For my project, I spent 2 days building and refining my web scraper to crawl amazon listings. In the process, there were two obstacles:

1. Amazon webpage content is dynamically generated by javascript, so that left me with selenium. Some buttons/links wouldn’t even generate until you actually scrolled down to a certain section of the site (for example, the ‘next page’ button)
2. The ID and CLASS tags on each listing varies from one to the next, and sometimes even the layout is completely different, so in the process I had to iterate through 10 listings to build a scraper that would capture most of the possible variance. Even then, there was still a bit of missing data, but thankfully I still had a lot of good data to work with.

I also wasn’t sure if I was going to compare the data with kickstarter, so I had been working on a kickstarter scraper simultaneously.

Once the data was cleaned, there was actually a lot of work cleaning the data. Here I spent another 2 days working with the data that was extracted mainly as text. For example, the product ranking, star rating, launch date, and category was embedded as one long string, and in various orders, so I had to implement a lot of regex parsing to grab the data I needed. If you check my “Amazon\_data\_clean\_and\_analysis” notebook, you’ll see that nearly 1/3 of the code was spent extracting data from the data that was scraped.

**Data Analysis**

For the data analysis section, I essentially revisited all the linear regression lecture notes from class and implemented **almost** everything. One thing I should’ve done was look through the Practitioner Workflow document, because in retrospect, I realize I was taking a shotgun approach and just applying every model (and validation/testing methodology) on all my features, hoping to get a higher score. After seeing everyone’s presentation, I realize I should’ve started with fewer features and worked my way up. Additionally, I apparently forgot all about applying logistic transformation to my obviously skewed data. In my most recent notebook, I have fixed the code to use logistic transformation, and my R2 has gone up to 0.42 on validation data and 0.38 on testing data with just simple linear regression. Interestingly, polynomials now have a detrimental impact on the R-score.

Meanwhile applying random forest boosts the score up to 0.678, which is surprisingly high for the limited data I have. I am aware that a lot of inaccessible are factored in the ranking system for amazon, so it’s pretty cool that I’ve gotten a 0.678 score with just what I managed to scrape.

Last interesting observation is that with LassoCV, the coefficient for ‘Review\_count’ was reduced to 0, but with Random Forest, it is actually the second highest in terms of ‘importance’. One coefficient they both have in common is 5 Star ranking, which comes out as first in terms of importance on both models.