Predicting Washington Hiking Trail Trip Reports and Ratings

Metis Project #2 (Luther)

Kristin Mussar

January 25, 2019

Motivation:

Hiking is one of the most popular summer activities for Seattleites, but the trails are increasingly experiencing more traffic. Due to the difficulty of finding a trail that is not crowded and one which has parking available, I set out to build a tool that would allow users to see a of the predicted foot traffic for a given trailhead and day. The goal was that users could use this tool to plan their trips accordingly, whether that meant choosing a different trail, a different day to hike, or a different time to arrive. This data could also be used by urban planners to optimize new bus routes to popular trails in the area.

Unfortunately, the data that I gathered for this project did not work well with linear modeling and I switched to a new target – predicting trail ratings based on hike attributes. Predictions for trail ratings could be used by park planners to introduce new hikes to the area – whether they are looking to design popular trails or to spread out the traffic to each trail in their park.

As I spent considerable time on both approaches, I will describe them both.

Approach #1: Number of Trip Reports

Design:

My initial design was to investigate the trip reports for trails in the Olympic National Park as the National Park website lists information about the number of trail users in each area of the park per month (link). I planned to connect the monthly number of trail users with trip report statistics to give a better real-world estimate of how many hikers to anticipate at the trails. However, trip reports are sparse in the Olympic National Park and my target variable signal was week.

To get a more robust number of trip reports to work with, I then scraped information from Rattlesnake Ledge, one of the most popular trails in the area. If my approach worked on this dataset, I would expand to other trails in the area. Unfortunately, the number of trip reports per day was still a weak signal. Furthermore, the distribution of my data was Poisson, which is not ideal for a linear model. I aggregated the # of trip reports per week, as suggested by Chad, but this did not drastically improve my distribution (Figure 1).

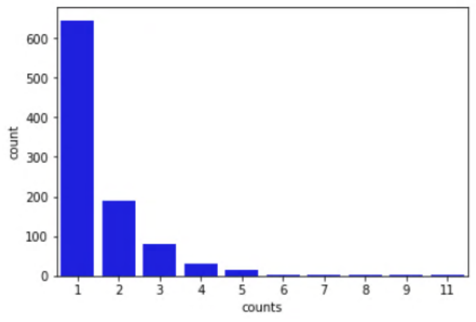

Figure 1: Distribution of trip reports for Rattlesnake Ledge aggregated by week from 2010 to present (Jan 2019).

Data:

- Information scraped from WTA:
  - For each trail I scraped the trail name, number of trail reports & trail rating. This information was used to then scrape each page of trip reports per trail.
  - For each report, I collected the report date, trail conditions, and URL. Reports went back to 1998.
- Weather data scraped from NowData (NOAA weather data):
  - Natalia Lichagina kindly shared her code with me for scraping this data. All code is hers (with minor tweaks). (Natalia's code is not on my github as I'm not sure how to properly give her credit for it on my repo).
  - Information collected per day included: max temp, min temp, avg temp, departure (difference from avg), HDD – heating degree days, CDD – cooling degree days, precipitation, new snow, snow depth.
  - Collected weather information for 3 stations – Seattle Area, Cedar Lake (near Rattlesnake Ledge), and Quillayute (in the Olympic Peninsula) as I wanted to see which would have a higher correlation with # of trip reports – local weather or Seattle weather.

Cleaning and Feature Engineering:

- Split date of hike into 3 columns for year, month and day. Create a column of the date formatted to match that of the weather data (index to merge on). Also convert the date to a datetime object, find the day of week, create new columns for day of week and "is_weekend" – True if it was a Saturday or Sunday.

Algorithm:

- Performed basic linear regression (train/val/test split) with scikit learn on my 19 numerical features to get an idea of how my MVP was performing. My R^2 value came out at 0.13.
- Plotted residuals vs predictions – no clear patterns emerged.

Jupyter Notebooks:

https://github.com/kmussar/metis_project_2/tree/predict_number_of_trip_reports

Conclusions:

After talking with both of you (Chad and Cliff), I decided to switch my target variable to trip ratings which had a better signal (across all trails) and a more normal distribution.

Approach #2 – Rating of the Trail

Design:

I switched my approach and instead collected information about each trail for all 3,604 trails on the WTA website. My distribution of ratings was approximately normal, which seemed much better to work with for linear regression (Figure 2).

.

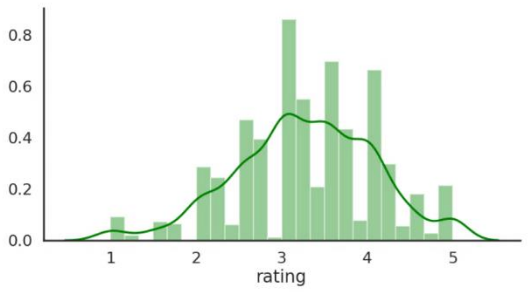

Figure 2. Distribution of trail ratings for 2,916 hikes.

Data:

- Information scraped from WTA:
    - I scraped the URLS for each trail listed on the website, which I then used to run through each listing to scrape the following trail attributes.
    - For each of 3,604 trail pages, I scraped the trail name, region, elevation gain, highest point, rating of the trail, number of ratings, number of trip reports, distance, trail attributes, permits, and GPS coordinates.

Cleaning and Feature Engineering:

- Filtered data to only include trails with ratings that were 20 miles or less (considered “day hikes”). – Left with 2,916 hikes.
- Created dummy variables for regions, permits, and trail attributes.
- Transformed number of trip reports and number of ratings to log scale

Algorithm:

- Basic Linear regression using scikit learn revealed an R^2 score of 0.03 (0.01 with cross-validation). As this indicated very little predictive power, I evaluated the residuals for my predictions, to see if there were any patterns that I could pull out with transformations (Figure 3). I did not see any patterns in these residuals.

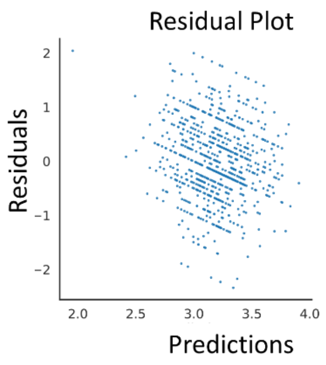

Figure 3. Predictions vs. residuals for basic linear regression (no transformations)

- I next looked at the residuals for each feature and found that two of them would benefit from log transformation (number of trip reports and number of ratings). Running basic linear regression on these transformed features increased my R^2 value slightly from 0.03 to 0.08.
- My exploratory data analysis had shown me that there were no strong correlations between my features and my target value, so a low R^2 value was not surprising. However, I wanted to test that I was not overfitting my model in my training set. I ran a ridge regression to correct for potential over-fitting to see if my scores improved. They did not as my R^2 from this model was -0.01 (0.07 with ridge CV). (I used StandardScalar with Ridge Regression and Ridge CV Regression to normalize the scales of my features).
- The negative R^2 value I can explain as my model was performing worse than a straight line or intercept. It had more variance than a straight line would.
- I also ran a learning curve (performance of model on training data vs performance of model on validation data) to see if my model was not converging on the validation set (Figure 4). (This is one reason to see negative R^2 scores). This was not the case, however.
- I then tried adding and removing features to see if I could improve my model, but this did not help either.

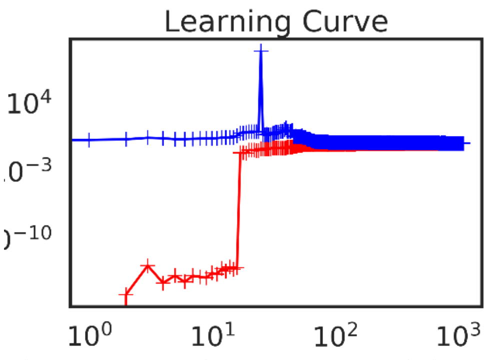

Figure 4. Learning curve on training set (blue) and validation set (red).

Conclusions:
From this, I concluded that I do not have enough, or the correct, information to make any predictions about trail ratings. Surprisingly, trails with certain attributes (such as waterfalls) do not consistently have significantly higher ratings than those without. Likewise, the region that the trail is in (such as national parks) does not seem to significantly impact the rating of a trail. This may be because the textual information on a trail website does not capture the most important factor of rating a trail – what the view looks like. It is also important to note that ratings on websites are inherently subjective and must be taken with a grain of salt. It would be interesting to run this same analysis using ratings from hiking books where trail scores are more objective and consistent. I would also love to include data on the traffic near trails, parking statistics, and trailhead log counts to more accurately predict how popular trails are.

Jupyter Notebooks:
https://github.com/kmussar/metis_project_2/tree/predict_ratings

Applicable to Both:
Tools:
- Selenium
- Beautiful Soup
- Jupyter Notebooks
- Pandas
- Pickles
- Seaborn
- Scikit learn
- Powerpoint * please note that some of my slides may appear cluttered or with images and text on top of each other due to the animations.