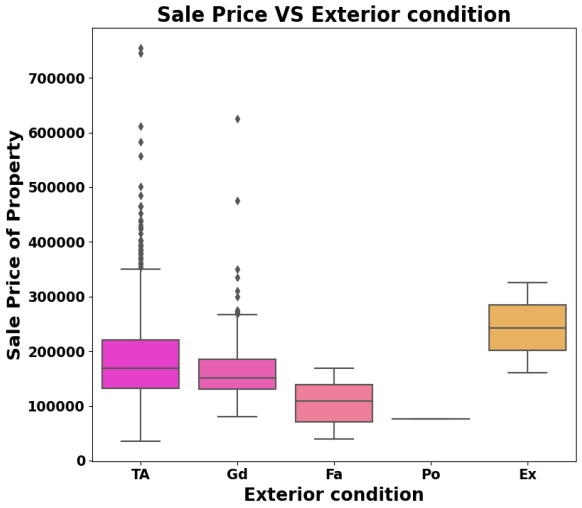


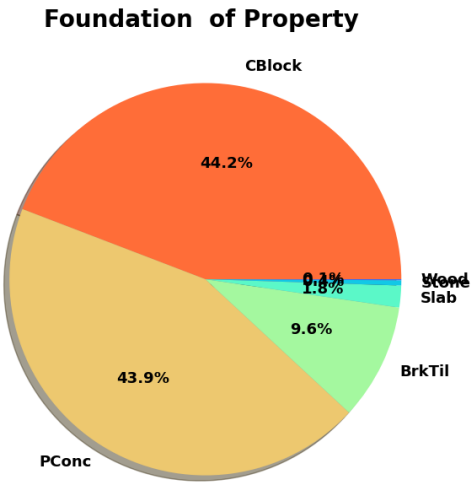


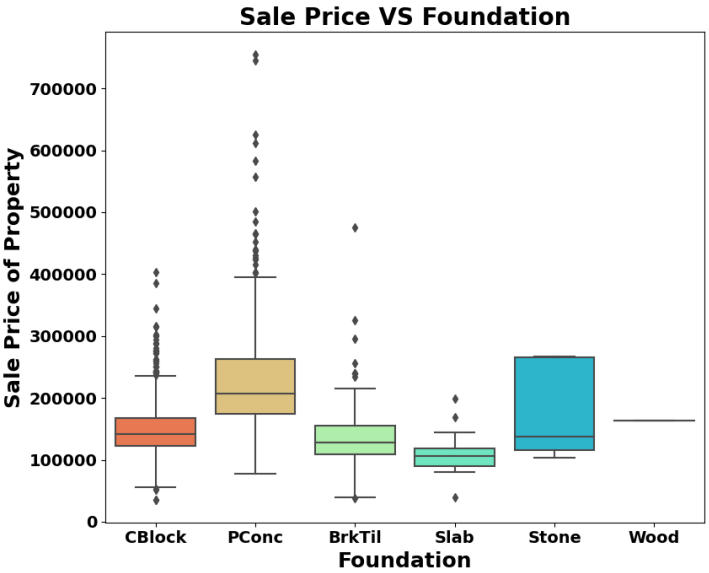
Observation: -

- 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.



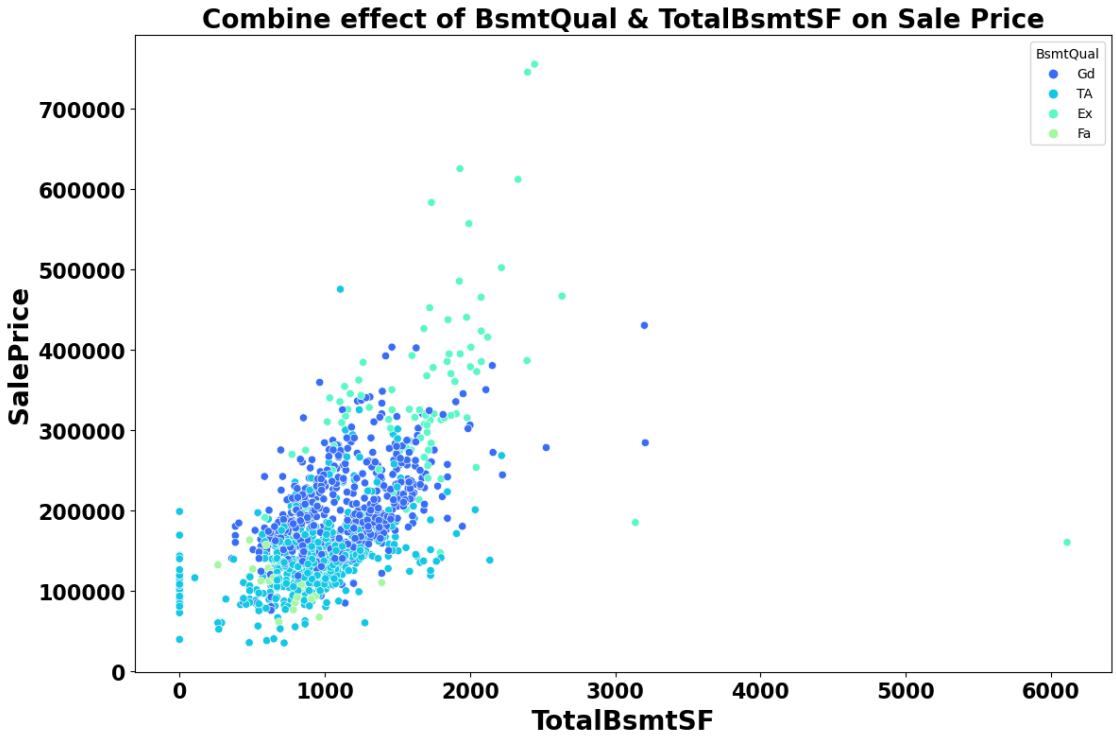






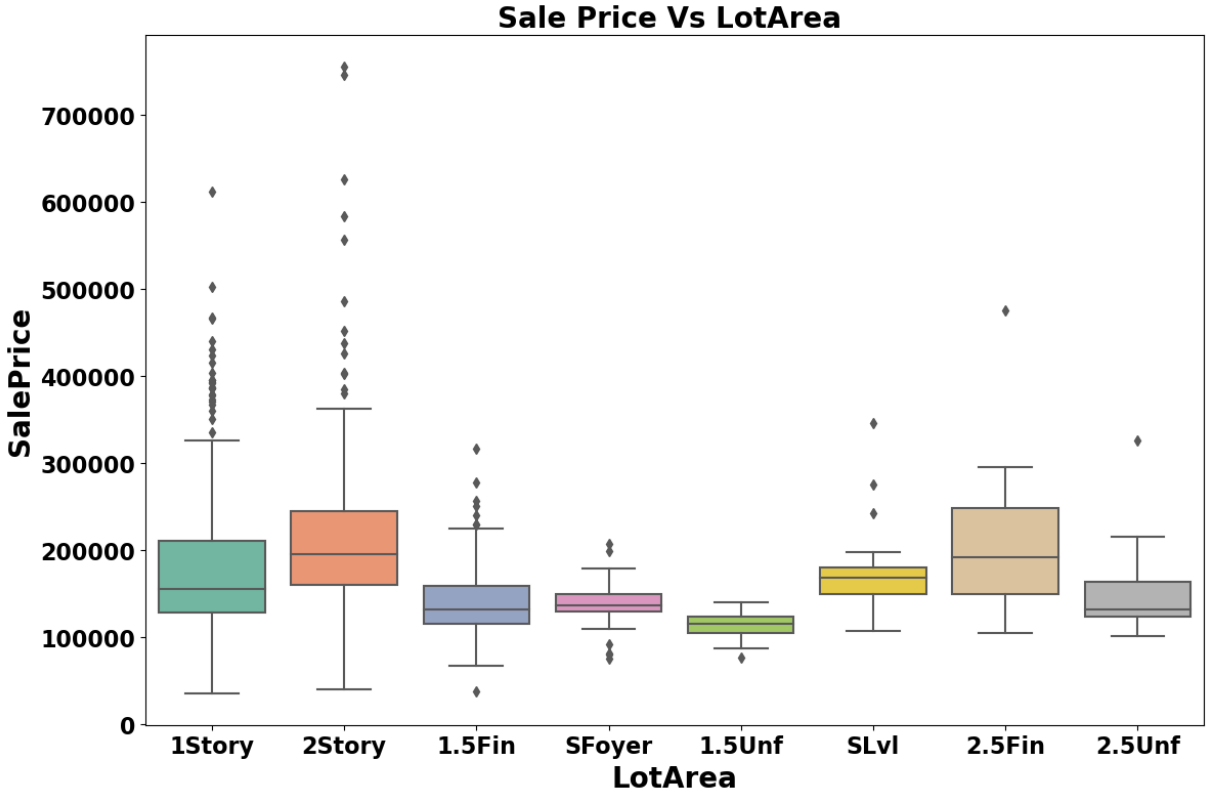
Observation: -

- 44.2% Properties with CBlock Foundation & 43.9% housing property come with PConc Foundation.
- Pconc Foundation are mostly use in costly housing properties.



Observation: -

- As Basement Quality increase in relation to it, sale Price also get increases.



Observation: -

- Two Story Building are costlier than remaining.

CONCLUSION

1. Key Findings and Conclusions of the Study

ALORITHM	R2Score	CV Score
Random Forest Regressor	90.27	82.89
XGB Regressor	86.67	82.05
Linear Regression	87.51	76.62 %
Decision Tree Regressor	61.16	69.60 %
Extra Tree Regressor	89.76	83.26 %
Ridge Regression	87.52	76.66 %
Random Forest Regressor Hyper Parameter Tuned Final Model	89.57	82.05 %

Observation:

- Random Forest Regressor giving us maximum R2 Score, so Random Forest Regressor is selected as best model.
- After hyper parameter tuning Final Model is giving us R2 Score of 89.57% which is close to earlier R2 score of 90.27%.

2. Limitations of this work and Scope for Future Work

- ANN can be used create more accurate model.
- Some additional feature can be added to data which enable us to perform Time series analysis.