Predicting Real Estate Sales Using Linear Regression

By Johnny McGregor

Problem Statement

This was my first machine learning project at General Assembly. Up to this point the only technique we had been taught was linear regression. The goal was simple - use linear regression along with any regularization and/or feature engineering techniques we choose to minimize loss. The sale price of houses in Ames, Iowa serve as the target variable in this project. The data comes from a Kaggle competition.

Data Cleaning/Preprocessing

The data itself is provided by Kaggle but there was plenty of cleaning and processing to do. The 82 columns in the data set include 23 nominal, 23 ordinal, 14 discrete, and 20 continuous variables (as well as 2 observation identifiers). More information on the data can be found at this link: http://jse.amstat.org/v19n3/decock/DataDocumentation.txt.

The ordinal columns such as "exterior quality" are represented from "poor" to "excellent". All of the columns that were entered in this way were changed to represent a scale from 0 to 4. A handful of these ordinal columns had a lot of missing values. Upon further examination I determined this to be due to a house not having a basement, garage, or pool. Instead of getting rid of these observations I filled those missing values with a zero to represent the fact that the observed house did not have these features.

I was able to convert most of the nominal columns to binary. For example, the Zoning column has six possible values:

- 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)

Nearly 1,600 of the 2,051 houses fell under the “RL” category, with 4 of the seven categories having less than 20 houses in them. For this reason, I changed the column to be a 1 if it was “RL” and 0 if it was “Other”. Of the few columns where transforming to binary was not a good option, I decided to make dummy columns. Neighborhood and foundation type were two examples of this.

Modeling

I used the .corr() function to see which features correlated the most with the target variable and visualized this with a heatmap in seaborn. I also used statsmodels.api to look at which features were considered statistically significant. The .summary() is a useful tool in stats models for looking at the p-values for each features.

Once I began to select / discard certain features, I created some interaction terms and polynomial features in an attempt to account for multicollinearity. Since total square footage, first floor square footage, and basement square footage correlate highly with one another I wanted to see if multiplying them together would be a good approach as opposed to using all three or deciding not to use two of the three.

I fit a linear regression model with no regularization, as well as models using lasso, ridge, and elastic net regularization. I fit these four models with selected features including the interaction terms, and then again adding in the polynomial features. The features I used can be found in the “02 – Modeling” notebook in this repo.

Of all the models and regularization I attempted, the lowest root mean squared error of 23,609.82 came from using ridge regularization with the interaction terms *and* polynomial features.

Conclusions

At this point we had not yet learned about grid searching but if I were to attempt this again with linear regression as my only option, I would see how much of a difference hyper parameter tuning would make with the regularization methods. This project was as much about learning how to clean and transform data as It was about modeling. In later projects we were able to utilize random forest, boosting, neural networks, support vector machines, etc. to optimize loss functions.