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

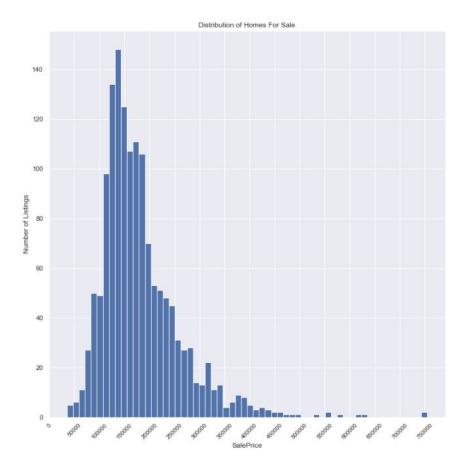
Inferential Statistic

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

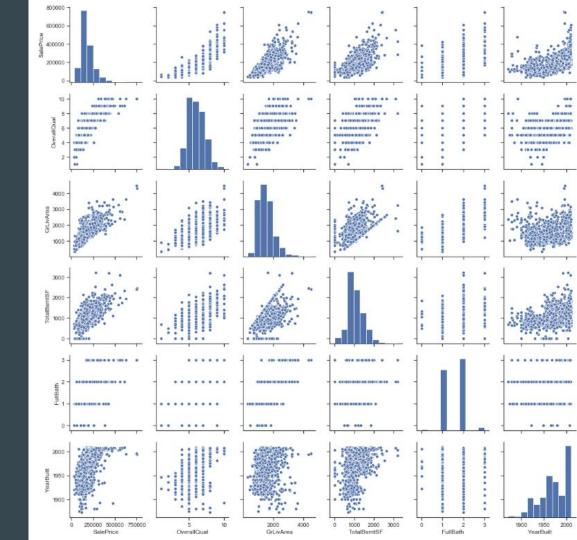
Machine Learning & Model Tuning

- 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

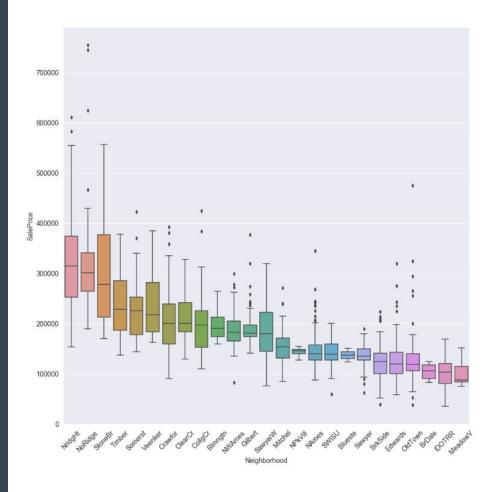


Correlation Between the House Price and its Features



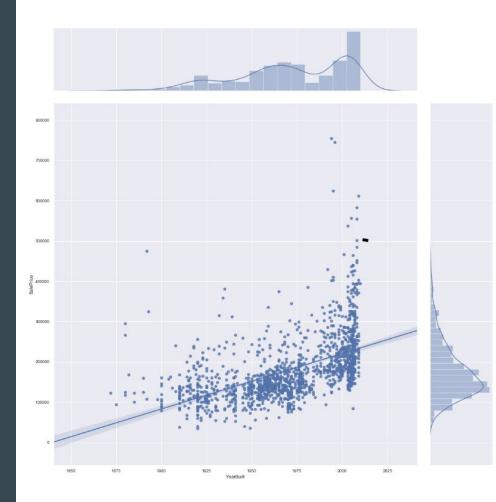
Average Sales Prices for Each of the Neighborhoods

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

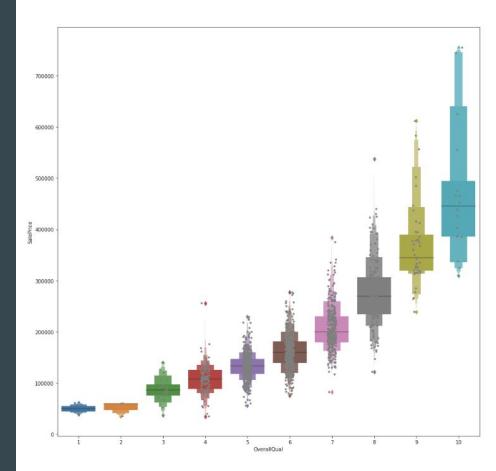


House Price Vs. Year Built

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
                   1458.000000
                                 1458.000000
                                   2.866255
                  180932 919067
                   79495 055285
                                   0.816323
                  34900.000000
                                   0.000000
                 129925.000000
                                   2.000000
                  63000.000000
                                   3.000000
                 214000.000000
                                   3.000000
                 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)
           scipy.stats.ttest ind(two bedroom, three bedroom)
          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
lstFlrSF	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
2ndF1rSF	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: floa

LotArea	1		0.0075	0.0075		-0.0033		0.041				0.0074							0.062		0.045	0.036	0.0057	
OverallQual	0.091	1	0.57		0.22					0.1								0.24			0.066	0.045	0.073	0.8
YearBuilt	0.0075	0.57	1		0.25			0.007	0.19									0.22			-0.05	-0.0052	0.013	0.52
YearRemodAdd	0.0075		0.59	1	0.12								-0.041							0.046	-0.038	-0.0034	0.022	
BsmtFinSF1	0.18	0.22	0.25	0.12	1	-0.52		-0.16		0.66	0.048	-0.0092								0.029	0.068		-0.0043	
BamtUnfSF	-0.0033				-0.52	1		0.0034		-0.42	0.29	-0.042						-0.0056					0.034	
1stFlrSF	0.27				0.4	0.33	1	-0.23	0.53	0.23	0.38	-0.14					0.48			0.06	0.095	0.063	0.041	0.63
2ndFlrSF	0.041		0.007		-0.16	0.0034	-0.23	1	0.69	-0.18		0.61	0.5	0.61			0.13			-0.024	0.042			0.32
GrLivArea	0.23	0.59	0.19				0.53	0.69	1	0.014	0.64			0.83		0.48	0.46						0.057	0.73
BsmtFullBath	0.15	0.1	0.19		0.66	-0.42	0.23	-0.18	0.014	1	-0.069	-0.035	-0.15	-0.063	0.13	0.13	0.17			0.00023	0.024	0.045	-0.022	0.23
FullBath	0.12	0.55	0.47		0.048	0.29		0.42	0.64	-0.069	1	0.13		0.55		0.47				0.036	-0.0075	0.046	0.055	0.56
HalfBath	0.0074	0.27	0.24	0.18	-0.0092	-0.042	-0.14	0.61	0.42	-0.035	0.13	1					0.16			-0.0048	0.073	0.013	-0.0086	0.28
BedroomAbvGr	0.12		-0.071	-0.041	-0.11	0.17	0.13	0.5		-0.15		0.23	1	0.68	0.11	0.086	0.065	0.047	0.094	-0.024	0.044	0.073	0.047	
TotRmsAbvGrd	0.18		0.091	0.19	0.011	0.25		0.61	0.83	-0.063	0.55		0.68	1	0.32			0.16	0.22	-0.0062	0.061	0.06	0.039	0.54
Fireplaces	0.26		0.14	0.11	0.24	0.052		0.19	0.46	0.13	0.24		0.11	0.32	1		0.26		0.16	0.012	0.19	0.069	0.051	
GarageCars	0.15		0.54	0.42	0.23			0.18		0.13		0.22	0.086	0.36	0.3	1	0.89	0.23			0.051	0.019	0.04	
	0.16					0.18		0.13		0.17		0.16	0.065		0.26	0.89	1	0.22				0.017		
GarageArea	0.10	0.24	0.40			-0.0056		0.09		0.17	0.19	0.10	0.047	0.16		0.03	0.22	1		-0.033				0.32
WoodDeckSF																								
OpenPorchSF	0.062		0.18	0.22	0.073	0.13	0.18			0.057	0.25		0.094	0.22	0.16	0.21	0.23	0.054	1	-0.0052				0.32
3SsnPorch			0.032			0.021		-0.024		0.00023								-0.033		NO MARKO	-0.031		0.029	
ScreenPorch	0.045	0.066			0.068		0.095	0.042	0.11		-0.0075		0.044	0.061	0.19	0.051			0.077	-	1	0.056		0.11
PoolArea	0.036	0.045	-0.0052	-0.0034	0.053	-0.035	0.063	0.074	0.12	0.045	0.046	0.013	0.073	0.06	0.069	0.019	0.027	0.069	0.033	-0.0074	0.056	1	-0.019	0.099
MoSold	0.0057	0.073	0.013	0.022	-0.0043	0.034	0.041		0.057	-0.022	0.055	-0.0086	0.047	0.039	0.051	0.04	0.032	0.022		0.029			1	0.046
SalePrice	0.27	0.8	0.52	0.51	0.41	0.21	0.63	0.32	0.73	0.23	0.56	0.28	0.17	0.54	0.47	0.64	0.63	0.32	0.32	0.045	0.11	0.099	0.046	1
	LotArea	VerallQual	YearBuilt	RemodAdd	smtFinSF1	SemtUnfSF	1stFIrSF	2ndFlrSF	GrLivArea	mtFullBath	FullBath	HalfBath	oomAbvGr	msAbvGrd	Freplaces	arageCars	arageArea	odDeckSF	an Porch SF	SSsnPorch	reenPorch	PoolArea	MoSold	SalePrice

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

Root-mean-squared error: 0.1380591560566311 Root-mean-squared-log-error: 0.010839457597885448 Mean-squared-error: 0.019060330571069223 R-squared: 0.8869350967152927

Training accuracy: 0.947952228094347

Random Forests Model

Training accuracy: 0.947952228094347
Root-mean-squared error: 0.15740947759552357
Root-mean-squared-log-error: 0.01240498717784635
Mean-squared-error: 0.02477774363689564
R-squared: 0.8530196956724805

Decision Tree Regressor Model

Training accuracy: 0.947952228094347
Root-mean-squared error: 0.2073968217128347
Root-mean-squared-log-error 0.016137377328512947
Mean-squared-error: 0.043013441656585334
R-squared: 0.7448464703846212

Extra Trees Regressor Model

Training accuracy: 0.947952228094347 Root-mean-squared error: 0.15150073877132939 Root-mean-squared-log-error 0.011961564354778064 Mean-squared-error: 0.02295247384825859

R-squared: 0.863847102434984

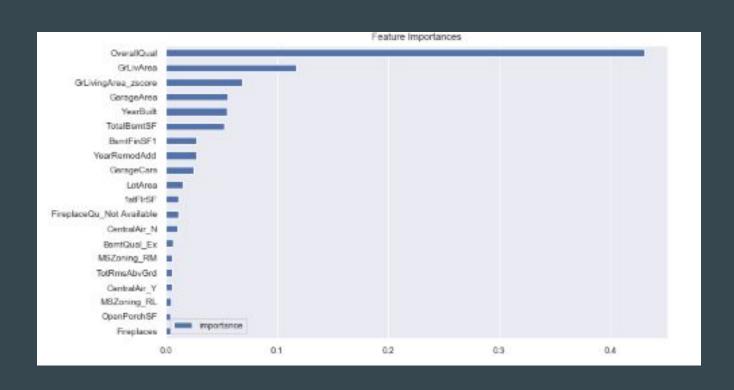
Gradient Boosting Regressor Model

Training accuracy: 0.947952228094347 Root-mean-squared error: 0.12735835790425343 Root-mean-squared-log-error 0.010032782763355334 Mean-squared-error: 0.016220151328067912 R-squared: 0.903782894303263

Hyperparameter Tuning

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