Predicting Residential House Prices

Brian McGuckin

Unit 03 Supervised Learning Capstone

Thinkful Data Science Bootcamp

1. Introduction

1.1 Why is predicting house prices important?

- A strong housing market:
  - Generates wealth for householders, increasing consumer confidence
  - Increases availability of credit from lending institutions
  - Incentivizes real estate investment and construction
  - Construction is a labor intensive industry and is a sizeable component of employment rate
  - Results in overall higher GDP and rate of economic growth
- House price forecasting:
  - Informs consumer spending behavior
  - Affects availability of credit, interest rates, investment, and construction
  - Can mitigate or prevent bubbles

1.2 Dataset: Ames, Iowa Housing Market Data

- Abstract Excerpt:

  "This paper presents a data set describing the sale of individual residential property in Ames, Iowa from 2006 to 2010. The dataset contains 2930 observations and a large number of explanatory variables (23 nominal, 23 ordinal, 14 discrete, and 20 continuous) involved in assessing home values."

- This dataset is a popular alternative to the Boston Housing Dataset and is currently being used for a Kaggle competition to predict house prices
- Note: Kaggle versions of the datasets were used for this project
  - Both the train & test data were introduced for cleaning, EDA & feature selection & engineering purposes
  - Models were fitted using only the train set, as there are no sale prices included with the test set
- Relevant links:
  - Dataset on Kaggle: https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data
  - Author Dean De Cock's Documentation: https://ww2.amstat.org/publications/jse/v19n3/decock.pdf

2. EDA, Cleaning, Feature Engineering

2.1 Data Cleaning: Outliers

- The dataset documentation makes special note of 5 outliers and recommends removing any rows where grlivarea is greater than 4000
- grlivarea: above grade living area in square feet
- The four observations shown here were removed per this recommendation (the fifth is in the test set)

2.2 Missingness

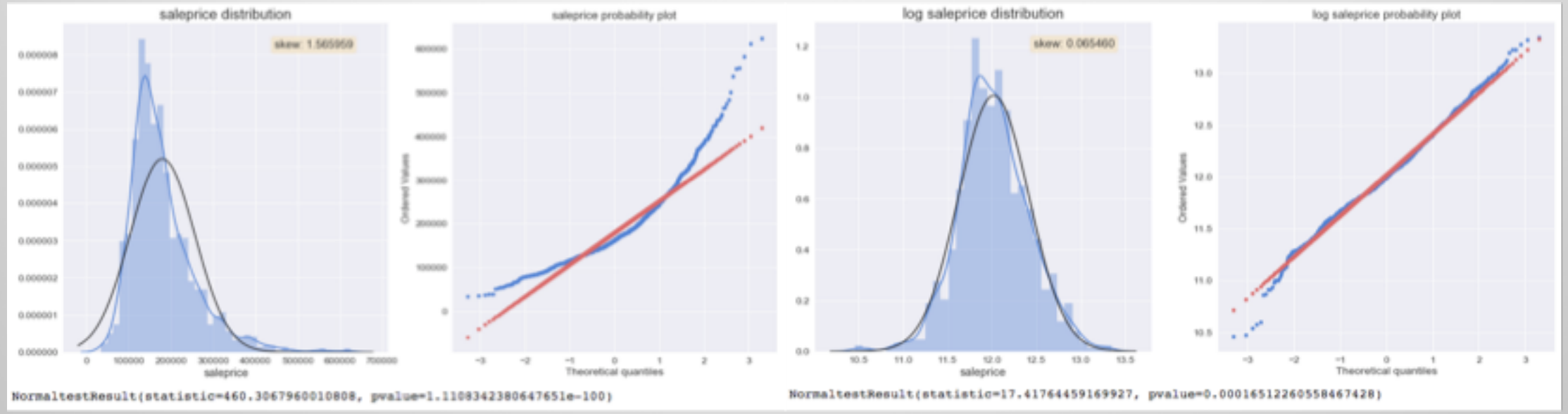
- Where missingness indicated lack of feature in question, imputed none or zero
- For missing categorical data where lack of feature was very unlikely or impossible, imputed column mode
- For missing numerical data where lack of feature/zero didn't make sense, imputed column mean

	variable	rows_missing
0	poolqc	2906
1	miscfeature	2809
2	alley	2716
3	fence	2344
4	fireplacequ	1420
5	lotfrontage	486
6	garagecond	159
7	garagequal	159
8	garageyrblt	159
9	garagefinish	159
10	garagetype	157
11	bsmtcond	82
12	bsmtexposure	82
13	bsmtqual	81
14	bsmtfintype2	80
15	bsmtfintype1	79
16	masvnrtype	24
17	masvnrarea	23
18	mszoning	4
19	bsmthalfbath	2
20	utilities	2
21	functional	2
22	bsmtfullbath	2
23	bsmtfinsf2	1
24	bsmtfinsf1	1
25	exterior2nd	1
26	bsmtunfsf	1
27	totalbsmtsf	1
28	exterior1st	1
29	saletype	1
30	electrical	1
31	kitchenqual	1
32	garagearea	1
33	garagecars	1

2.3 Target Variable Distribution

Sale price is not normally distributed

Log transformed to normalize



2.4 Independent Variables

Exploring independent variables and their descriptions from the data description file revealed a few issues:

1. Categorical variables represented numerically
   - Data for variables mssubclass and mosold was recoded as categorical data
2. Separate month & year variables
   - Separating month and year provides insight to seasonal and annual trends
   - Lacks ‘big picture’ chronologic aspect
3. Variable ‘yearbuilt’
   - Year house was originally constructed in
   - Does not explicitly reflect the age of the house when sold
4. Ordinal variables
   - Author specifies 23 ordinal variables but does not offer advice on how to treat them
   - Ordinal data type inconsistencies, mix between categorical & numeric without any apparent reasoning

2.5 Feature Engineering & Selection

- New features created:
  - Yrmo_sold variable combining month & year of sale
  - Age_at_sale variable (yrsold – yearbuilt) to represent house age when sold
- Addressing collinearity:
  - Garagecars (garage number of cars) and garagearea (garage size in sq. ft.)
    - Convey the same basic idea of garage size
    - Garagecars: more highly correlated with saleprice and a more intuitive representation
    - Kept garagecars, dropped garagearea
  - Square footage and room count variables
    - Tried a few methods to consolidate this explained variance
    - Simplest and best results from keeping sf variables (more highly correlated with saleprice) and dropping room count variables
  - Lotarea and lotfrontage
    - Neither had a very high correlation with saleprice
    - Scaled & averaged into a single feature

2.6.1 Ordinal Data Treatment

Ordinal Data as Categorical	
Pros	Does not require any additional assumptions Provides certainty that ordering isn't overstated
Cons	Lose information from ordering Can limit analysis: increase complexity (dummies), decrease statistical power

Ordinal Data as Continuous	
Pros	Preserves ordering Provides for analytic flexibility (interpretability)
Cons	Ordinal data isn't actually numeric Assumes evenly spaced categories (can result in overstating effects of order)

2.6.2 Ordinal Variables

The following methods were tried on ordinal data:

- All categorical
  - Recoded two ordinal variables where scale was entered numerically, created dummies for all
- All continuous
  - Categories were converted to numeric scales
  - Performed PCA on collinear variables
  - Principle components created for garage, basement, and quality variables
- Mixed: intuitively selected
  - Continuous when values not necessarily limited to defined categories (e.g. exterior quality: excellent, good, average, fair, poor)
  - Categorical when values limited to specified categories (e.g. driveway: paved, partially paved, dirt/gravel)
- Mixed: ordinal variable types unchanged
  - Ordinal scales were not converted, created dummies for categorical variables
  - Included to control for the possibility inconsistent types were deliberate

3. Modeling

3.1 Models

- Linear:
  - Ridge/L2 regularization parameter
    - Penalizes large coefficients using sum of all coefficients squared
  - Lasso/L1 regularization parameter
    - Tries to force small parameter estimates to be zero
    - Acts as feature selection
  - Elastic Net
    - combined L1 & L2 regularization parameters
    - Increasing L1 ratio value increases weight given to L1 parameter
- Ensemble:
  - Random Forest: bagged decision trees
  - XGBoost: popular implementation of the gradient boosting decision tree algorithm
- Excluded:
  - Unregularized linear regression: consistently the worst performing linear model
  - SKLearn Gradient Boosting: outperformed by XGBoost
  - Support Vector Regression: very slow to train, poor performance

3.2 Evaluation Procedure

- Primary evaluation metric: Root Mean Squared Error
  - Measures the difference between predicted values and observed values
  - Produces a non-negative value, lower is better value where 0 indicates perfect fit
  - Absolute measure of fit (as opposed to relative measures such as r-squared values)
  - Scale dependent: comparisons are only valid on the same dataset
- Model evaluation on this dataset:
  - Models were evaluated using the mean RMSE from 5-fold cross validation
  - RMSE was calculated between the log of the predicted value and the log of the observed sale price
  - Same metric used for the ongoing Kaggle competition to predict house prices using this dataset, can get an idea of how models are performing in larger context
  - Using log normalizes distribution of results (as a consequence of this also normalizes residuals)
- Secondary consideration: runtime
  - Dummies created from categorical variables increased dataset size considerably
  - Measure the impact on runtime for different ordinal data types

3.3.1 Results

5 Fold Cross Validation RMSE Means				
	Ordinal Categorical	Ordinal Continuous	Ordinal Mixed	Ordinal Original
Ridge	0.11799	0.11381	0.11455	0.11460
Lasso	0.11477	0.11052	0.11103	0.11215
Elastic Net	0.11471	0.11056	0.11117	0.11209
Random Forest	0.14114	0.13652	0.13661	0.13798
XGBoost	0.12977	0.12049	0.11994	0.12172

Runtime in Seconds				
	Ordinal Categorical	Ordinal Continuous	Ordinal Mixed	Ordinal Original
Ridge	2.60	1.36	1.63	1.87
Lasso	16.62	9.44	10.37	12.31
Elastic Net	124.93	65.33	72.1	89.90
Random Forest	555.84	416.02	439.63	432.82
XGBoost	195.30	148.076	162.10	180.81

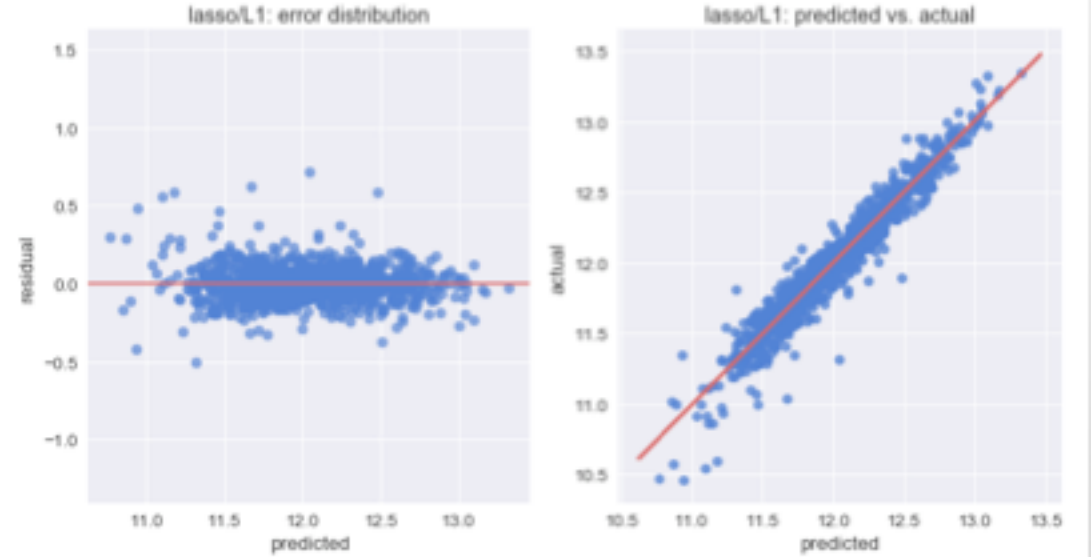
3.3.2 Results: Best Performing Models

Continuous Ordinal, Lasso: 0.11052

```
alpha broad search: 0.0005
1st cv rmse score mean: 0.11052414078710895

alpha tuned: 0.0005
2nd cv rmse score mean: 0.11061853170758451

total runtime: 9.44 seconds
```

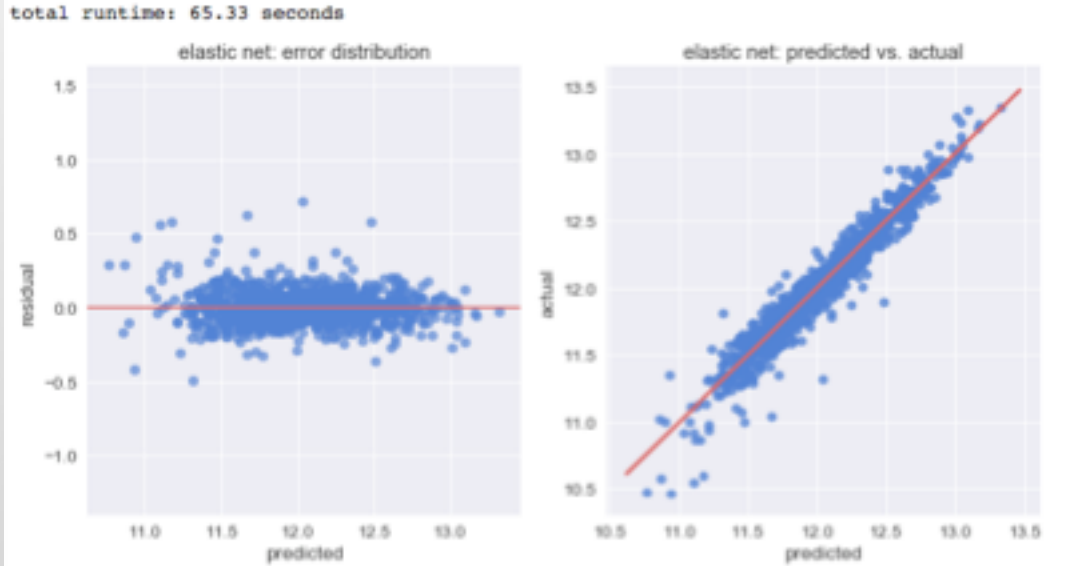


Continuous Ordinal, Elastic Net: 0.11056

```
l1 ratio broad search:  0.9
alpha broad search:  0.0006
1st cv rmse score mean:  0.11074299814416791

l1 ratio tuned:  0.95
alpha tuned:  0.0005
2nd cv rmse score mean: 0.11055610976909888

total runtime: 65.33 seconds
```



3.3.3 Results: Analysis

- Ordinal variable treatment
  - All models except XGBoost performed best on continuous ordinal data
  - All models ran quickest on continuous ordinal data
  - In general, more continuously represented ordinal data resulted in higher predictive accuracy and faster runtime
- Lasso and Elastic Net
  - Produced lowest RMSE within each subset of ordinal data treatment in addition to lowest overall RMSE
  - Tuning elastic net L1 ratio resulted in values approaching but never quite reaching 1 (full L1, no L2)
  - Between subsets, elastic net RMSE variance lower than lasso ($\sigma^2 = 2.51\ vs.\ 2.69\ x\ 10^{-6}$)
- Ensemble model performance
  - Random forest returned the highest RMSE for each subset
  - XGBoost was the only model that performed best on mixed ordinal data
  - XGBoost was the most consistent performer across all subsets ($\sigma^2 = 1.58\ x\ 10^{-5}$)

3.4 Conclusions & Thoughts Moving Forward

- Ordinal variables
  - For this data, information in ordering explains a significant amount of variance
  - How ordinal data is used can measurably impact prediction accuracy
  - Ordinal variables should be carefully considered
  - Continuous data performed best here, but that may not always be the case
- Model performance
  - Feature selection was an important aspect; L1 regularizer is helpful for dealing with large amounts categorical data/dummy variables
  - As shown by elastic net, L2 parameter does provide additional value even in small doses
  - XGBoost's resilience to ordinal data type may prove useful in cases where treatment is unclear or highly subjective
- Preferred model: Elastic Net, for ability to combine best components of L1 (feature selection) and L2 (overfitting prevention)
  - Honorable mention: XGBoost, while not the best predictor for this task, this model's boosted tree implementation compared favorably to other tree based models, both in predictive accuracy and speed