CS5800
Final Project Report
Trail Recommender
Matthew Quaglia, Wudassie Walle, Fabian Gaziano, Briggs Twitchell
4/3/2023

Finding the Best Trail:

A Recommender System for Hikers using Gower Distance & Merge Sort

Introduction

Hiking is an increasingly popular outdoor activity that offers a range of benefits for both physical and mental health. However, selecting the right trail can be challenging, especially for new or unfamiliar hikers. When choosing a trail, many factors must be considered, such as length, difficulty level, and scenic views. Additionally, hikers may have unique preferences and constraints, such as desired distance, preferred difficulty level, and terrain type. Therefore, a personalized trail recommendation system that considers these factors and preferences can significantly enhance the hiking experience by providing optimal and enjoyable trails.

This project aims to develop a trail recommender system using a merge sort sorting algorithm and Gower distance to provide personalized and efficient trail recommendations. We use sorting algorithms to sort the trails based on user preferences, such as preferred distance within a given radius and numeric features, such as geolocation, popularity, elevation gain, difficulty rating, and average rating. In addition to this, the trail recommender system recommends hikes based on categorical features such as route type, features like dog friendliness, children friendliness, sceneries, etc., and in addition to this, activates such as birding, camping, hiking, mountain-biking, etc. With the help of merge sort algorithms and Gower distance, we aim to determine the optimal trails that meet the user's preferences. This system aims to provide hikers with personalized and efficient recommendations that optimize enjoyment and efficiency, ultimately improving the hiking experience.

CS5800
Final Project Report
Trail Recommender
Matthew Quaglia, Wudassie Walle, Fabian Gaziano, Briggs Twitchell
4/3/2023

Context

Fabian Gaziano
Trail hiking induces mixed feelings in many people. In some cases people view it as a lot of hard work for little pay off, while others view it as an amazing way to get in touch with nature. However, in many cases finding a suitable trail becomes a difficult decision between many variables, often deterring those who are close to losing motivation to enjoy hiking. On a personal note I had very little appreciation for hiking, until I had a very positive experience and began to rethink my view on hiking. This hike presented ideal conditions including, temperate arid climate, beautiful views, and minimal interaction with harmful (with respect to the hiker) wildlife, while also presenting an achievable challenge that kept me engaged. A couple years later I experienced the opposite that brought me to question the potential enjoyment of hiking. An application that can find the optimal conditions for a hike would minimize the potential for experiencing a bad hike, which can simultaneously increase safety and enjoyment.

Briggs Twitchell
Today, algorithms that curate content relevant to a user are a staple of the internet. We encounter these algorithms in social media, entertainment, and online shopping. We'd like to use this same approach in the context of outdoor recreation. Our project seeks to create an algorithm that recommends trails based on a user's hiking preferences. Many hikers have something specific in mind when they search the internet for nearby hiking trails. They may be looking for something light and easy, dog-friendly, and short. Or they may want to find something remote that's quite steep and difficult. Additionally, first responders rescuing an injured hiker may want to see a list of the safest trails within a trail network. A reliable trail recommender would be helpful in each of these cases.

This project is personally relevant to me because I'm an avid hiker. I founded the Roux Outdoor Club, which is a student interest group at Northeastern's Portland, ME campus. I moved to Maine recently and therefore don't know all the trails in the region very well. Apps such as AllTrails are helpful in searching for trails, but they lack a recommendation system that links a trail's features to my own preferences. I'd like to help make outdoor exploration as seamless as possible, enabling more people to engage and connect with the natural world.

Matthew Quaglia
Roughly ten years ago, I moved to western Massachusetts to begin my undergraduate degree. An unexpected outcome of this move was that hiking became a huge part of my life, with much closer proximity to the mountains. From then on, I have spent a great deal of my weekends finding new places to hike, and that search often comes with finding a hike that's within a certain range of miles, is in line with my skill level, has a rewarding view at the end, and is relatively safe in terms of weather and trail conditions. As I'm currently completing a co-op experience out in Salt Lake City, I am particularly

interested in applications that can help me find hikes that fit my needs and can inform me about what conditions can be expected, as I have little knowledge about the trails in Utah. It would feel incredibly rewarding to participate in a project that helps myself as well as fellow hikers plan our trips with a little more ease and confidence.

Wudassie Walle
Hiking and trail walking is a popular recreational activity worldwide, and many people often seek a recommendation on the best trails to explore. I am a toddler mom, and trail walking with a toddler can be both fun and challenging. When going on a trail walk with my son, I always choose a trail suitable for his age and abilities; most importantly, safety comes first.

A trail recommender system that can provide personalized recommendations based on the user's preferences, activity level, and local weather conditions can offer several benefits, including improved user experience, increased engagement, and enhanced safety. Many trail recommender systems use a one-size-fits-all approach, recommending the same trail to all users without considering their preferences and abilities. With the advancements in technology and the increased availability of data, it is now possible to develop a trail recommender system to suggest the best hiking trails for people based on their preferences and interests.

Question

The main question that this project seeks to answer is: How can we develop an effective trail recommender system that provides personalized recommendations for hiking trails based on user preference and interests?

In addition, the project will try to answer the following sub-questions.

1) What are the relevant data sources that we can leverage to build a trail recommender system? This includes trail information, user data, and reviews, among other sources.
2) How can we preprocess and analyze this data to extract relevant features and insights that will inform the development of the recommendation engine?
3) What algorithms and techniques can we use to build an accurate and effective recommendation engine?
4) How can we continually improve the accuracy and effectiveness of the recommendation engine over time? This includes incorporating user feedback, monitoring user behavior, and regularly updating the system with new data.

Overall, the project seeks to address these sub-questions and develop a comprehensive trail recommender system that can provide personalized recommendations for hiking trails based on user preferences and interests.

CS5800
Final Project Report
Trail Recommender
Matthew Quaglia, Wudassie Walle, Fabian Gaziano, Briggs Twitchell
4/3/2023

Methodology

1. Problem definition

The problem addressed by this project is the development of a trail recommendation system that suggests trails to users based on their geographic location and trail preferences. The system uses a dataset of trails sourced from Kaggle and employs preprocessing techniques such as data cleaning, binarization, and adjustment of average ratings based on the number of reviews. The system then filters trails based on a user's location and a distance threshold and uses the Gower distance metric to measure the similarity between the user's trail preference and all the trails in the dataset. Finally, a merge sort algorithm is used to sort the resulting data frame based on the proximity of the desired location. The objective of this project is to develop an efficient and accurate trail recommendation system that assists users in finding trails that suit their preferences and location.

2. Data collection and preparation

The dataset used for this project is sourced from Kaggle's National Parks dataset, exported as a CSV file. The Python program used the Pandas library to read the CSV file and perform several preprocessing steps. First, it converted strings containing lists to actual lists to make them easier to work with. Next, it binarized some categorical data using the MultiLabelBinarizer and OneHotEncoder methods from the scikit-learn library to transform them into a format that machine learning algorithms can understand. The program also dropped duplicates and null values to ensure the data was clean and reliable for analysis. Finally, the program adjusted the average rating of trails based on the number of reviews to account for the fact that some trails may have a higher or lower average rating simply because they have more or fewer reviews.

3. Design and Development

The preprocessed data of trails were filtered based on a user's geographic location and distance threshold. The process involved calculating the distance between the user's location and the location of each trail using the Haversine distance formula provided by the geopy library. The distances were then stored in a separate column in the data frame, and the filtered trails were identified based on a user-defined distance threshold.

The preprocessed data of trails were filtered based on a user's geographic location and a distance threshold by using the following steps:

1. ***User input:*** The user was prompted to input their geographic location and a distance threshold in miles.
2. ***Distance calculation:*** Trail distance from a user is calculated using geodesic distances, which uses a spherical model of the earth to determine the shortest distance on the surface of an ellipsoidal model (source). We use the World Geodetic System (WGS84) as our coordinate reference system. Geolocation could be obtained either by their device location or by some other input such as zip code.
3. ***Distance storage:*** The distances calculated for each trail were stored in a separate column in the data frame.
4. ***Filtering:*** The filtered trails were identified by selecting only the trails with distances less than or equal to the user-defined distance threshold. If a user also selects trail length in their set of preference considerations, the database will be filtered again to exclude trails that are more than 5x the user's desired trail length. This enables the min-max normalization in the Gower distance function to behave more reasonably, reducing the effect of trails with outlier trail lengths.

The Gower distance is a distance metric that can handle mixed data types, including numeric, categorical, and ordinal data. It considers the different data types and scales in a dataset and calculates the distance between two observations as a weighted sum of the individual feature distances. Below is the formula for the Gower distance function (source):

$$S_{\text{Gower}}(x_i, x_j) = \frac{\sum_{k=1}^{p} s_{ijk}\delta_{ijk}}{\sum_{k=1}^{p} \delta_{ijk}}$$

In the case of our trail recommendation system, we have different types of features such as trail difficulty, length, type of trail, and location. These features have different scales and data types, which makes it challenging to use a traditional distance metric such as Euclidean distance. This is where the Gower distance comes in handy, as it can handle different data types and scales, allowing us to calculate the distance between the user's trail preference and all the trails in the data frame accurately.

Once the Gower distance is applied to the trails data frame, a distance matrix can be achieved, where each element represents the distance between two observations in the data frame. Then the distance matrix is used to calculate the distance between the user's trail preference and all the trails in the data frame. The resulting distances are then appended as a new column to the original data frame, allowing us to measure the similarity between the user's trail preference and each trail.

The identification of trails that are most similar to the user's trail preference and the recommendation of those trails to the user is achieved by interpreting the distance as a measure of similarity. The Gower distance is regarded as a powerful tool for trail recommendation systems, as it permits accurate measurement of similarity between different trails, enabling recommendations based on the user's preferences.

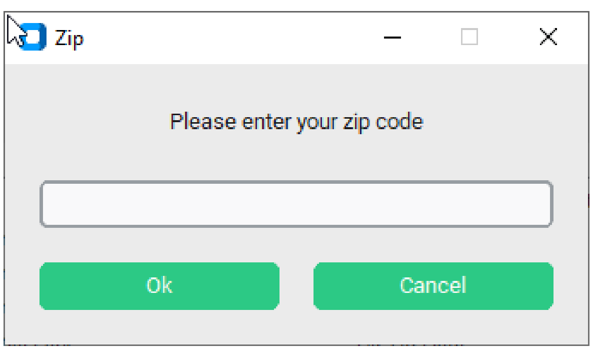
Our specific use of the Gower distance function only penalizes trails (i.e. computes a lesser degree of similarity) that don't contain a feature specified by the user. If, for example, the user just wants their trail to be dog friendly, then only trails without the dog-friendly tag will be penalized. Trails that have features that the user did not ask for will not be penalized. For numerical values, the algorithm uses min-max normalization to compute a distance value from the user's input. This value will be between 0 and 1. The closer the numerical value is to the user's input, the closer it will be to 0.

Once the degree of similarity is calculated, a merge sort algorithm for sorting comes into play for sorting the data frame based on the proximity of the desired location. The merge sort algorithm is a recursive divide-and-conquer sorting algorithm that operates by dividing an input list into two halves, recursively sorting each half, and then merging the sorted halves back together. This algorithm has a time complexity of O(n log n), which makes it one of the most efficient sorting algorithms for large datasets.

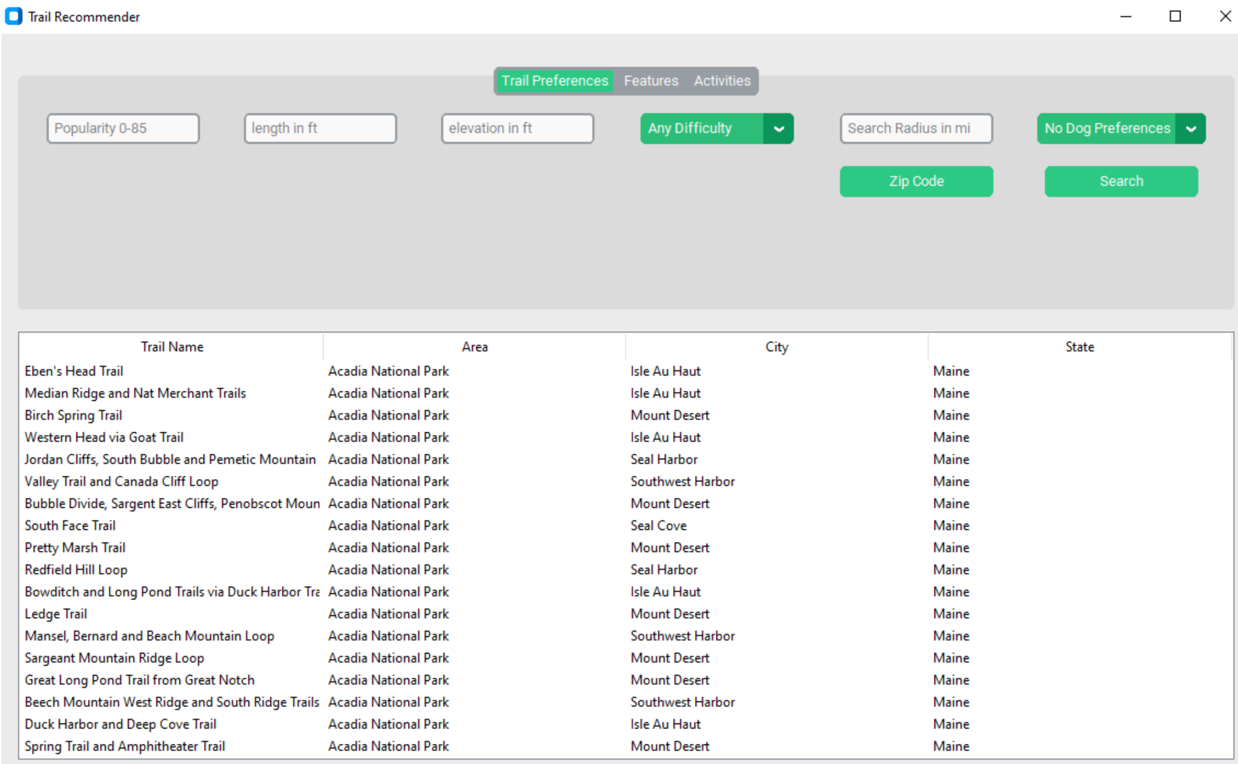
4. Implementation and Deployment

We combined our design above with a simple user interface, relying on the Tkinter API, to display the user preference options and output the recommended trail list. A user toggles, selects, and inputs trail features into their search, which then runs our algorithm and returns back the list of recommended trails:

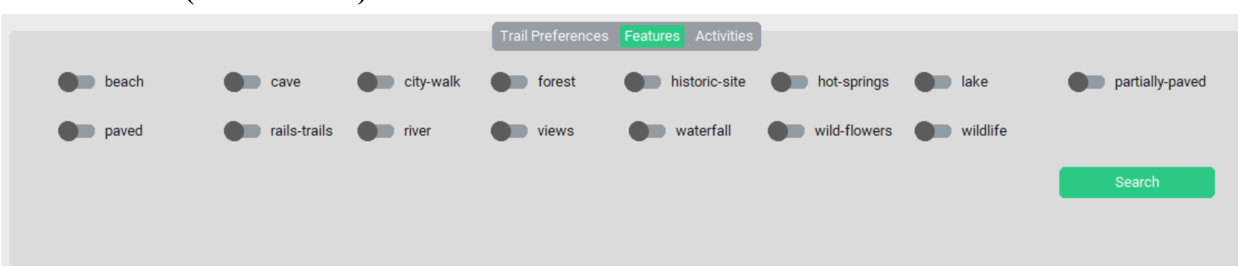
Opening window(Zip Code):



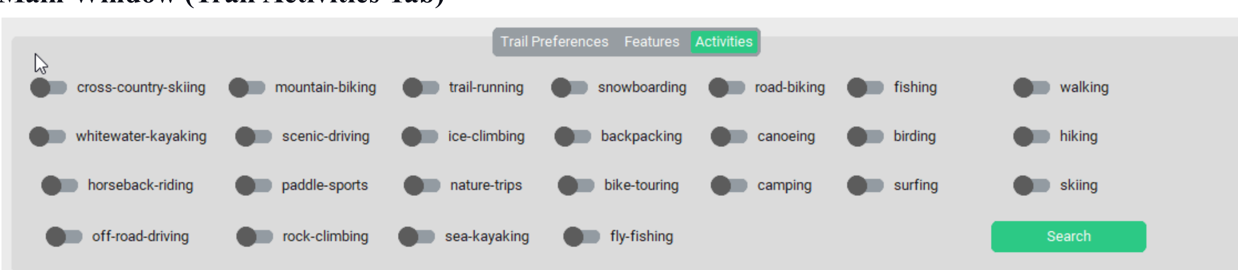
Main Window (Trail Preferences Tab & results)



Main Window (Features Tab)



Main Window (Trail Activities Tab)



We created this implementation simply to demonstrate the algorithm's capabilities. The algorithm's best use would be its integration into an outdoor exploration application, such as AllTrails or TrailForks.

5. Monitoring and Maintenance

We did not perform much monitoring or maintenance because the scope of this project only extended to creating a minimally viable program to recommend trails. In a commercial deployment of this technology, the following would be necessary to consider in scaling up the program to production level:

- Obtaining and formatting additional trail network datasets
- Adjusting the weighted significance of user preference inputs based on user feedback
- Determining the most relevant trail information to present back to the user once search results have been returned
- Potential effect of recommendations on the local area
- Tracking of user trail use over time to create recommendations based on inferred preferences

CS5800
Final Project Report
Trail Recommender
Matthew Quaglia, Wudassie Walle, Fabian Gaziano, Briggs Twitchell
4/3/2023

Conclusion

Answer to Question

Our question boiled down to, how can we implement one or more algorithms in the form of an application that will give users the ability to find the best trails for them by simply selecting their personal needs and preferences. We hoped to eliminate the need for the user to scour the internet to find a trail that suits their needs. We “answered” this question by creating an application that implements merge sort and Gower distance to efficiently tell users what trails are ideal for them. We were able to combine these methods with a user-friendly interface, with the help of the Tkinter package, to intuitively guide the user through the process, saving them time and stress while providing them with accurate information.

Weaknesses and Limitations

With most applications in the early phases of development, there are some limitations that the trail recommender system currently faces. For one, it only relies on one data set, affiliated with the National Park Service. Therefore, the user may not discover optimal trails for them, simply due to the fact that these trails, potentially state parks, city greenspaces, etc, are not being taken into consideration. Another limitation is that we are restricted to the features and categories in the dataset. There may be some users who wish to receive recommendations based on specific features that are not currently included in the recommendation system. A third limitation we faced is the timeframe we had to work on this project. With more time, we can modify the user interface, add additional features, and greatly expand the locations being considered.

Avenues for Further Research

We could extend our application by finding ways to implement various datasets, rather than simply the sole dataset we currently use. Some datasets may have different features, may be organized differently, etc, so an avenue for research would be the potential to automatically identify basic structures of the data set, and clean, organize, and filter the dataset accordingly. For example, the dataset’s features and activities columns were lists of the values for each trail, and we had to figure out a way to store them into individual columns based on the variables being considered.

Another avenue of further research would include user feedback and what the users feel will best meet their needs. This might include the filters we offer the user in specifying their trail preferences. There may be desired filters that are not currently included in the application, such as trails that have boulders along the way or at the end of the trail that rock climbers may be interested in. In this scenario, we might consider adding a ‘climber friendly’ option. As we find out what the users would like to see added to the filters section, we can implement a more well-rounded and versatile application.

Another way user data and user feedback could impact how we update and maintain our application is by finding out what areas need more trail options, what data doesn't appear to contain accurate information, and what geographic areas the application could be expanded too, especially outside of the US. As we expand geographically, we would need to continue doing research on what features may become important in these different areas, especially based on what types of hikes are available and what type of ecosystem the hikes are in.

Group Member Conclusions

Briggs Twitchell

This project was a good opportunity to practice my Python programming skills. My contribution primarily entailed prepping the dataset, creating the filter function, and creating the degree-of-similarity calculation. I learned how to apply a distance function, in this case the Gower distance function, to a real-world problem. Seeing the distance calculation work on data I had become familiar with was very clarifying in understanding how the formula works. I am very satisfied with our work and final product. Our team worked very well together, and I learned a lot from my fellow teammates. I will be proud to put this project on my resume. I'm passionate about this topic, and it was a very good team effort.

Wudassie Walle

I have learned that working on a group project is a fantastic experience that provides collaboration, diversity, accountability, learning, and support. It allows group members to create a comprehensive, innovative solution considering different perspectives. Being accountable for my contribution motivated me to meet deadlines and stay on track. My team members' diverse skills and experiences provided knowledge-sharing and learning opportunities, while their support helped me overcome challenges. Through this project, I also gained problem-solving, programming, data analysis, collaboration, teamwork, and project management skills that apply to different fields and industries. Participating in this project in a group was a valuable learning experience that helped me to achieve more than I could have done alone.

Matthew Quaglia

While working on this project, I have learned more about how to apply sorting algorithms, and specifically merge sort, to real-world problems. This project has allowed me to view these concepts in a different way that will benefit me long-term in my professional career. I also had the opportunity to learn a bit about how UI works while assisting in visualizing the results for our user interface. This project has also been a great experience in terms of working in a team, which involved a great deal of regularly communicating, sharing ideas, helping each other out with our respective expertise, and general collaboration to get all aspects of the project perfected. This project will certainly be of value to me in the future. It will be a great project to discuss during future interviews as well as a great reference point for projects I may have to work on in the future, both academically and professionally. It has been a great experience combining my passion for hiking with such crucial topics in algorithms.

Fabian Gaziano

This project offered a unique experience to create a practical application centered around an algorithm. Essentially working from the ground (Algorithm) up which required many interesting decisions to make the full application complete. A large part of my contribution was related to creating and fine tuning the user input side of the application, including adding the UI input widgets, designing the flow and organization of the UI, and connecting it to the inputs of the algorithm. This required extensive communication with team members and collectively finding creative solutions to complete our tasks and create a seamless product. I also worked on repository organization and management as well as UI polishing.Working with my group mates has been a genuine pleasure owing to everyone's flexibility, organization, and professional work. I will retain several takeaways such as algorithm application, a model of good group work, and a confidence I can apply these principles to other areas.

Appendix

Link to GitHub repository