Alec Lovlein - Journal of Progress

This bike ad dataset has somewhat detailed text data, some low res images, and a simplified csv that maps image to price. I'm giving myself 2.5 hours to "complete" an analysis of the dataset, so the problem we tackle shouldn't be too complex. I would be surprised if I got ay result at all in 2.5 hours, I'm really just shooting for a start at a few different stages where the missing glue between can be easily explained.

Let's approach it from a business perspective: we want to make money flipping these bikes. I was amazed when selling our house at the increase in price a house was able to get with professional pictures. Things like lighting, angle, etc really influence buyers and may push them over the edge to schedule a showing. More showings -> more money. That probably applies here as well. I bet we can buy a cheap bike, take a better picture, and flip it for a profit.

For this let's restrict ourselves to used bikes only. It is probably not possible to flip a brand new bike. Next, let's see bikes with pretty similar characteristics in their text description and other meta data. We are looking to group bikes based on what we think should obviously influence price, without looking at price. Once we have clustered bikes (independent of listing price - that is important), we can start to look at price trends inside a cluster. I'm willing to bet (and will investigate) that the lower end of prices in each cluster are due to non-flattering images. If we get some stats to confirm that, we have a real shot at flipping some bikes.

Now, what makes an image flattering? That may get out of scope on this mini project, but as a start we can maybe look at a color histogram and average brightness for the image. My gut says a color histogram should have a few peaks signifying an image that is not too boring, but not too busy. For example, this one (8296) is too busy for my taste:

Now this bike (5964) looks more expensive for many reasons, but I think the picture sells it much better.

Ok cool let's get started. A good first step is to convert the json files to tables, filter to used bikes, and get started on cleaning the remaining relevant meta data. I'm going to kick off the 2 hour time at this point, which will give me 15 minutes post-analysis to wrap up conclusions and document potential improvements or further work.

Alright, first glance at the data really shows that eBay is a general resale site, and Bike Exchange is made for bikes. Let's use the features in Bike Exchange and see if we can get matching data from eBay.

That function is read_bike_data, and it is pretty quick and dirty. Ideally I'd have more robust combinations using the free text fields. We are throwing away a lot of potentially good classification data with the current method.

Ok, so in 90 minutes I was able to get the data read, unioned, and crudely cleaned / transformed. We haven't done anything with the images - let's at least look at a 3D bar chart with average color, brightness, and price to see if there is any correlation there. A more robust solution would tackle that color histogram, but that is harder to visualize as an input to another stage of the model.

Shoot, ran out of time there as well. In the remaining 20 minutes I was able to get the image read and converted to numpy so we could calculate the most dominant color(s) and the perceived brightness.

Overall, my next steps would be to continue attacking the problem from both sides. I'm interested in seeing if color and brightness have an influence on price. That is pretty straight-forward from here: get most dominant color for each image, classify the rgb value as a color of the rainbow (red, orange, yellow, green, blue, indigo, violet), get the perceived brightness, then use those two values (color, brightness) as independent variables and test the assumption that price is a dependent variable. Even a weak correlation here is enough to let us know we should keep pushing down this path.

Next, we group the photos based on features of their text descriptions using clustering. We probably just use title and seller notes to help fill in actual feature columns like size,

type, brand/model, brakes, etc. Once the actual feature columns are as full as they reasonably can be, we can just go with categorical encoding https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html#sklearn.preprocessing.OneHotEncoder on each column to build the final feature set. We could then feed those to kmeans++ and hope to find somewhat significant differences in price statistics across clusters. Maybe one cluster is more volatile, and another has a very high average, etc.

To finish out the business case, we could find “underpriced” bikes in each cluster, purchase them, then retake photos with the right color/brightness to hopefully get a profit.