Summary for Project Luther

Predicting Used Sailboat Prices

Sean Davern
Seattle, Fall Cohort, weeks 2-3

Project Design

I used the website Boats.com to scrape listing prices and boat information and then modeled the listing price. Two goals for the model were initially proposed: 1) enable a quick, rough estimate for situations like insurance renewals or when considering significant projects on a boat one owns and 2) for someone to be able to more easily compare across manufacturers, models, age, location and other cost-impacting factors. While a model might facilitate the first goal, I later realized that in such a situation, the user is likely to have all the necessary information to use available tools, like the NADA Guides for a reliable estimate. The second goal, however, is a more legitimate need especially for someone at the early stages of considering buying a boat. A pricing "model" might facilitate something like a collection of sliders where the user could play with slider positions to see how the factors impact the price. Alternately, a model might be used to create (perhaps interactive) 2D graphs that show cost relationships between factors. The object of *this* project was, however, just to see if a reasonable model could be regressed from the available data. Given that 10% error is typical for the NADA Guides estimate[1] and given that bigger uncertainty is generally acceptable for early in a project, model projections that approach 10-20% would have been very encouraging.

To keep the problem manageable and comply with the assignment guidelines I narrowed the data collection and modeling to sailboats in the 40-50 ft range. This turned out to be a reasonably rich collection of information as discussed below in the Data section. Given the value of boats in this size range (generally $100K- $1M) people listing their boats are motivated to provide extensive information.

The MVP[2] used the scikit-learn Linear Regression model to simply fit price as a function of manufacturing year. I had initially implemented the use of the Statsmodels Python package because I really like the OLS Regression Results output - a format very familiar. However, hours before the MVP was due I was struggling to get a range of fit metrics and was encouraged to switch to scikit-learn Linear Regression. I made the switch and was later really happy because it made it much easier to incorporate the scikit-learn DataFrameMapper with StandardScaler and OneHotEncoder preprocessing packages as well as make_pipeline to automate model construction, expansion and execution.[3] Both the MVP and later regression work used test_train_split to create training (80%) and test (20%) data splits.

Heteroscedasticity in the MVP model of $Price = f(year)$ motivated applying a log transform for price. This was not surprising considering such a transform is common for measures in dollars and given the prices ranged over more than an order of magnitude.

Soon after the MVP was completed I implemented the pipeline preprocessing and make_pipeline packages. After very quickly expanding the model to include boat length I embarked on including boat make, the first categorical model factor. EDA showed there to be 404 unique manufactures that one hot encoding would turn into 403 modeling terms. Even with nearly 3,000 data records, this seemed like a lot of factors to add. EDA also showed that there were 298 of the 404 unique manufactures that only had 1 or 2 data records. I decided to create a separate factor, 'mfg' from 'make' column that recatagorized the 298 manufacturers with only 1 or 2 boats as mfg="other". This resulted in 107 manufacturers, including the 354 boats (~10%) lumped together as "other". At this point I also implemented LassoCV using 5 fold cross validation. Even when the initial attempt resulted in an alpha (lambda) on one end of the default range used, several manufacturers where shrunk to zero. I finally added boat "class", a collection of sailboat sub-categories with 16 unique values, and implemented a method to ensure the resultant alpha was approximately in the middle of the range ultimately specified.

Results

I ended up running out of time before all available factors could be integrated into the model. I focused my time on the factors I suspected would have the biggest contributions as well as the amount of time required to incorporate them. Probably the most aggregious ommission is some aspect of location. Listing came from all over the US, Canda, Australia, Great Britain, Europe and New Zeland among other places. In an 11th hour effort, I incorporated a "currency" factor representing the type of currency the listing price was provided in as a surrogate for location. Surprisingly, this factor didn't contribute significantly to modeling result and so I commented out the code that incoporated this effort (though still visible in the jupyter notebook).

The final model R^2_{Test} was 0.67 and MAE_{Test} = 0.25. Given the log transform of the price, the MAE equates to about a 30% error. My sense is that a 20% error might be achievable implemening these potential improvements: 1) integrating location of the boat into the model, 2) adding the Age*Antique interaction and 3) allowing regularization to cope with ALL the 404 manufacturer values rather than loosing the manufacturer information by lumping.

Tools

- Python
    - Data acquisition (scraping): BeautifulSoup
    - Data Storage: pandas and csv files
    - Data Analysis: scikit-learn LinearRegression, LassoCV, Jupyter Notebooks
    - Presentation: matplotlib
- Microsoft PowerPoint and Markdown

Data

Originally the Boats.com site reported 3,249 sailboats in the 40-50 ft range. Data collection involved first scraping the 204 pages of listings to get url's to the individual boat detail pages. It turned out that this list included a number of new boat listings that didn't include a price as well as random advertisements interspersed within the listings. The resultant collection of 3,020 urls were then scraped for boat detail information. 19 of the urls turned out to not resolve (perhaps boats sold, perhaps other site issues). The scraping effort netted 3,001 boats worth of records. All scraping was done using the BeautifulSoup package.

While scraping I collected quite a bit more than the 4 factors modeled (year, length, make, and class). However, some of the factors scraped where known to have significant correlation with each other. For example, Make and model would perfectly correlate with class, nominal length, length overall, and displacement. By not modeling with the boat model I avoided both this correlation as well as discrepancies in the model text. Boat Model is also unlikely to add any information that year and nominal length don't already provide. Furthermore, length is a much more useful factor than boat model for the type of equation model I ultimately desired.

One detail about the data I noticed was that only ~750 boats were from the US (listed in currency of US$). More than twice that many were listed in euros. That was a complete surprise and a significant motivation to incorporate location into the model. Because the currency symbol was part of the price string, my focus had originally simply been to convert the currencies to US dollars. It wasn't until much later that I did the EDA to discover this detail!

What I Would Do Differently Next Time

I scraped a bunch of text information about electronics, equipment, rigging, and sails that came with the boat. This took a fair amount of time and a high percentage of boats didn't provide this information. I thought that I might as well grab it since delays I built into the scraping led to about a 4 hr (at night) run time. I didn't end up using it so I could have skipped that work.

When I see prices included in other currencies I won't assume the data is mostly from the US.

Regularization was a completely new concept for me! It really came home to roost when I saw how it could shrink individual coefficient values of categorical factors. Next time I would at least try letting the regularization handle the many factor levels that turn into model terms before investing effort in a scheme that discards potentially important information in a lumping scheme. Those 1-off manufacturers might well have been high or low value builders encoding important information!

The pipeline modeling tools were awesome. I'd do that out of the gate next time!

---

1. Boat Values and Pricing Guide, BoatTrader.com↩

2. Davern, Sean. MVP.ipynb. Jupyter Notebook created from a commit completed when the MVP was due to facilitate presentation graphics. ↩

3. Davern, Sean. Modeling.ipynb. See "Setting up the analysis pipeline" section.↩