NAME: Maxfield Green
DATE: 5/04/2019
Final Project Technical Review
STAT/CS 387

Document Overview

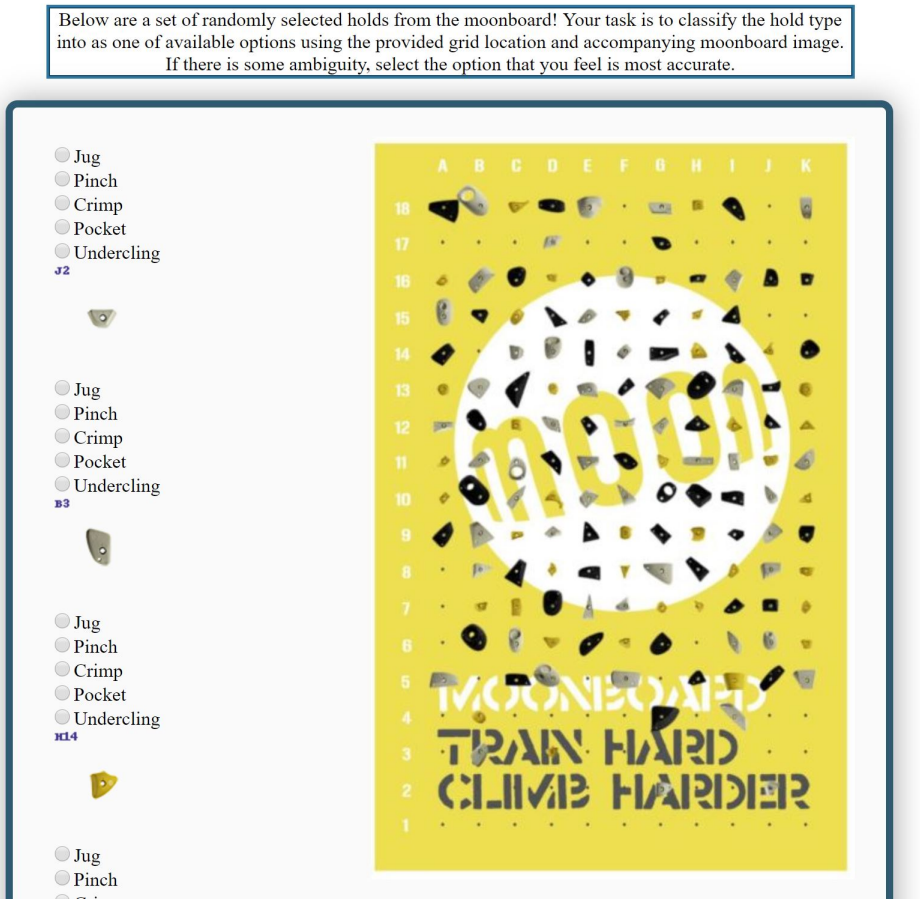
A technical review of this project can be broken down into three different sections: data collection, data processing and analysis/model building. Each section presented unique challenges and learning opportunities. For each section, I will outline the process, involved code, challenges and extraneous methods if applicable. At the end of these sections, I will discuss the scope of the final project in comparison to the initial project proposal.

Data Collection

There were two work heavy components of data collection. The data set from this project is made of up three different parts, climbing route profiles form the MoonBoard web/mobile app, hold labels and depth measurements. Initially, I was in contact with the North American branch of MoonBoard who had excitedly agreed to supply me with a sample of the MoonBoard data from climbing routes set on MoonBoard walls in the United States. Over the course of the first week of this project, the North American branch went out of business leaving only the original European contingent of the company. My emails from the North American branch began to bounce back as all of the accounts had closed. I reached out to the European side but did not receive any response. However, the moon board website had a library of every single climb set using the app. I spent many days writing scripts attempting to login into the moon-board website and scrape the information written into the HTML. I learned to use python packages Beautiful Soup and Selenium to perform page changes and text scraping. However, this limited me to information coded directly into the website structure. I started reading more about how data is sent from data bases to websites and discovered AJAX request that are used for quick data transfer. After hunting around, I located the AJAX request used to populate the library of climbs displayed on the website. By copying the cURL request directly into my command prompt, my local machine received the response which contained a sample of the data I was interested in. I was able to modify the cURL to collect the full dataset. The modified cURL is saved in the req.sh shell script. I wrote in some commands to change the page and increase the number of climbs reported per request. This script saved the JSON response files to a local directory. The files are in a subdirectory titled routes_json.

I was not satisfied with the features available to me from the MoonBoard website. I wanted more information at the level of each hold. I am personally very familiar with the MoonBoard and use one frequently. I debated categorizing the holds myself. However there is a level of subjectivity when classifying a climbing holds type, edge cases between the classes can be different for different people. Labeling them myself introduced. bias into the model, thus this seemed like a great opportunity to collect several labels per hold to estimate the true labels. Thus, I designed and implemented my own web crowd-sourcing platform for hold label estimation. I made a very simple html form template that shuffled the holds exposed for classification every time the page was loaded. After performing a few initial informal tests, I discovered that limiting the number of holds classified to 20 per page maximized volume while maintaining quality. The user responses were sent redirected to a "google sheets" spreadsheet that populated the sheet cells with the user responses. The html and JavaScript for the website are stored in final_project/work/index.html.I am extremely inexperienced in writing JS so I apologize for the cleanliness of the code. There are a few egregious instances of long lists of strings that represent different paths and files. It's likely that this could have been executed better. However, I was focusing on the functionality of the website. The website is hosted on the UVM silk server and is accessible at https://mgreen13.w3.uvm.edu/web_scrape/index.html The "GoogleScript" that intercepts the AJAX response and fills the google sheets is stored at final_project/work/hold_rec.gs. This was written specifically for this purpose by the Google web support team. Figure 1 displays a snapshot of the top of the website. Each hold has a picture and a grid coordinate that workers use to locate the hold on the supplied moonboard image. The images were also collected via a cURL script off the moonboard website. This script is located at final_project/work/hold_curl.sh

Figure 1: Crowd Sourcing Platform



Building this website was a serious challenge for me and required lots of frustrating learning. Specifically, linking grid coordinate to hold image to spreadsheet cell was not a trivial process. This ultimately lead to the messy long strings in lines 123 -126 of index.html. It is worth mentioning that all of these strings were automatically written by a python script see final_project/work/string_conversions.py. However, because this whole process was initially so confusing, my code documentation is excellent. Writing extremely clear comments helped me not get lost in writing java script functions. I estimate that this component of the project took me over 30 hours. I choose to design my own platform as opposed to using mechanical turk or a similar surface for a number of reasons. I did not want to pay, and this classification task required expertise in the subject. Workers completed this task on a volunteer basis and were approached from the Metro Rock Climbing Gym. The sourced labels were then downloaded in csv form and analyzed in expectation_max.py. This script saves another file, estimated_holds.json that is a dictionary of hold id and hold type key value pairs.

Data Organization and Feature Engineering

Once I had all three components of the data, I needed to come up with a good way to aggregate everything together into one data structure for analysis. I choose store all data in python dictionaries that could easily be written to JSON files. This data structure suited the tree like structure of the data better than a txt file or csv other alternative. The data aggregation process is performed and documented in final_project/work/data_cleaning.py.

The script is organized in a modular way and produces a final JSON file route_corpus.json. This file contains all climb and hold information and houses the data analysed in final_project/work/exploration.py.

One confusing component of this process is the number of name conversion dictionaries. I work between a few different ways of specifying and computing with a given climbing hold. Had I altered the incoming data, this would be more straightforward to someone else reading the script. However, I actually found an error present in the way that the moon board website differentiates climbs. There was a route leaked from an alternative climbing wall set up that I had to delete from the corpus. The original grid coordinate system uses numeric and alphabetic axes. I transformed these to calculate the relative position of every hold on the board. See lines 122 to 142 on data_cleaning.py. The final distances are reported in metric cm to be consistent with the hold depths. Finally, edge depths and hold classes are added to each route in the route corpus. These are assigned in lines 145 - 175 . Much of this process could have happened simultaneously in one for loop as opposed to three or four separate loops. However I was adding different parts of the data over the course of the project and I preferred not to augment old code. There are also a number of try except states that I use to throw an exception due to a discontinuous list of hold ids associated with each hold. Further, the grades are transformed the IRCRA scale moving from (6b+ - 8A) in the french scale to (14 - 28) in the IRCRA scale.

Model Building

Several models were produced, compared and accessed for goodness of fit to determine the final model predicting dichotomous route quality. Different models on the same dataset are compared using the AIC. We first build a one vs. rest Logistic Regression model, dichotomizing the predicted variable, quality into 3 stars vs rest. We then look at potential non-linearity in the logit by considering different fractional polynomial transformations. We see that there are significant non linear transformations to continuous variables, Average Movement Distance and Average Usable Depth. A log transformation fits the movement distance while a -0.5 polynomial transformation fits the usable depth. When we compare the model containing the fractional polynomial transformations with the initial full model, we see that the initial model is a better fit due to a small AIC score.

Table 1: Full Logistic Regression Model

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-0.67	0.11	-6.22	0.00
Difficulty	0.16	0.001	24.62	0.00
Avg. Depth	-0.16	0.03	-5.50	0.00
Hold Type	-0.65	0.13	-4.73	0.00
Avg. Movement Dist.	0.85	0.01	1.76	0.08

Table 2: Full Logistic Regression Model with Transformations

	Estimate		Std. Error	z value	Pr(>|z|)
(Intercept)	-2.4661	0.2737	-9.01	< 0.001	
Avg. Movement Dist.*	0.1646	0.0565	2.91	0.0036	
Avg. Usable Depth**	1.4163	0.3051	4.64	< 0.001	
Mode Hold***	-0.6428	0.1376	-4.67	< 0.001	
Difficulty	0.1624	0.0067	24.35	0.0059	

*log transform, ** (-0.5 fractional polynomial transform),***(Pocket vs. Rest)

Project Scope Reflection

The initial impetus for the project was to conduct this analysis so that I could write an algorithm to suggest climbs on the moon-board app. I began to work on this as soon as a I had a very basic logistic regression model to satisfy my basic model building goals. However, the data acquisition process took much to long for me to put enough time into building the generative algorithm. I have a very naive version that samples from the quality distributions of each co-variate to select the next hold, but I was hoping to use a Markov chain model to produce this. The acquisition and analysis were the main goals of the project and the algorithm was a stretch goal. From the first check-in, I realized and reported that designing the generative algorithm would be a stretch in the given time frame. The time was mostly constrained by how long the web development took. I incorporated good scientific programming practices by using good naming conventions for variables and creating well commented code.