# House Prices: Advanced Regression Techniques

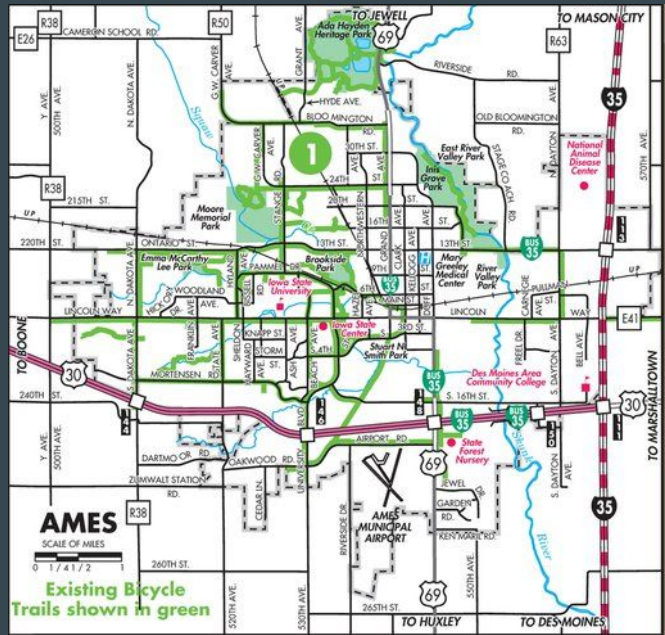
Grannel Pinto
November, 2019

# Introduction

The following dataset, Ames Housing dataset was compiled by Dean De Cock, it is a modernized and expanded version of the Boston Housing Dataset. Buyers in the real-estate industry can use this tool to find the features they are looking for in a house and match it with a price. Sellers in the real estate industry can use this to determine the cost of their home and identify which features have a bigger impact on the sale price.

# Objective

The objective of this Capstone project is to predict the sale price of each house in correlation to its features by using different machine learning algorithms to find the lowest root mean squared logarithmic error.



# Tasks Performed

## Data analysis

- Conceptualize the data
- Remove unnecessary variables
- Deal with missing data
- Handle Outliers

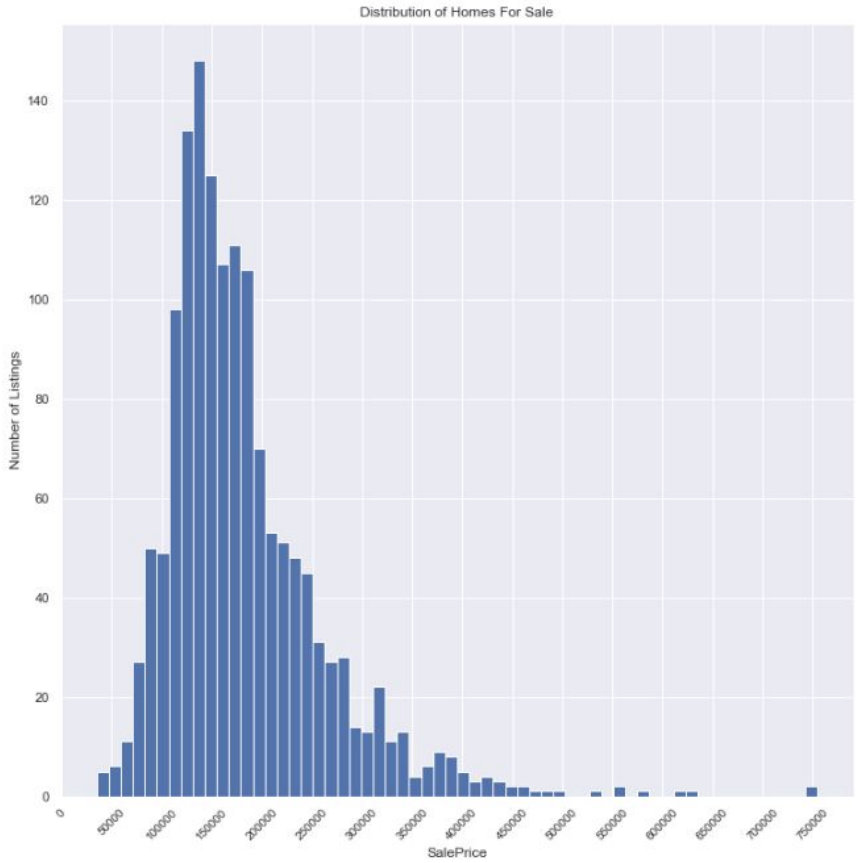
## Inferential Statistic

- Will help to observe between homes that have 2 bedrooms above grade versus 3 bedrooms above grade.

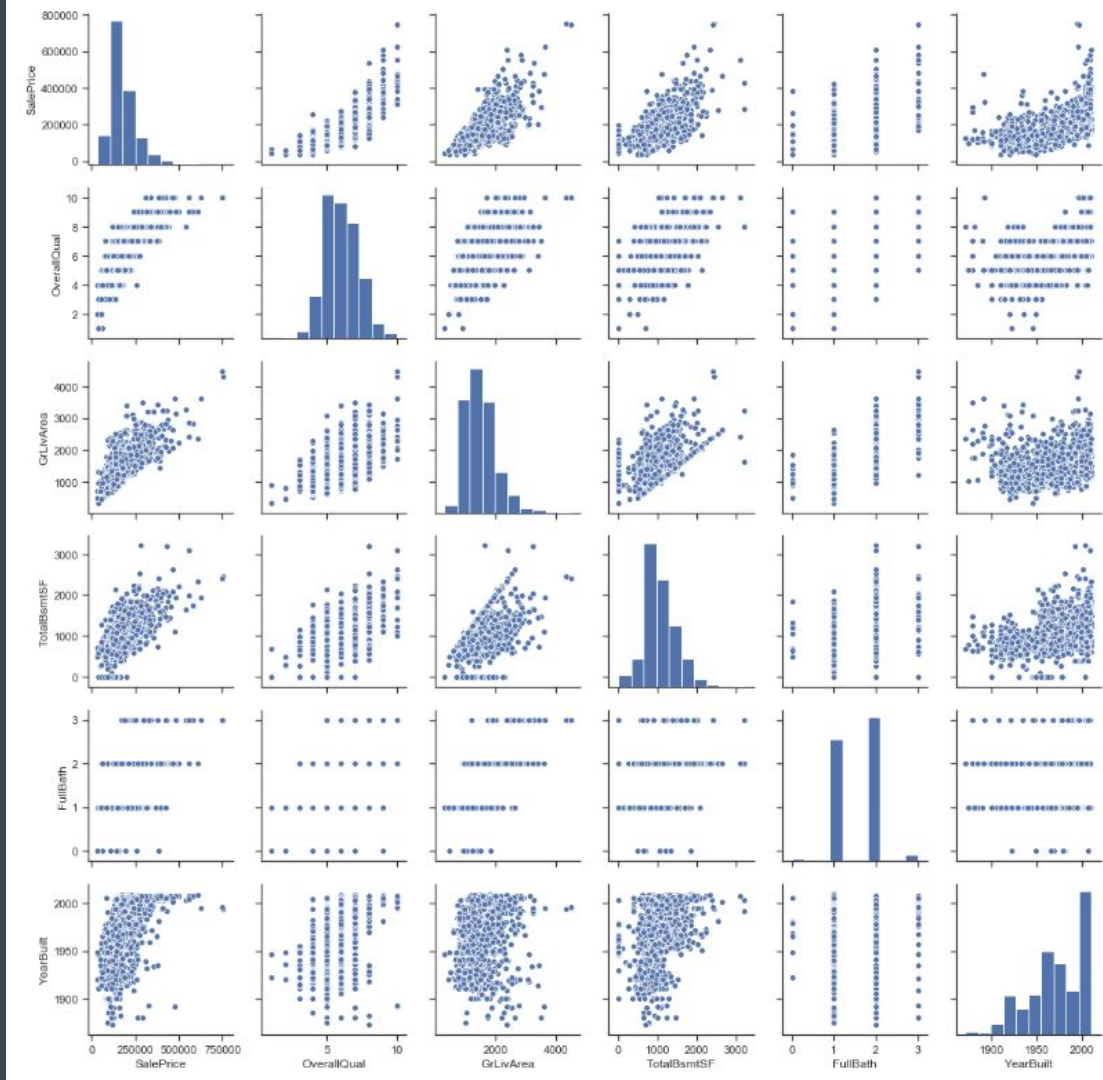
## Machine Learning & Model Tuning

- Linear regression, random forests, decision tree regressor, extra trees regressor and gradient boosting regressor will be used to determine the training accuracy, root mean squared error (RMSE), root mean squared logarithmic error (RMSLE) and R-squared.
- Hyperparameter Tuning will yield a lower error

# Data Analysis

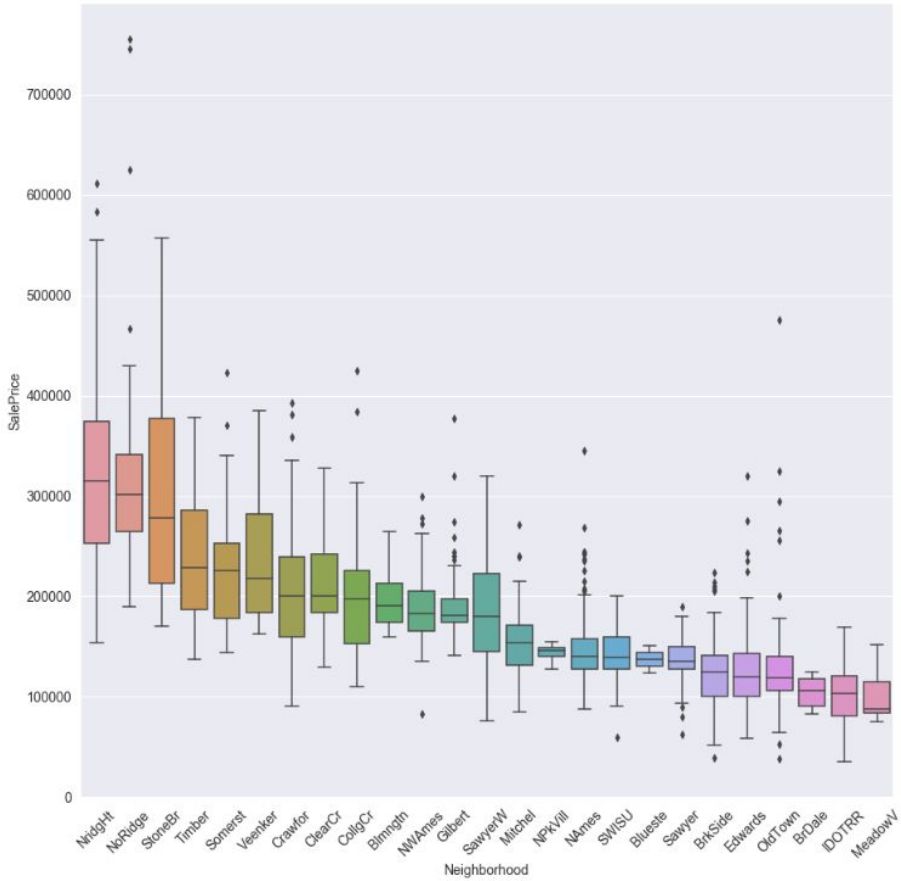


# Correlation Between the House Price and its Features



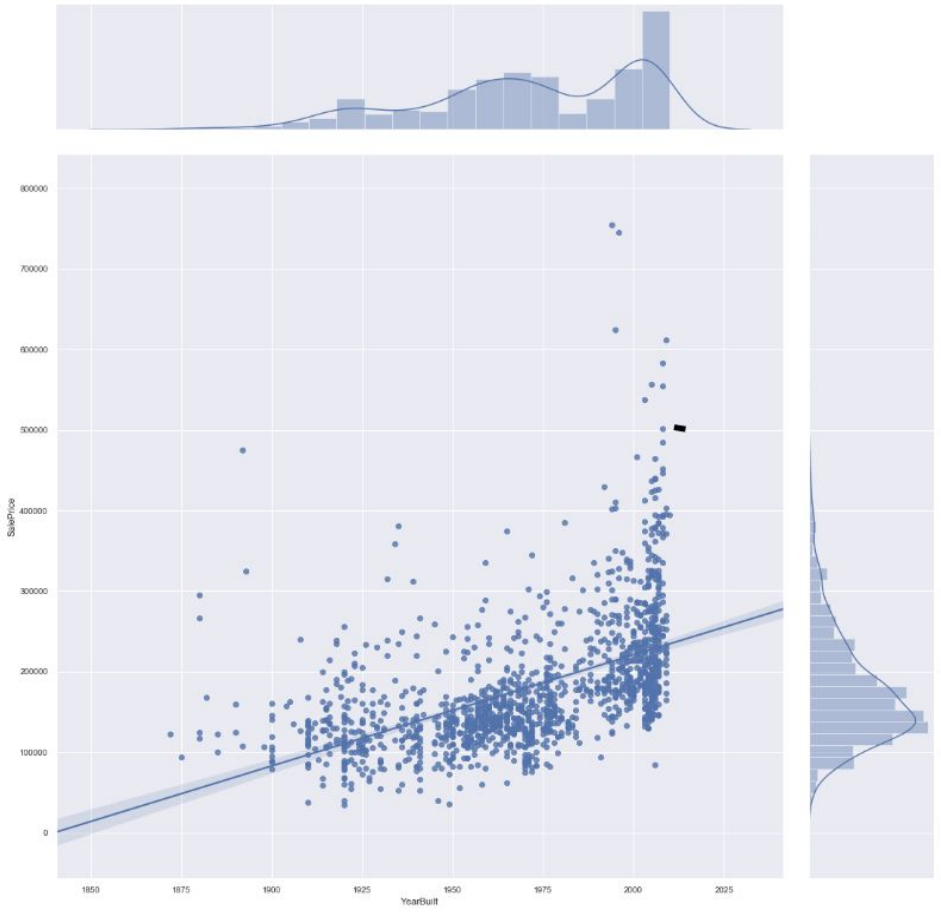
# Average Sales Prices for Each of the Neighborhoods

The neighborhood NridgHt has the highest average sales price. That said, the neighborhood NoRidge has three of the most expensive homes.

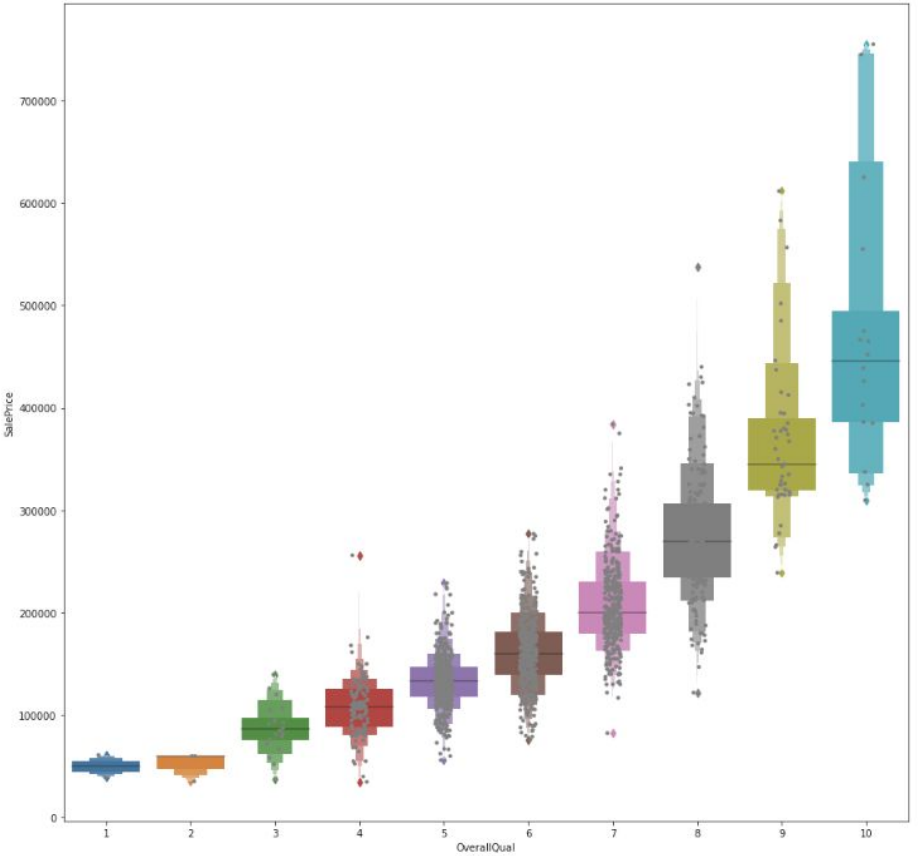


# House Price Vs. Year Built

The joint plot shows as increasing linear regression. Prices are higher for new homes.



# House Price Vs. Overall Quality



# Statistical Data Analysis

```
In [44]: train_df[['SalePrice', 'BedroomAbvGr']].describe()
```

Out[44]:

| | SalePrice | BedroomAbvGr |
|---|---|---|
| count | 1458.000000 | 1458.000000 |
| mean | 180932.919067 | 2.866255 |
| std | 79495.055285 | 0.816323 |
| min | 34900.000000 | 0.000000 |
| 25% | 129925.000000 | 2.000000 |
| 50% | 163000.000000 | 3.000000 |
| 75% | 214000.000000 | 3.000000 |
| max | 755000.000000 | 8.000000 |

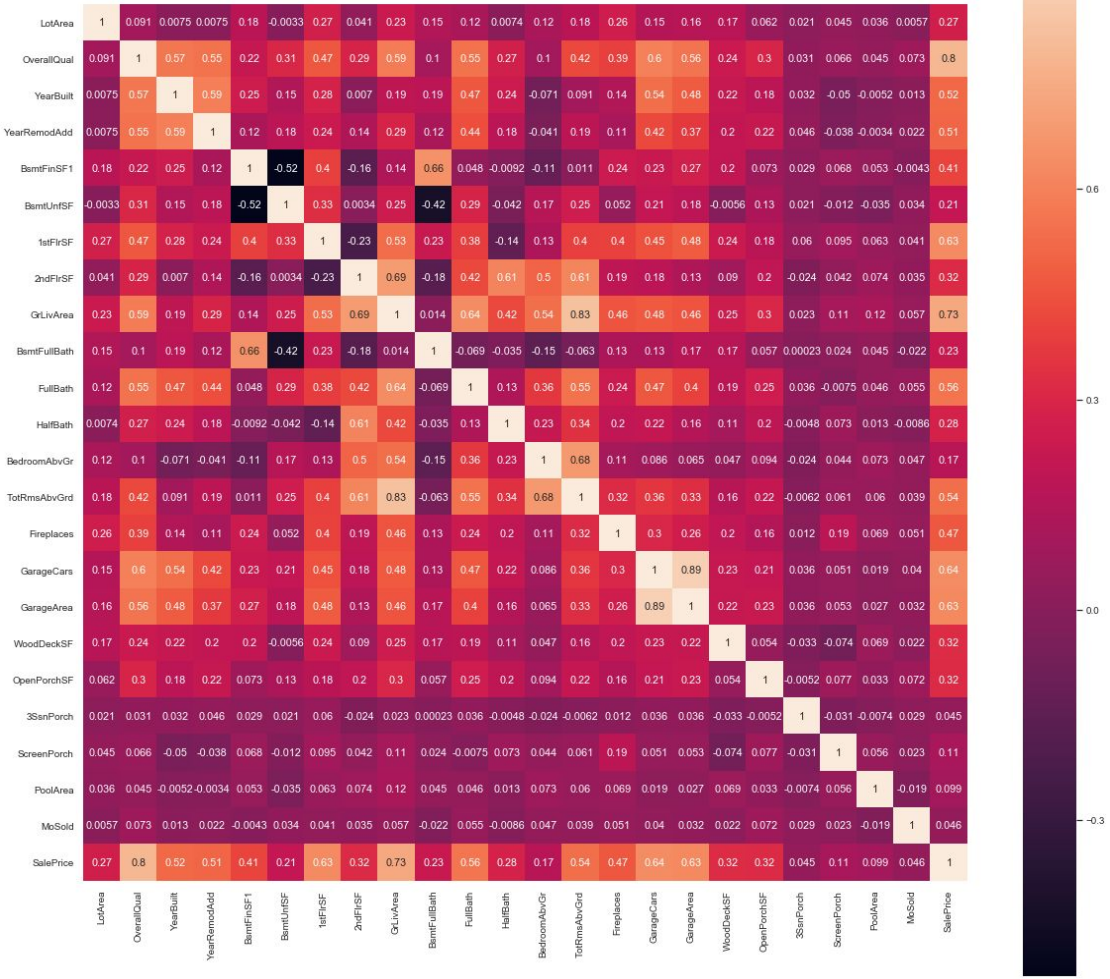
```
In [35]: two_bedroom = train_df.SalePrice.loc[train_df.BedroomAbvGr == 2]
         three_bedroom = train_df.SalePrice.loc[train_df.BedroomAbvGr == 3]

In [36]: import scipy
         scipy.stats.ttest_ind(two_bedroom, three_bedroom)

Out[36]: Ttest_indResult(statistic=-5.246983309011471, pvalue=1.8374907162225815e-07)
```

The above shows code for a t-test. The t-test observes homes with 2 bedrooms and 3 bedrooms above grade. Since the p-value is less than 0.05, we reject the null hypothesis. This is because the average sales price for a 2 bedroom house will be lower than that of a 3 bedroom house.

# Correlation Matrix

| | |
|---|---|
| SalePrice | 1.000000 |
| OverallQual | 0.790982 |
| GrLivArea | 0.708624 |
| GarageCars | 0.640409 |
| GarageArea | 0.623431 |
| TotalBsmtSF | 0.613581 |
| 1stFlrSF | 0.605852 |
| FullBath | 0.560664 |
| TotRmsAbvGrd | 0.533723 |
| YearBuilt | 0.522897 |
| YearRemodAdd | 0.507101 |
| MasVnrArea | 0.475241 |
| GarageYrBlt | 0.470177 |
| Fireplaces | 0.466929 |
| BsmtFinSF1 | 0.386420 |
| LotFrontage | 0.334901 |
| WoodDeckSF | 0.324413 |
| 2ndFlrSF | 0.319334 |
| OpenPorchSF | 0.315856 |
| HalfBath | 0.284108 |
| LotArea | 0.263843 |
| BsmtFullBath | 0.227122 |
| BsmtUnfSF | 0.214479 |
| BedroomAbvGr | 0.168213 |
| ScreenPorch | 0.111447 |
| PoolArea | 0.092404 |
| MoSold | 0.046432 |
| 3SsnPorch | 0.044584 |

Name: SalePrice, dtype: float64



# Machine Learning

The purpose of this project is to explore advanced regression techniques, so I will be using linear regression, random forests, decision tree regressor, extra trees regressor and gradient boosting regressor models to evaluate which of the following gives the best RMSLE results.

# Machine Learning Models

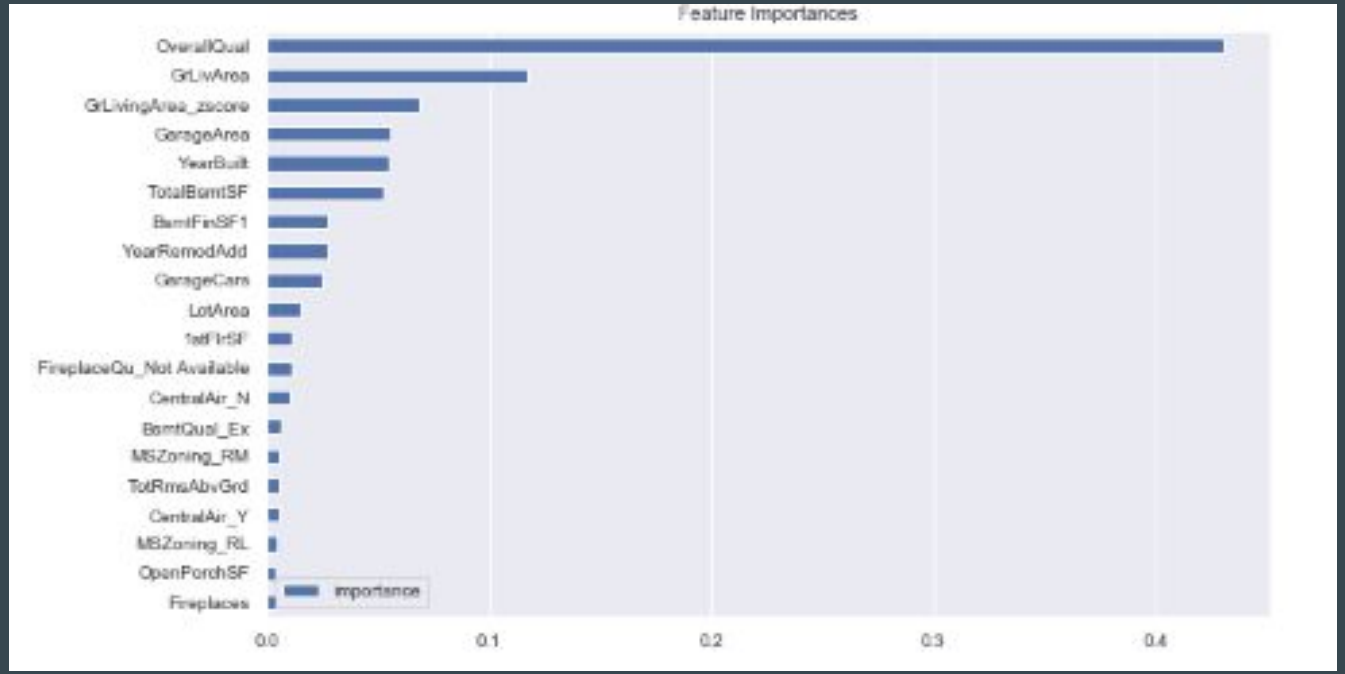
Gradient Boosting Regressor gave the best results for RMSE and RMSLE. This is the model that will be tuned using hyperparameter tuning to get better results.

| Linear Regression Model |
| --- |
| Training accuracy: 0.947952228094347<br>Root-mean-squared error: 0.1380591560566311<br>Root-mean-squared-log-error: 0.010839457597885448<br>Mean-squared-error: 0.019060330571069223<br>R-squared: 0.8869350967152927 |
| **Random Forests Model** |
| Training accuracy: 0.947952228094347<br>Root-mean-squared error: 0.15740947759552357<br>Root-mean-squared-log-error: 0.01240498717784635<br>Mean-squared-error: 0.02477774363689564<br>R-squared: 0.8530196956724805 |
| **Decision Tree Regressor Model** |
| Training accuracy: 0.947952228094347<br>Root-mean-squared error: 0.2073968217128347<br>Root-mean-squared-log-error 0.016137377328512947<br>Mean-squared-error: 0.043013441656585334<br>R-squared: 0.7448464703846212 |
| **Extra Trees Regressor Model** |
| Training accuracy: 0.947952228094347<br>Root-mean-squared error: 0.15150073877132939<br>Root-mean-squared-log-error 0.011961564354778064<br>Mean-squared-error: 0.02295247384825859<br>R-squared: 0.863847102434984 |
| **Gradient Boosting Regressor Model** |
| Training accuracy: 0.947952228094347<br>Root-mean-squared error: 0.12735835790425343<br>Root-mean-squared-log-error 0.010032782763355334<br>Mean-squared-error: 0.016220151328067912<br>R-squared: 0.903782894303263 |

# Hyperparameter Tuning

| Before Hyperparameter Tuning Gradient Boosting Regressor | After Hyperparameter Tuning Gradient Boosting Regressor |
|---|---|
| Training accuracy: 0.947952228094347<br>Root-mean-squared error: 0.12735835790425343<br>Root-mean-squared-log-error 0.010032782763355334<br>Mean-squared-error: 0.016220151328067912<br>R-squared: 0.903782894303263 | Training accuracy: 0.947952228094347<br>Root-mean-squared error: 0.11924170052390301<br>Root-mean-squared-log-error: 0.009355913564395696<br>Mean-squared-error: 0.014218583143832172<br>R-squared: 0.9156560941055718 |

# Feature Importances for Gradient Boosting Regressor



# Conclusion

From the following machine learning models used in this project, the gradient boosting regressor gave the best result. After hyperparameter tuning this model the RMSLE value retreived was 0.00936.