AMES HOUSE SALE PRICE PREDICTION

YunTzu, Yu

07/2023

Introduction :

In this project, we'll create a predictive and effective model to assist the potential house buyer make an informed decision and help the developer to determine a competitive house selling price aligning with the market demand.
Since house prices are subject to fluctuation based on the economic condition and various factors related to the house, understanding the factors that could impact the house price becomes crucial. Identifying key factors that could significantly impact the house price can help us to build a robust predictive model to estimate house prices accurately.

The primary goal of this project is two-fold:
House Price Prediction: Develop an accurate predictive model that can estimate house prices based on relevant features. This model will assist potential buyers in evaluating the fair market value of a property and making informed decisions.
Identify Key Factors: Through data analysis and feature importance evaluation, we aim to identify the critical factors that have a substantial impact on house price changes. This knowledge will help both buyers and developers to comprehend the driving forces behind the fluctuations in the real estate market.

Data Description:

For this project, we'll use the Ames housing dataset available on Kaggle. The dataset was compiled by Dean De Cock and covers residential properties in Ames, Iowa, recorded from 2006 to 2010. It contains a total of 81 features, each with 2,920 observations.

Using historical data from 2006 to 2010, the model will be trained to understand the relationship between various features and house prices. Once the model is trained, it could be utilized to predict the house price for the upcoming year, offering meaningful insight to buyers and developers alike.

Exploratory Data Analysis (EDA):

Understanding dataset is the crucial first step for any machine learning project. Since the main goal of my project is to predict the residential house price, it makes sense to filter out non-residential observations from the data. Based on the description of MSZoning, we can identify the residential types and retain only those observations in our dataset.

MSZoning: Identifies the general zoning classification of the sale.

```
A    Agriculture
C    Commercial
FV    Floating Village Residential
I    Industrial
RH    Residential High Density
RL    Residential Low Density
RP    Residential Low Density Park
RM    Residential Medium Density
```

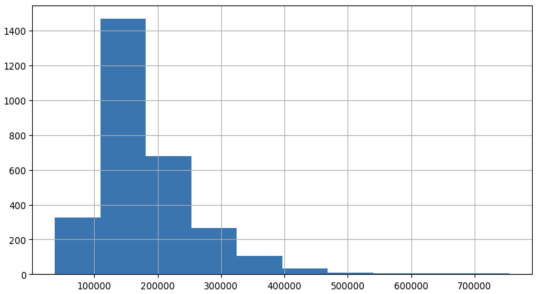
```
train0=train0.apply(lambda row: row[train0['MSZoning'].isin(['RL', 'RM', 'FV', 'RH'])])
test0=test0.apply(lambda row: row[test0['MSZoning'].isin(['RL', 'RM', 'FV', 'RH'])])

train0['MSZoning'].unique()
```

```
array(['RL', 'RM', 'FV', 'RH'], dtype=object)
```

Next step, I examine the descriptive statistics of the target variable.
The range of house prices is from $37900 to $755000 with an average of 181654.94 and a mode of $163945.
So we can know that it is mildly right-skewed, this means that the tail of the distribution extends toward higher house prices, indicating that there might be a higher concentration of houses with higher prices in the dataset.
From the boxplot, we can see there are several outlier points at the higher house price.

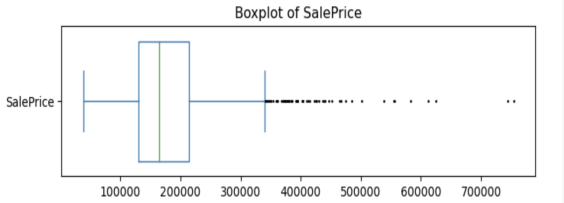
```
import matplotlib.pyplot as plt
plt.figure(figsize=(10,5))
data2['SalePrice'].hist()
```

```
<Axes: >
```




```
data2['SalePrice'].plot.box(figsize = (8,2), widths = 0.8, vert=False,
flierprops = dict(markersize=0.7),title = 'Boxplot of SalePrice')
```

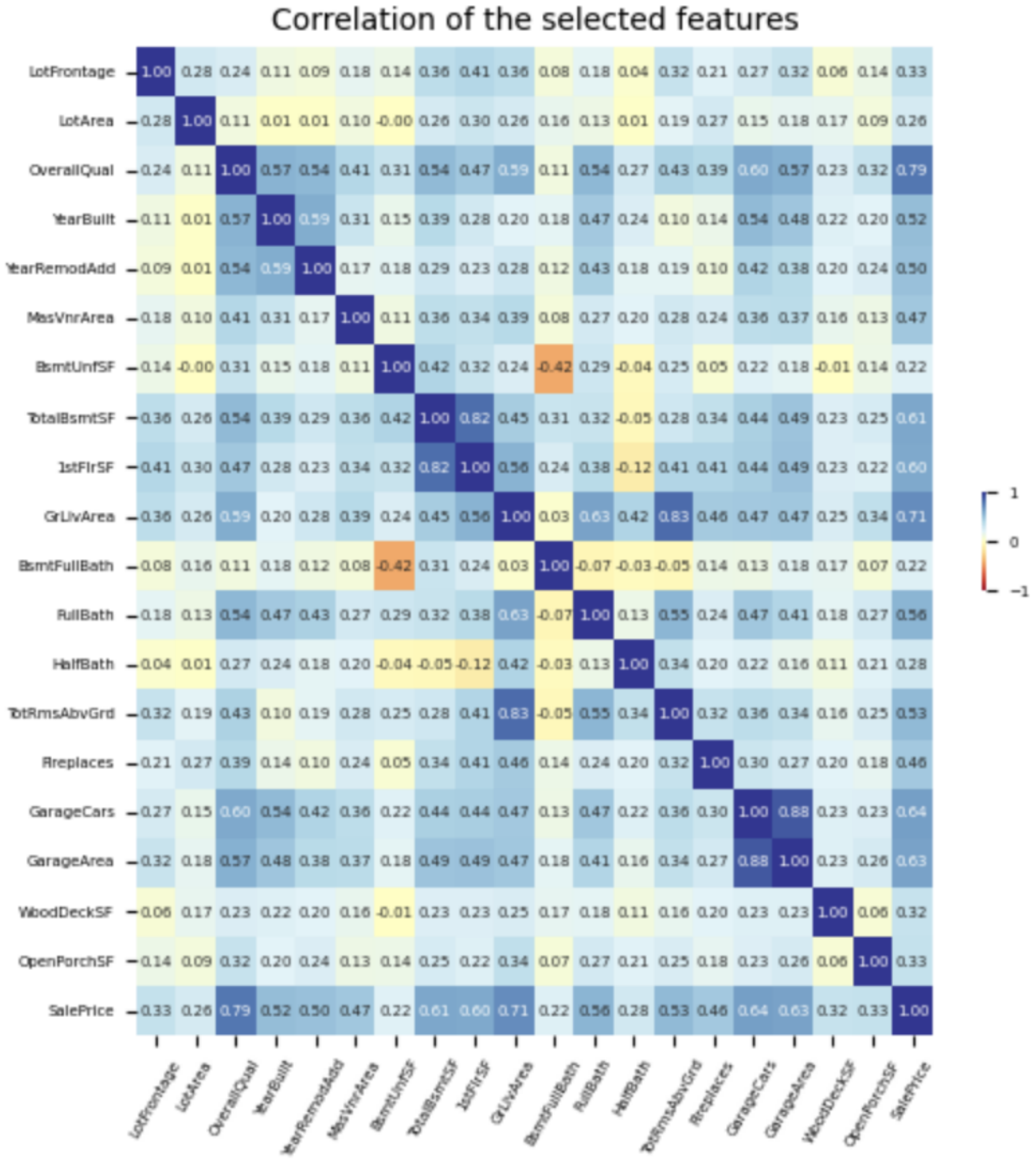
```
<Axes: title={'center': 'Boxplot of SalePrice'}>
```



Cleaning

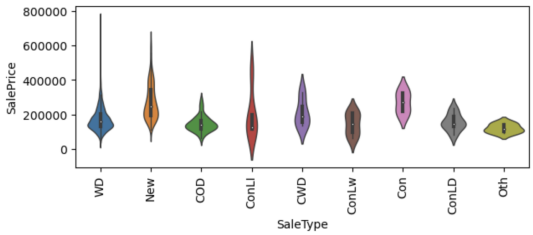
Here I handled the missing value by checking the description of the dataset, and if the missing value in a certain feature has a meaningful interpretation, I replaced the missing value with “None”. Otherwise, if the missing value does not have any specific, I’ll remove the row containing missing values.

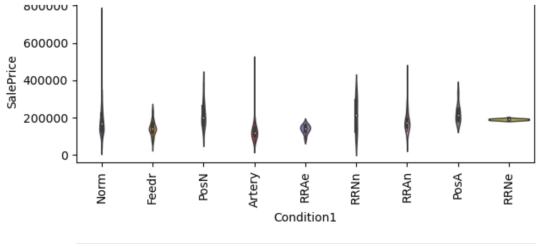
Since I am dealing with a large number of features I’d like to use a heatmap to perform the correlation analysis for numerical features first to identify and remove the features that are lowly correlated with the target variable (house price) and remove features that are highly correlated with each other I to prevent multicollinearity, which can negatively impact the performance of regression models.

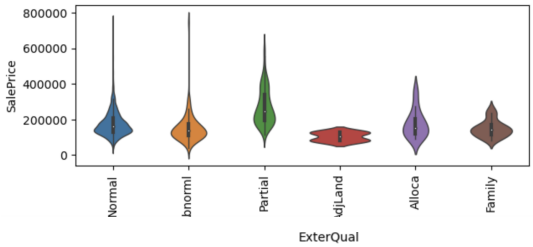


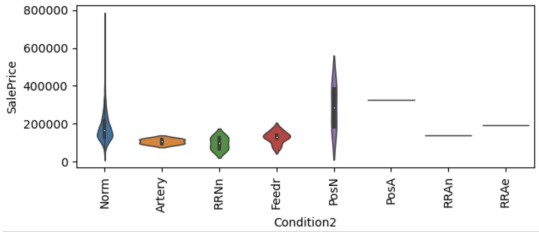
As to categorical feature, we can violin boxplot to filter out those features who has a significant impact on my target value as it changes.

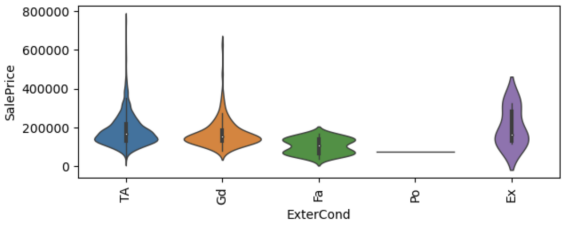
I used violin plots to identify categorical features that exhibit both variation in the sales price and a reasonable distribution across different values of each category.

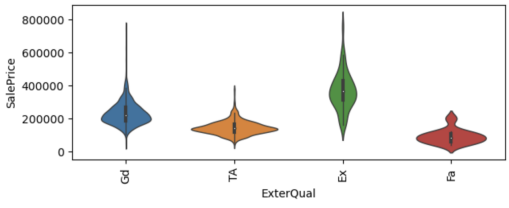

SalePrice
800000
600000
400000
200000
0
WD
New
COD
ConLI
CWD
ConLw
Con
ConLD
Oth
SaleType

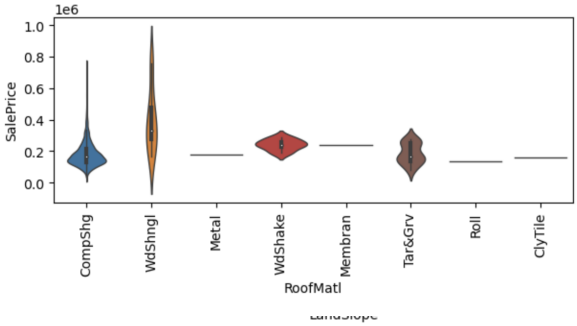

SalePrice
800000
600000
400000
200000
0
Norm
Feedr
PosN
Artery
RRAe
RRNn
RRAn
PosA
RRNe
Condition1

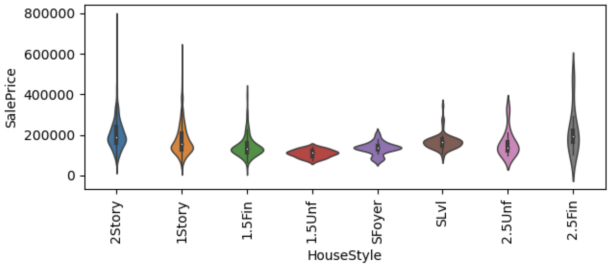

SalePrice
800000
600000
400000
200000
0
Normal
Abnorml
Partial
AdjLand
Alloca
Family
ExterQual

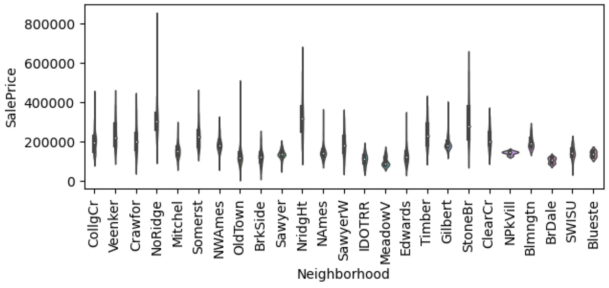

SalePrice
800000
600000
400000
200000
0
Norm
Artery
RRNn
Feedr
PosN
PosA
RRAn
RRAe
Condition2

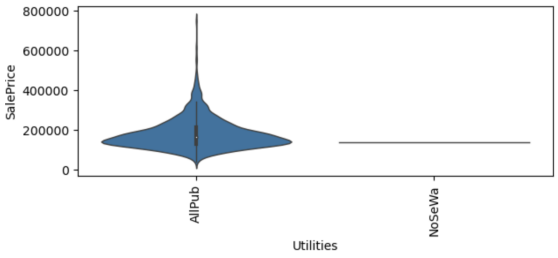

SalePrice
800000
600000
400000
200000
0
TA
Gd
Fa
Po
Ex
ExterCond

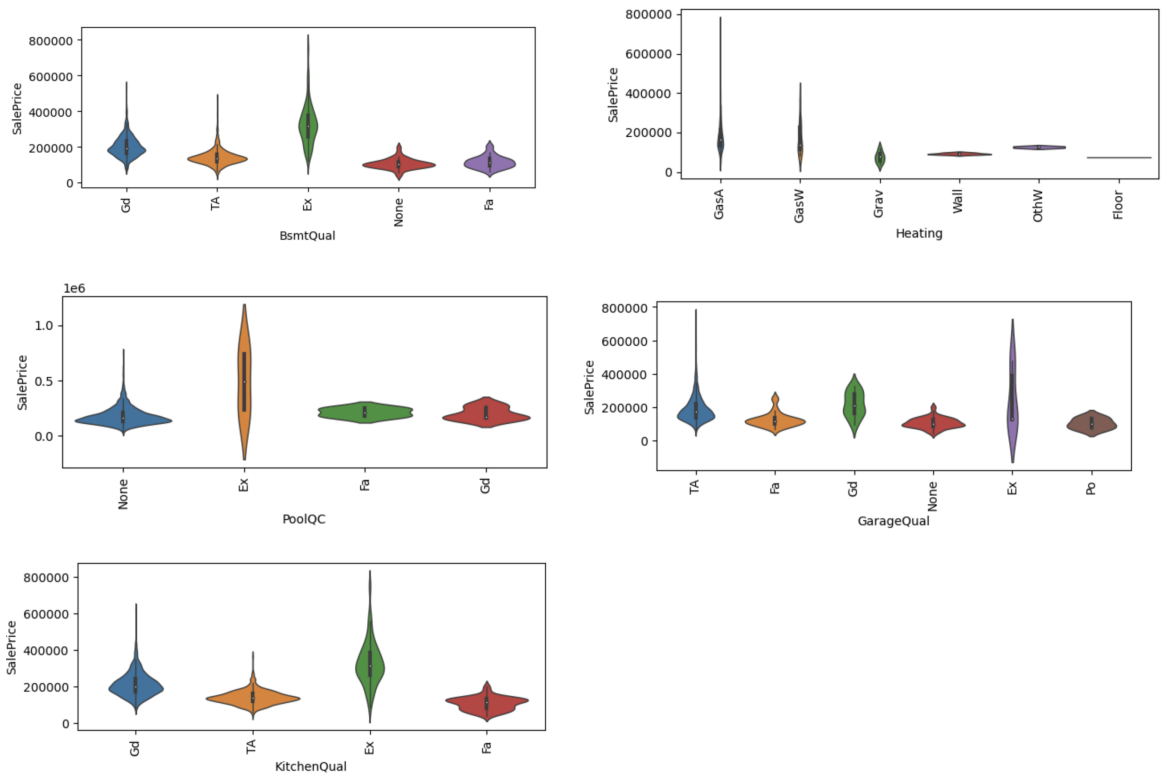

SalePrice
800000
600000
400000
200000
0
Gd
TA
Ex
Fa
ExterQual


1e6
SalePrice
1.0
0.8
0.6
0.4
0.2
0.0
CompShg
WdShngl
Metal
WdShake
Membran
Tar&Grv
Roll
ClyTile
RoofMatl


SalePrice
800000
600000
400000
200000
0
2Story
1Story
1.5Fin
1.5Unf
SFoyer
SLvl
2.5Unf
2.5Fin
HouseStyle


SalePrice
800000
600000
400000
200000
0
CollgCr
Veenker
Crawfor
NoRidge
Mitchel
Somerst
NWAmes
OldTown
BrkSide
Sawyer
NridgHt
NAmes
SawyerW
IDOTRR
MeadowV
Edwards
Timber
Gilbert
StoneBr
ClearCr
NPkVill
Blmngtn
BrDale
SWISU
Blueste
Neighborhood


SalePrice
800000
600000
400000
200000
0
AllPub
NoSeWa
Utilities



Data Preprocessing

We can check the skewness of the numerical data and do the log transformation on the features whose skewness is higher than 1 to make them a normal distribution.

Before			After		
LotFrontage	Skewness: 02.51	Kurtosis: 021.28	LotFrontage	Skewness: -0.64	Kurtosis: 003.13
LotArea	Skewness: 12.17	Kurtosis: 201.73	LotArea	Skewness: -0.14	Kurtosis: 004.67
OverallQual	Skewness: 00.24	Kurtosis: 000.09	OverallQual	Skewness: 00.24	Kurtosis: 000.09
YearBuilt	Skewness: -0.63	Kurtosis: -00.40	YearBuilt	Skewness: -0.63	Kurtosis: -00.40
YearRemodAdd	Skewness: -0.51	Kurtosis: -01.26	YearRemodAdd	Skewness: -0.51	Kurtosis: -01.26
MasVnrArea	Skewness: 02.67	Kurtosis: 010.05	MasVnrArea	Skewness: 00.49	Kurtosis: -01.63
BsmtUnfSF	Skewness: 00.92	Kurtosis: 000.47	BsmtUnfSF	Skewness: 00.92	Kurtosis: 000.47
TotalBsmtSF	Skewness: 01.52	Kurtosis: 013.23	TotalBsmtSF	Skewness: -5.14	Kurtosis: 027.53
1stFlrSF	Skewness: 01.38	Kurtosis: 005.75	1stFlrSF	Skewness: 00.08	Kurtosis: 000.15
GrLivArea	Skewness: 01.37	Kurtosis: 004.92	GrLivArea	Skewness: 00.01	Kurtosis: 000.25
BsmtFullBath	Skewness: 00.59	Kurtosis: -00.85	BsmtFullBath	Skewness: 00.59	Kurtosis: -00.85
FullBath	Skewness: 00.04	Kurtosis: -00.87	FullBath	Skewness: 00.04	Kurtosis: -00.87
HalfBath	Skewness: 00.67	Kurtosis: -01.09	HalfBath	Skewness: 00.67	Kurtosis: -01.09
TotRmsAbvGrd	Skewness: 00.68	Kurtosis: 000.88	TotRmsAbvGrd	Skewness: 00.68	Kurtosis: 000.88
Fireplaces	Skewness: 00.64	Kurtosis: -00.22	Fireplaces	Skewness: 00.64	Kurtosis: -00.22
GarageCars	Skewness: -0.34	Kurtosis: 000.23	GarageCars	Skewness: -0.34	Kurtosis: 000.23
GarageArea	Skewness: 00.16	Kurtosis: 000.87	GarageArea	Skewness: 00.16	Kurtosis: 000.87
WoodDeckSF	Skewness: 01.53	Kurtosis: 002.95	WoodDeckSF	Skewness: 00.15	Kurtosis: -01.90
OpenPorchSF	Skewness: 02.23	Kurtosis: 007.35	OpenPorchSF	Skewness: -0.03	Kurtosis: -01.78
SalePrice	Skewness: 01.91	Kurtosis: 006.61	SalePrice	Skewness: 00.24	Kurtosis: 000.58

Modeling

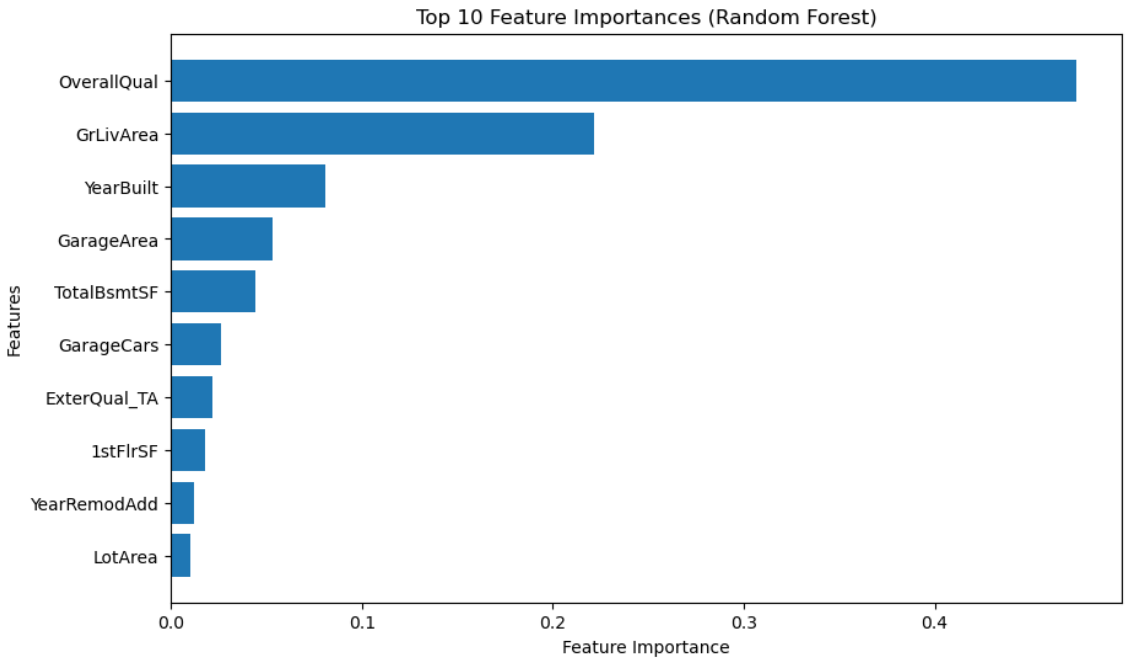
RandomForest

After identifying the numeric and categorical features to include in my model and after imputing all missing values for these features, I dumified categorical features, resulting in a total of 139 features, then I split my dataset into the training dataset and testing dataset. I have my dataset ready to create models to predict the sales price.

I initiated the training and evaluation of a Random Forest model with the specified hyperparameters as follows indicating how the random forest model will be constructed during training.

```
RandomForestRegressor
RandomForestRegressor(max_depth=10, max_features='auto', max_leaf_nodes=10,
                      max_samples=150, min_samples_leaf=15,
                      min_samples_split=20, random_state=50)
```

By identifying the top 10 most important features from your Random Forest model we can understand the key factors that influence house prices. By focusing on these features, we can make more informed decisions when it comes to evaluating and predicting house prices.



R-squared on Training and Validation Datasets:

- The R-squared value of 0.782 on the training dataset indicates that the predictor variables I've chosen explain approximately 78.2% of the variance in the target variable within the training data.
- The R-squared value of 0.773 on the validation dataset (test dataset) suggests that the model explains around 77.3% of the variance in the target variable within new, unseen data.

Model Performance Assessment:

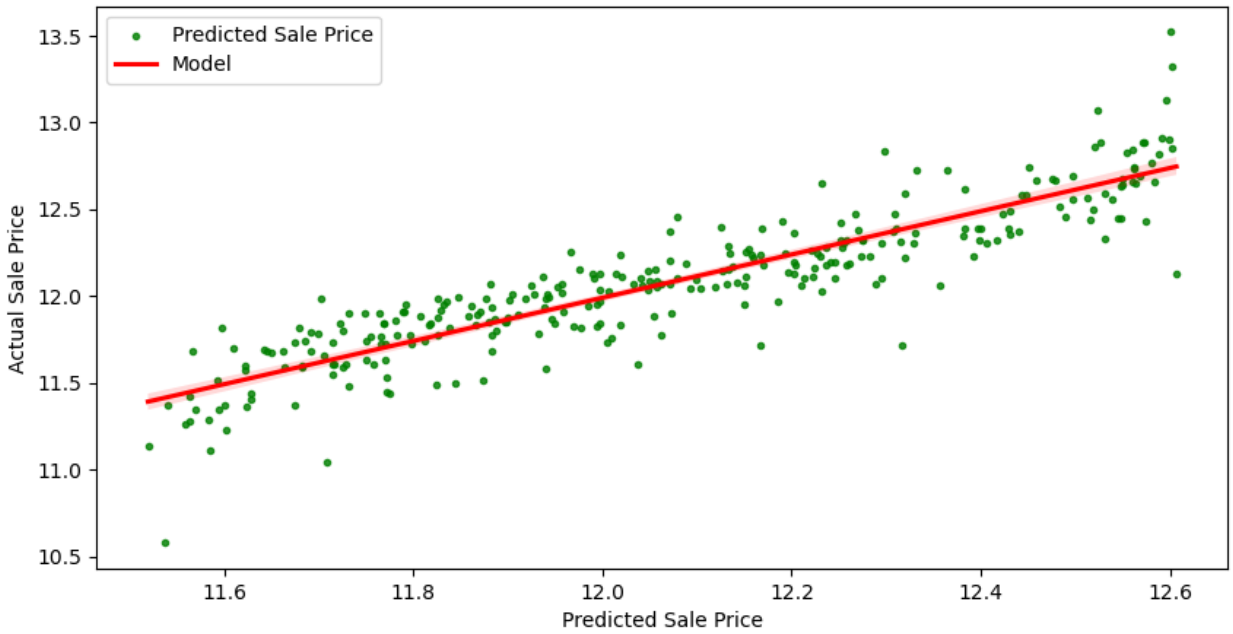
- The R-squared value of 0.773 is fairly high, indicating that the model performs well in explaining the variability in the target variable.
- The small difference of 0.009 (0.782 - 0.773) between the R-squared values on the training and validation datasets highlights that the model is not overfitting. This close proximity validates the model's ability to generalize effectively from training to validation data.

```
#What is the  R2  on training data
y_predict = estimator.predict(X_test)
y_predict_val = estimator.predict(X_val)
rscore1=r2_score(y_test,y_predict)
print("the model  R2 on training data is {:8.3f}".format(rscore1))
#What is the  R2  on validation data?
rscore2=r2_score(y_val,y_predict_val)
print("the model  R2  on validation data is {:5.3f}".format(rscore2))
```

```
the model  R2 on training data is    0.782
the model  R2  on validation data is 0.773
```

The next step, we can create a regression plot to compare predicted sale prices with actual sale prices using Seaborn to visually assess the performance of your predictive model.

We can see the presence of few outliers on both sides suggests that while the model is generally performing well, there are instances where it struggles to accurately predict sale prices.

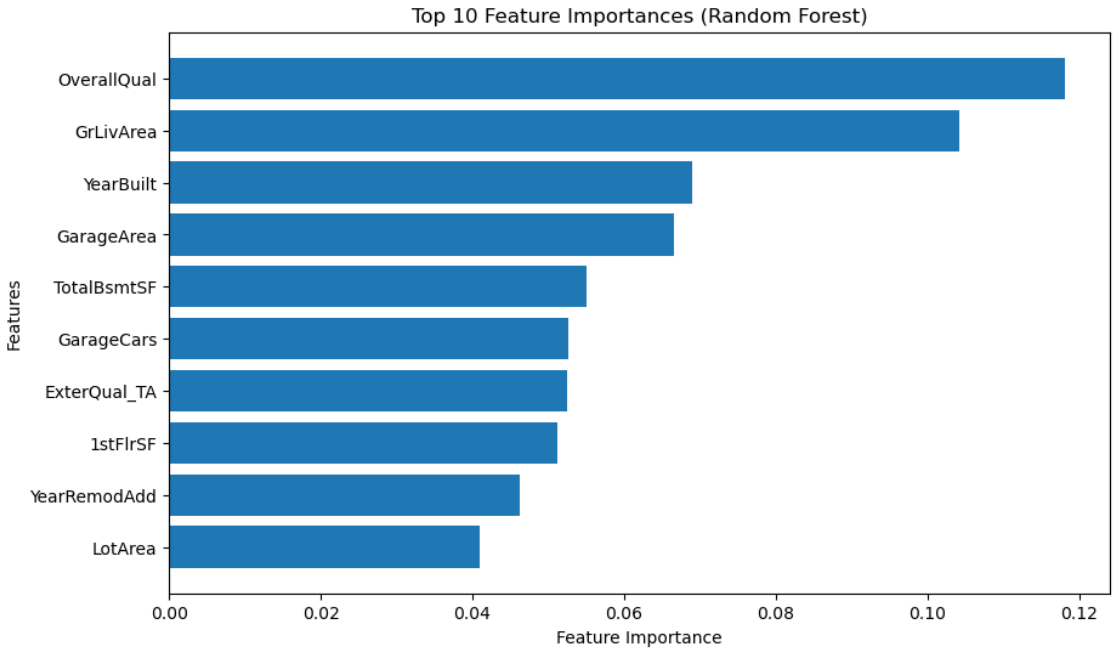


Gradient Boosting Regressor

To improve the predictive model here I transition the model from Random Forest to Gradient Boosting Regressor which is another powerful ensemble technique that can often provide better predictive performance than individual decision trees.

```
▾                          GradientBoostingRegressor
GradientBoostingRegressor(learning_rate=0.001, loss='huber', max_depth=4,
                          max_features='sqrt', min_samples_leaf=15,
                          min_samples_split=10, n_estimators=10000,
                          random_state=42)
```

Here we identifyed the top 10 most important features from your Gradient Boosting Regressor ,it appears that the top 10 most important features identified by both the Gradient Boosting Regressor and the Random Forest models are quite similar. This consistency suggests that these features have a significant impact on predicting house prices and are robust across different modeling techniques.



Here we got 0.920 R-squared score on training dataset and 0.886 R-squared score on validation dataset.

```
#What is the  R2  on training datadata?
y_predict_gbr = estimator2.predict(X_test)
y_predict_val_gbr = estimator2.predict(X_val)
rscore1_gbr=r2_score(y_test,y_predict_gbr)
print("the model  R2  on training data is {}".format(rscore1_gbr))
#What is the  R2  on validation data?
rscore2_gbr=r2_score(y_val,y_predict_val_gbr)
print("the model  R2  on validation data is {}".format(rscore2_gbr))
```

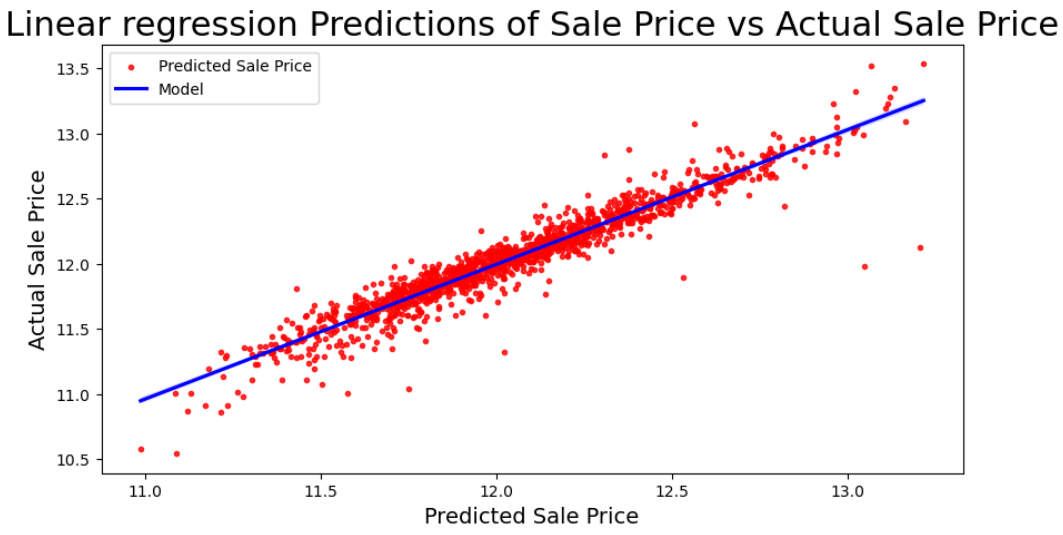
```
the model  R2  on training data is 0.9200409059410134
the model  R2  on validation data is 0.8861527652569887
```

An increase in the R-squared values on both the training and testing datasets when applying a Gradient Boosting Regressor indicates that the model is capturing more variance and performing better than before and a slightly higher difference of 0.033 (0.920 - 0.886) in R-squared values is still acceptable.

By checking the plot to compare predicted sale prices with actual sale prices in your Gradient Boosting Regressor model , it shows a closer alignment between the predicted values and the actual values compared to the plot from the Random Forest model.

This suggests that the Gradient Boosting Regressor is performing better in terms of predicting house prices accurately.

The closer alignment between predicted and actual values indicates that the Gradient Boosting Regressor model is capturing the underlying patterns and relationships in the data more effectively. This improvement in predictive accuracy can be attributed to the inherent strengths of the Gradient Boosting algorithm, which leverages an ensemble of weak learners to create a strong predictive model.



Conclusion

This project showcases the development and assessment of predictive models for house price estimation. By identifying key influential factors and enhancing model performance, our endeavor aids potential buyers and developers in making informed and strategic decisions in the dynamic real estate market.