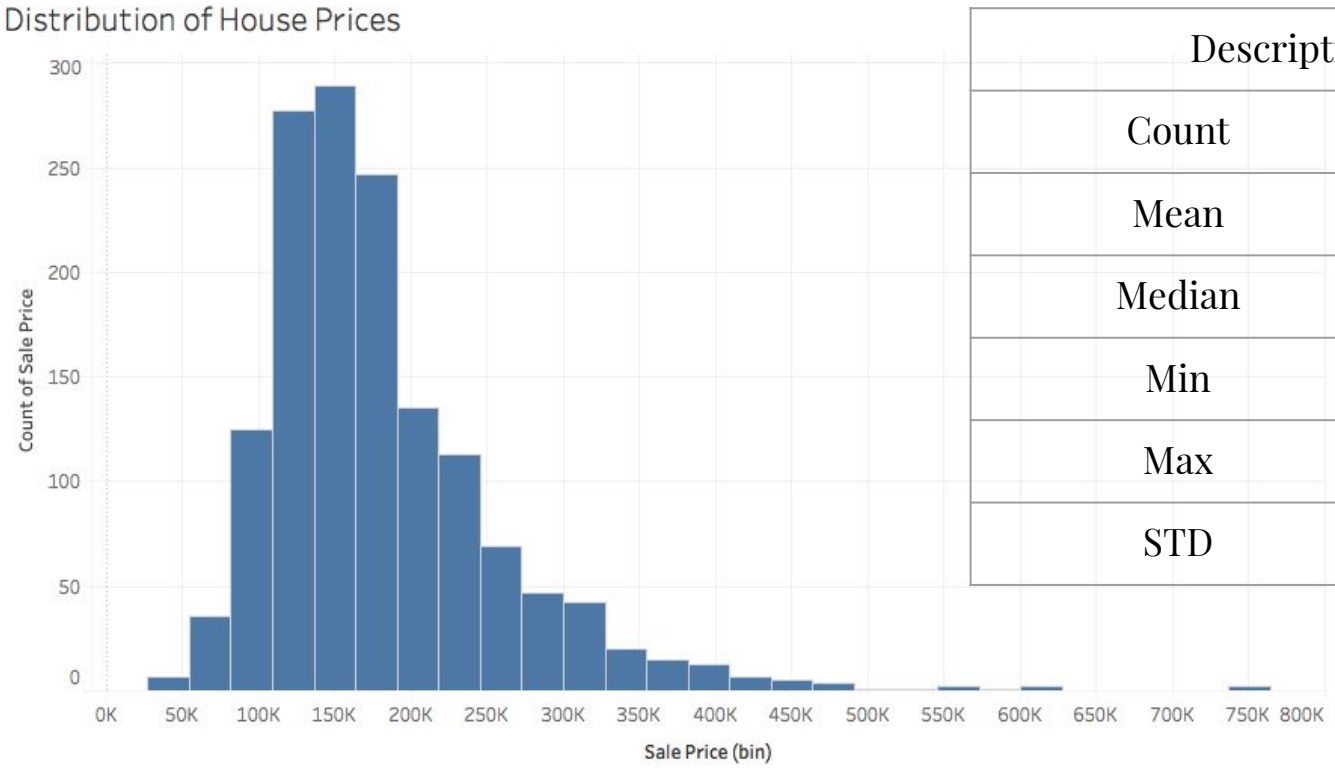
House Price Prediction Using Regression Analysis

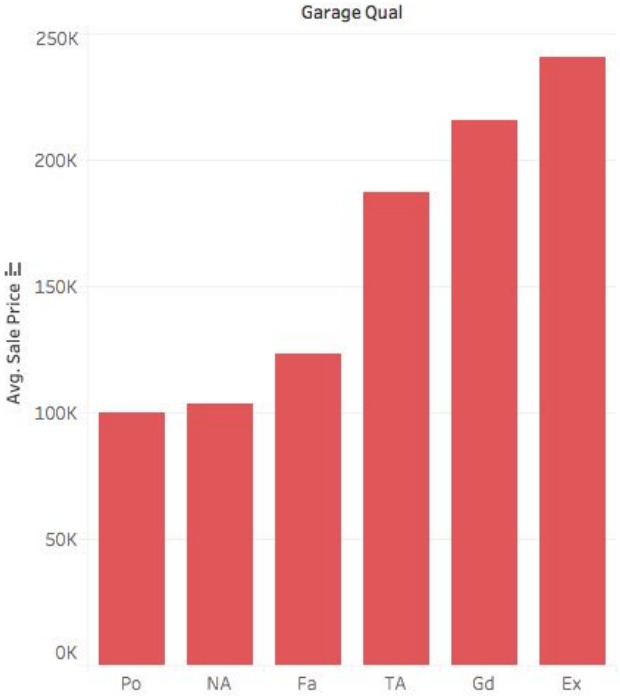
House Sleuths Team
8/27/18
NYC Data Science Academy

Dependent Variable: Sale Price

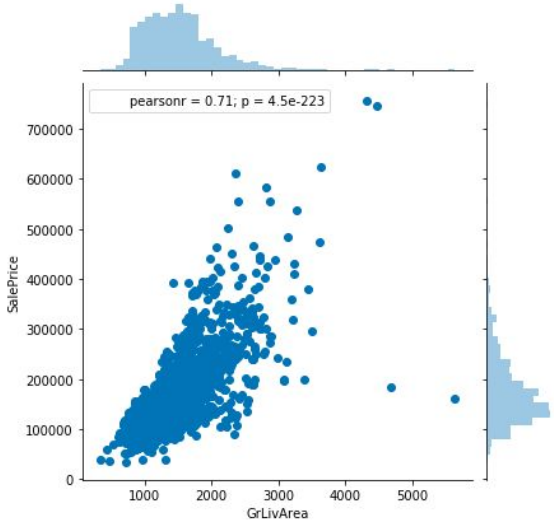


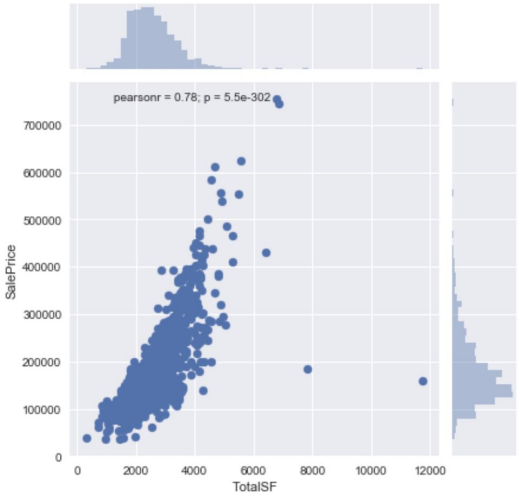
Descriptive Statistics	
Count	1460
Mean	180921
Median	163000
Min	34900
Max	755000
STD	79442

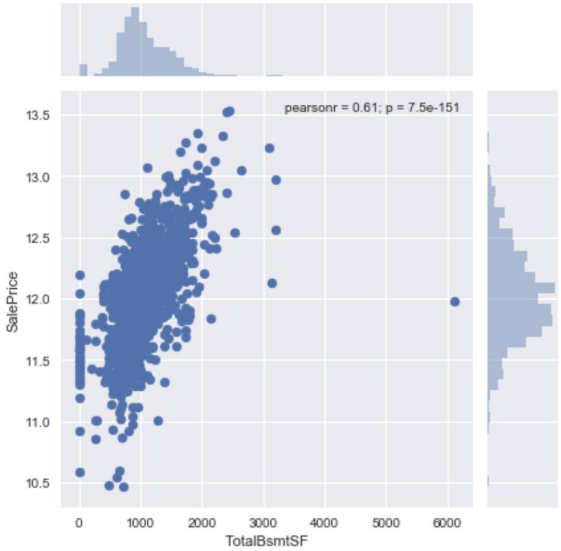
Categorical Variables



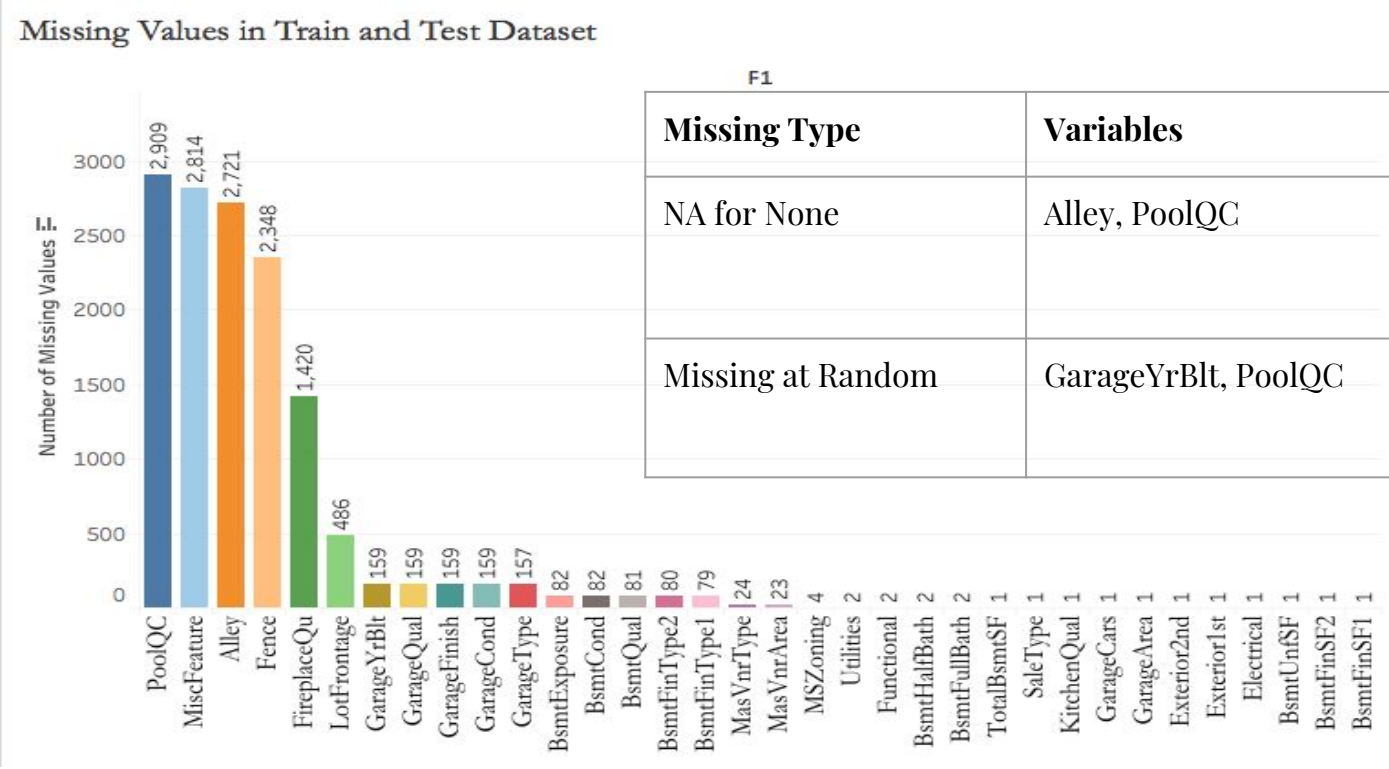
Numeric Variables







Missingness



Missing Type	Variables	Imputation Action
NA for None	Alley, PoolQC	o for numeric values NA for categorical values
Missing at Random	GarageYrBlt, PoolQC	Impute the value using other variables

Feature Transformation

Transformation

	Type	Feature	Values	Transformation
Categorical	Ordinal	Basement Quality	Ex, Gd, TA, Fa, Po	5,4,3,2,1
		Lot Shape	Reg, IR1, IR2, IR3	4,3,2,1
		Land Slope	Gtl, Mod, Sev	3,2,1
		Garage Finish	Fin, RFn, Unf, None	4,3,2,1
	Nominal	Neighborhood	CollgCr, Veenker	One-Hot Coding
Numeric	Skewed	Sale Price Basement Total SF	Right Skewed	Log Transformation
	Not Skewed	Garage Area	Normal Distribution (Roughly)	No Transformation
		Fire Places		

Feature Engineering Insights

DROP

1. YearBuilt ~ GarageYrBlt = **0.83**
   Off all total values in train and test set, 75.8% of the YearBuilt and GarageYrBlt values were the same.

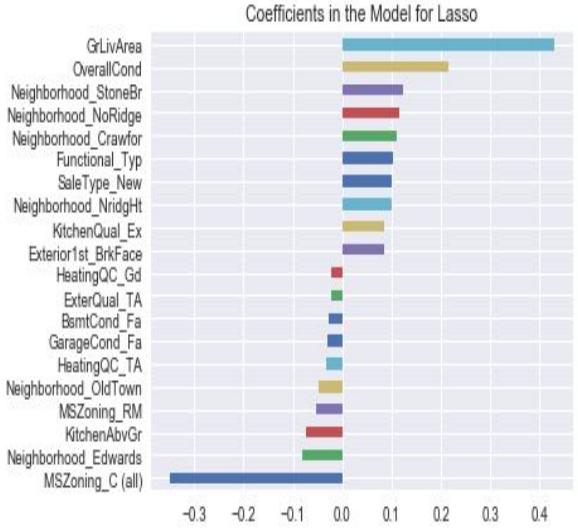
CREATION

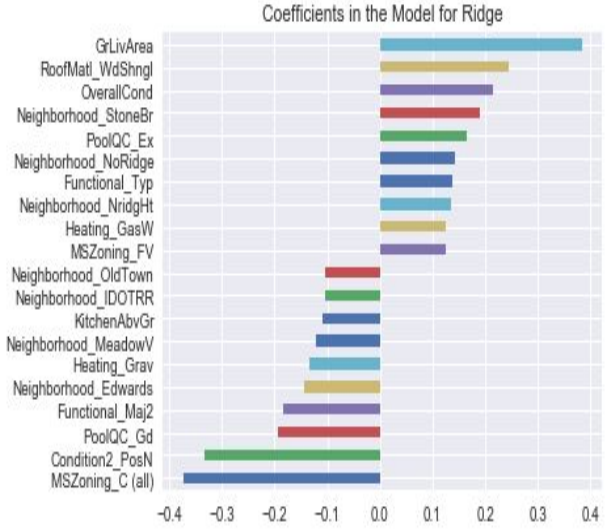
1. ‘AgeWhenSold’ = YrSold - YearBuilt
2. ‘YearsSinceRemod’ = YrSold - YearRemodAdd

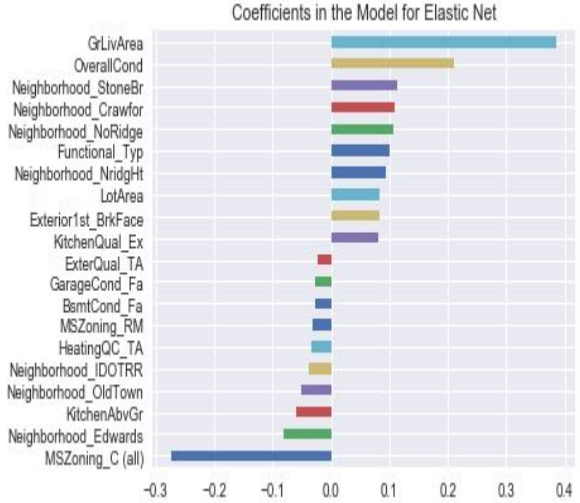
Regularization

- Automatically does the feature selection for you
- Penalizes extra features, sets coefficient to zero
- Reduce the model complexity
- Possibility of overfitting with linear regression
- Addresses Multicollinearity
- Use Lasso, Ridge and Elastic Net regression

Feature Selection

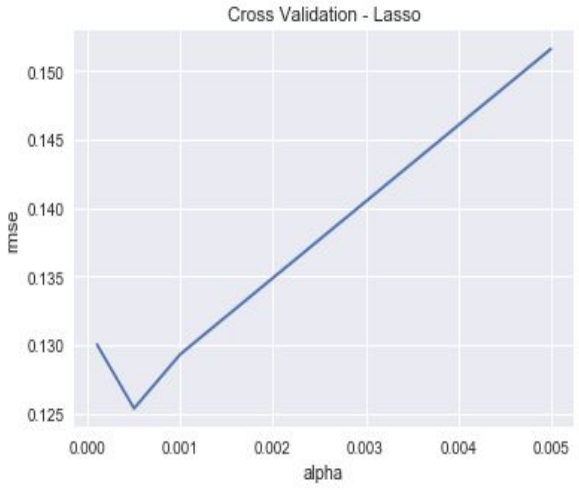


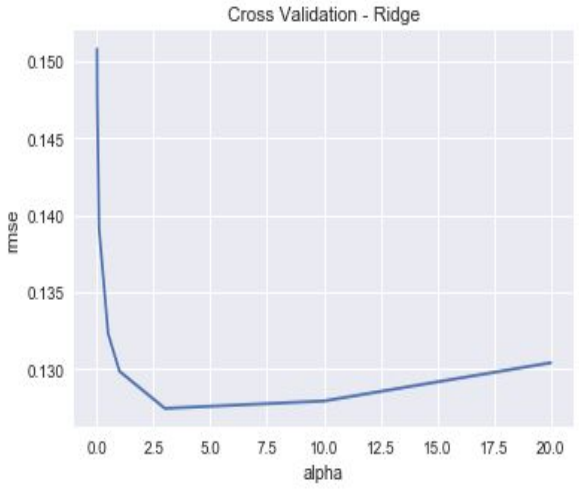


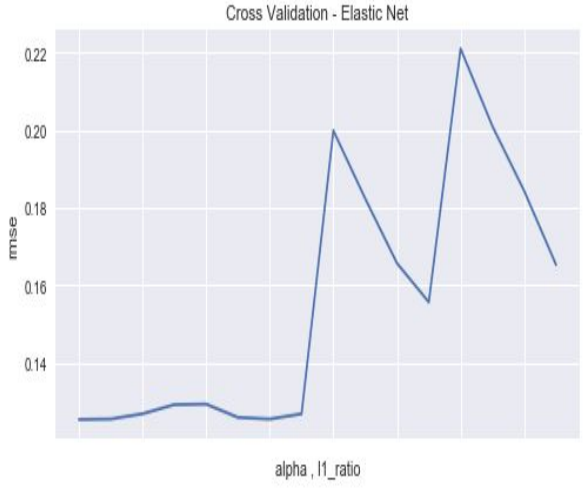


Cross-Validation

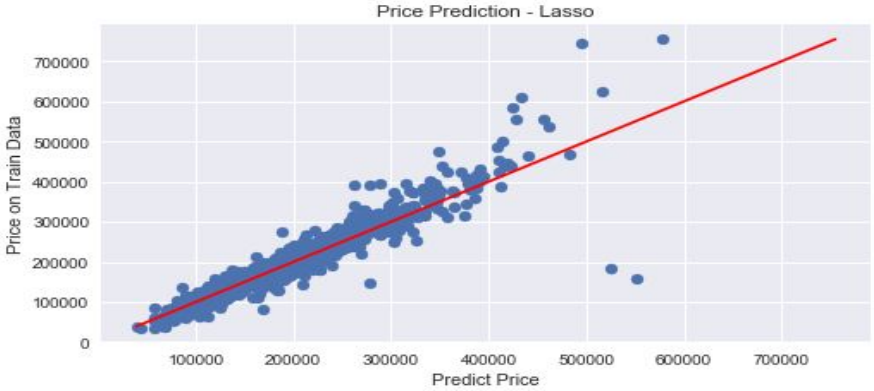
Lasso: Alpha = 0.0005, **Ridge**: Alpha = 2.8 and **Elastic Net**: (Alpha = 0.0005 and L1_Ratio = 0.9)

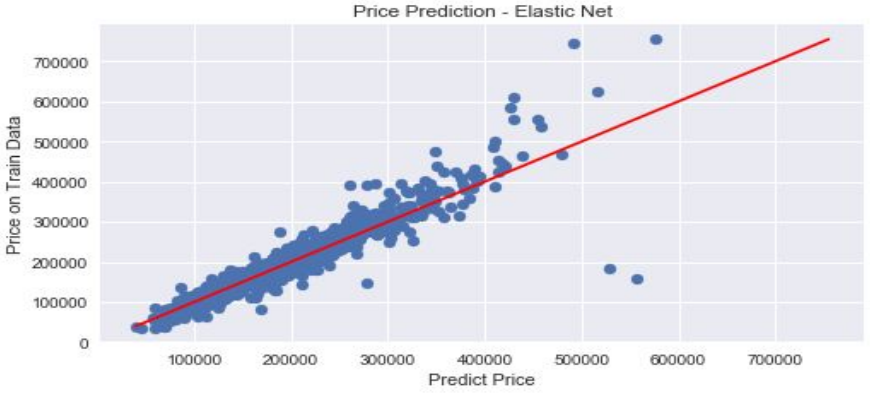


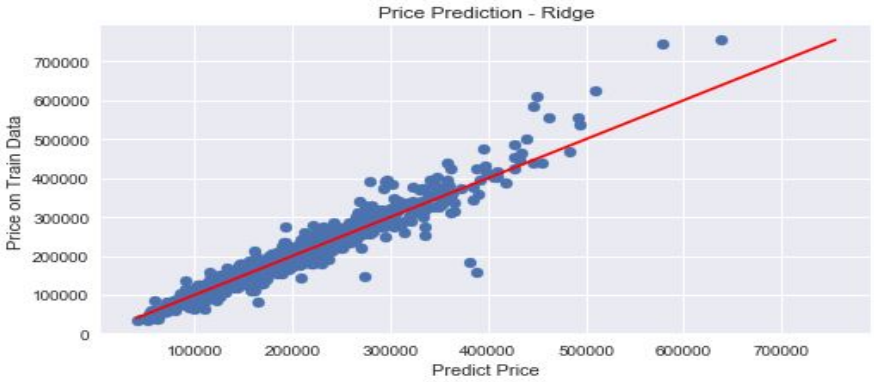




Pricing Prediction



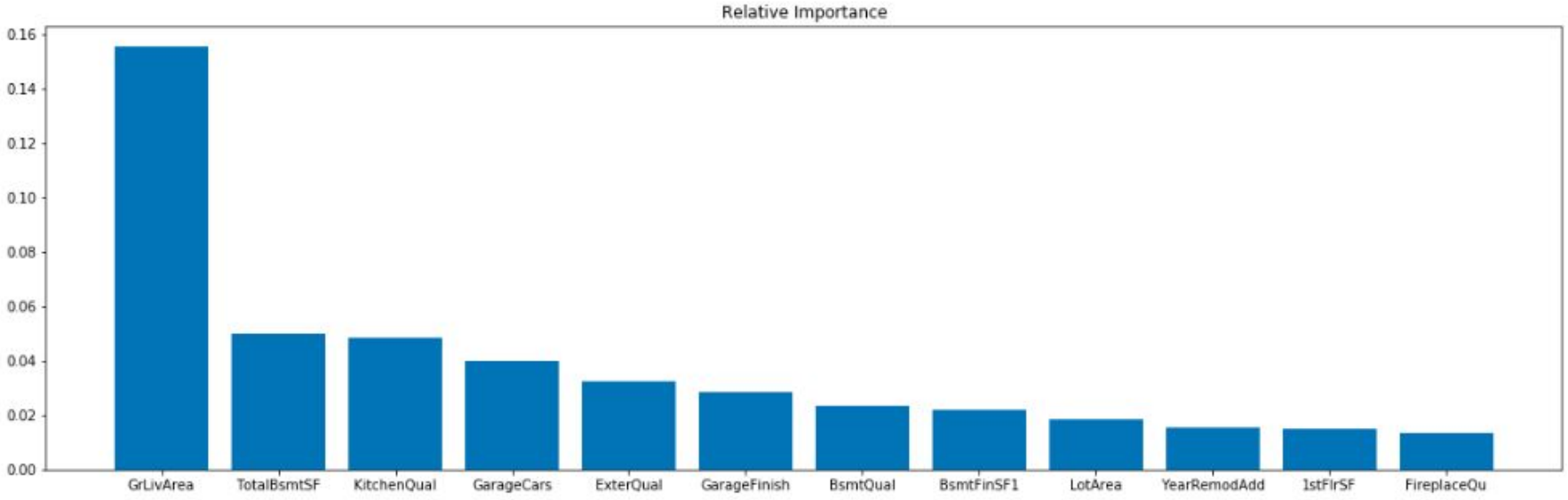




RMSE Log Error
Lasso: 0.127
Ridge: 0.129
Elastic Net: 0.126

Gradient Boosting Regressor

- Gradient Boosting Is One of Our Best Performing Algorithms
- Trained Gradient Boosting Machine Using the Entire Set of Features
- Tuned Hyperparameters Via Grid Search
- Best Model Parameters: Learning Rate (0.05), # of Estimators (2,000), Max Depth (3)
- Most Useful Features in Construction of Boosted Decision Tree were Above Grade Living Area Square Feet, Kitchen Quality, Total Square Feet of Basement Area, and Size of Garage in Car Capacity



PCA + Gradient Boosting Regressor

- Attempted to Improve Baseline Result by Reducing Dimensionality
- High Dimensional Data Can Be Sparse or Spread Out
- Used Standard Scaler to Normalize the Data
- First 150 Principal Components Accounted for Over 85% of Variation
- After Implementing PCA our Result Did Not Improve
- Reducing Dimensions Caused Loss of Some Important Information Not Only Random Noise

PCA + Gradient Boosting Regressor

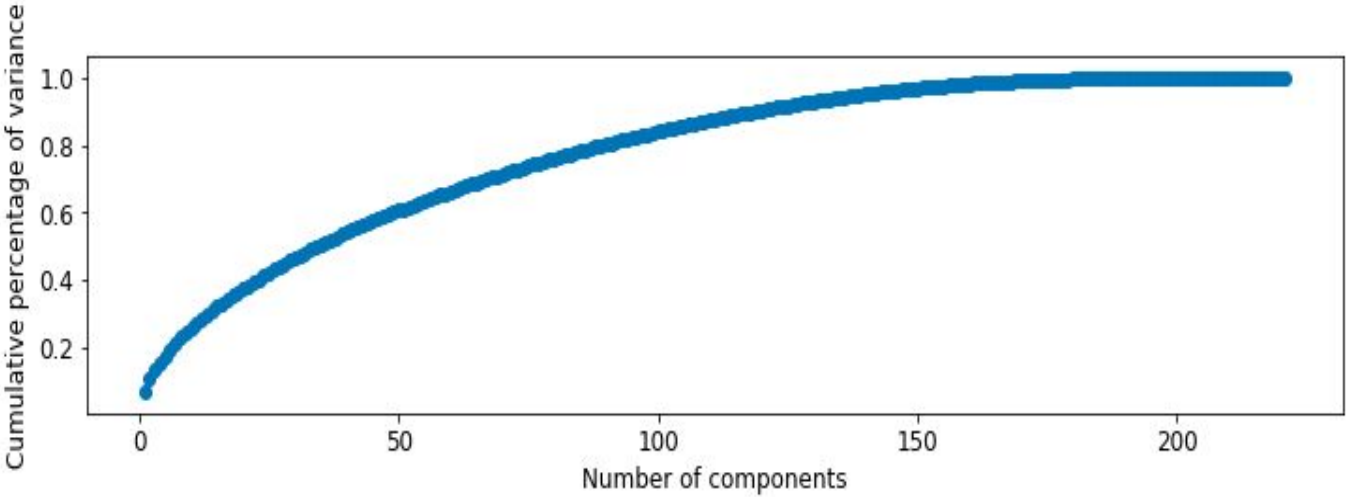
Best Parameters:

# of components - 150

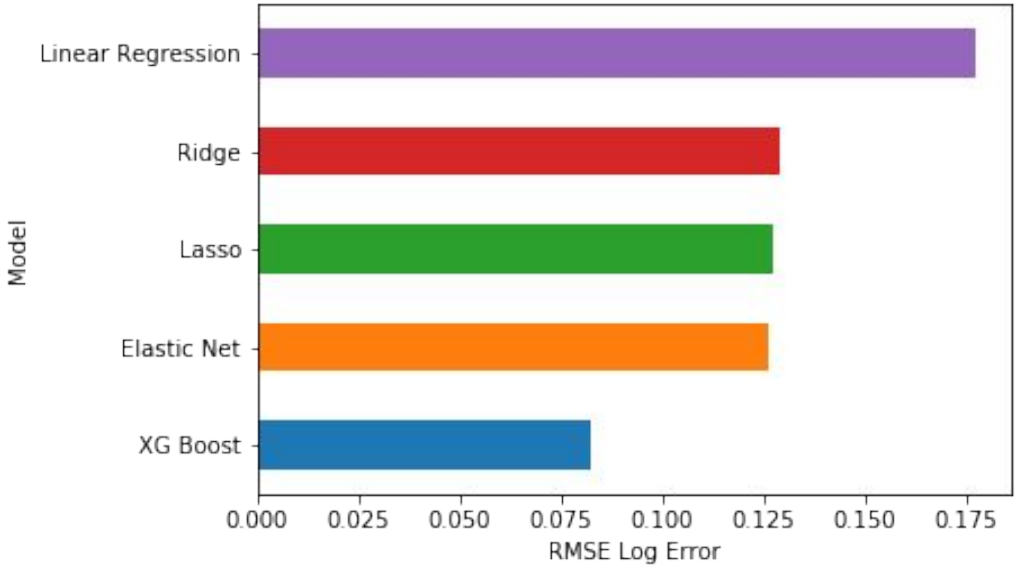
# of estimators - 2,000

Learning rate - 0.05

CV score: 0.87 vs.
CV score of 0.91 for
Gradient Boosting
without PCA



Model Comparison



What can we do better?

Ensembling and Stacking

RANK	MODEL	HYPERPARAMETERS	KAGGLE LEADERBOARD SCORE
1	Stacked model with ENet, Random Forest 1 (low depth), Random Forest 2 (high depth), Gradient Boosting 1 (conservative), Gradient Boosting 2 (aggresive) with linear meta model	ENet: alpha = 0.0005, l1_ratio=0.9 RF1: max_depth=4, n_estimators=1000, RF2: max_depth=9, n_estimators=1000 GBM1: n_estimators=3000, max_depth=3, min_samples_split=10 GBM2: n_estimators=2000, max_depth=9, min_samples_split=10	0.12260
2	Ensembling 50% ENet and 50% XGBoost	ENet: alpha=0.0005, l1_ratio=0.9 XGBoost: learning_rate=0.05, max_depth=3, gamma=0.0468	0.12485
3	XGBoost	learning_rate=0.05, max_depth=3, n_estimators=2200	0.12923

Conclusions