Understanding and Predicting Trail Maintenance Needs Using Machine Learning Techniques

Data Science, Machine Learning

Luke Rand, Isaac Olson

Final Report

New Mexico Supercomputing Challenge 2024-2025
Santa Fe Preparatory School
Santa Fe, NM, US
March 30 2025

Contents

1 Introduction

1.1 Executive Summary

Countless Miles of Trail have been built across the United States. With limited funds and man hours to maintain those trails, improper allocation of resources harms the overall health of our trail systems and in turn limits community access to the outdoors, which is important to mental, physical, and emotional health. Our project achieves a method to better understand and standardize trail maintenance needs, allowing both large and small trail maintenance organizations to allocate funds and resources in a way that better maintains the health of the entire system.

This study used supervised machine learning to train a model to predict trail maintenance hours from environmental factors by providing a dataset of human determined maintenance. Geographic and maintenance hour data was obtained as a dependent variable from the Pacific Crest Trail Association, deemed the most efficient and leading group in trail maintenance due to the size of their organization and detailed reporting. Independent variables such as traffic, slope, and precipitation data were then correlated with maintenance hours in order to create the training data for a machine learning model. Before generating the model, multiple graphs were created to analyze to determine the model's accuracy. Lastly, the model was tested on a local trail.

This process determined that these environmental variables do influence trail maintenance needs, or at least how often the Pacific Crest Trail Association maintains the trail. Increased slope and traffic both increase maintenance, while increased precipitation decreases maintenance, likely because both slope and traffic encourage erosion, while precipitation discourages it by encouraging the fostering of healthy plant root systems. Still, the correlation between these variables and maintenance is existent, but somewhat weak. Thus, the model provides insight as a starting point, but is imperfect. However, the ability to estimated maintenance hours of a trail using a function that interfaces with our model provides an invaluable starting point for government and local trail organizations.

The model's limitations highlight that the Pacific Crest Trail Association does not determine maintenance data purely using the above variables, despite citing them as the most influential. Thus, the model could be improved with more perfect data, perhaps by analyzing a smaller acreage of trail more carefully and performing maintenance truly as needed, discovering additional independent variables, or working with the Pacific Crest Organization to improve both their methods and our model. However, despite the limitations of the model, the project is import framework which contributes to producing more efficient trail maintenance practices. Additionally, the ability to create a dataset for each of

these environmental variables at a latitude-longitude point is invaluable on its own to many of these organizations.

1.2 Problem Statement

Routine trail maintenance is necessary to maintain the health of our trail systems. Due to limited funding, limited resources are available to complete this maintenance and sustain or trail systems for the community. At the same time, physical activity improves mental and bodily health and fosters relationships between people and with the environment, and trails provide learning spaces for children, improving emotional, physical, and mental health. The importance of trails to a healthy community and world is undeniable. As such, it is important to properly allocate available funding and resources to maintain these systems. Currently, resource allocation decisions are made by humans, leaving some areas of trails prone to being under maintained or receiving more maintenance than necessary at the determent of other trails. Inefficient trail maintenance results in the United States Forest Service spending $80 Million yearly while still having a 157,000 mile maintenance backlog.[1] Our model seeks to streamline the trail maintenance process and ensure that all trails receive the necessary maintenance to be a sustainable resource to the community.

1.3 Tools Used

The model was developed using Python, a programming language known for its numerous libraries and readability. DynamoDB was chosen as the database software for its scalability and price. Git and GitHub streamlined the process of working in a team by allowing for version control and a remote repository.[2]

Essential libraries included `boto3`, a tool for interacting with the DynamoDB remote database, `threading`, a library for implementing multithreading, `scikit-learn`, a machine learning library, `matplotlib` for graphic representations, `PIL` for image handling in webscraping, and `pyppeteer` for webscraping. A full list of libraries used can be found on GitHub.

1.4 Current State of the Field

Currently both government and private trail maintenance organizations make decisions about trail maintenance allocation based on trail assessments by users and agency personnel. The United States Forests Service has a program, Trail Assessment and Condition Surveys (TRACS) that keeps track of trail condition based on surveys completed by personnel and trail crews. Although this program does help allocate trail maintenance, our research yields no organizations that use machine learning models to predict trail maintenance need. Our model is a first step in automating trail maintenance planning and has the potential

[1] Data from 2018
[2] GitHub linked in 9.7

to standardize and revolutionize the field in a way never done before. The application of machine learning in this field has the potential to democratize the trail maintenance process and ensure that all trails are maintained to the community standard at drastically decreased annual spending numbers.

2 Methodology

- First, extensive research was conducted in order to determine the factors that influence trail maintenance. Slope, weather, and traffic were determined to be the primary independent variables.
- Next, datasets from the Pacific Crest Trail Association containing information on trail maintenance projects with geographic data were used to generate a table of geographic points containing the hours of maintenance per mile per year at the location.
- Independent variables were taken from API's or web scraped at the location of each point and appended to the items of the table.
- Two dimensional graphs were constructed relating each independent variable to the maintenance value, and then linear regression was performed.
- Three dimensional graphs relating two independent variables to the dependent variable were constructed with multivariate linear regression.
- All five graphs were analyzed to determine whether patterns were present in the data and to understand the efficacy and accuracy of the linear regressions performed.
- A four dimensional model relating all three independent variables with the work on the trail was constructed to allow smaller or less efficient organizations to emulate the process of the Pacific Crest Trail Association to more effectively carry out maintenance.
- The model was further tested and verified using local trails in Santa Fe.

3 Research and Interviews

Initial research was conducted using Internet resources and reaching out to local and national trail maintenance organizations. Multiple organizations shared informative information about how they conduct trail maintenance and what factors they take into account when allocating their resources. These resources were valuable and were the basis for choosing trail slope, traffic, and precipitation as independent variables for the model.

The Pacific Crest Trail Association (PCTA) responded to our email and was kind enough to provide us with a large amount of trail maintenance data and allow us to conduct an interview. An extensive interview was conducted with Galen Keily, the Geographic Information System (GIS) Specialist at the PCTA. Mr. Keily explained how the PCTA trail maintenance data set was collected and laid out. He then outlined how the PCTA conducts trail maintenance with the goal of maintaining its trails in accordance with the PCTA Comprehensive Management Plan. Further insights provided by Mr. Keily and in this document around how the PCTA makes decisions around trail maintenance proved invaluable in understanding the outcomes of the model, and aided in determining the PCTA to be the leading organization in maintenance efficiency, and thus a strong provider of training data.[3]

4 Data

4.1 Database

Due to the relatively large size of the datasets that we use for this project, setting up a database was necessary to implement persistent and flexible storage. Additionally, a remote database reduced the size of the git repository and local storage requirements. With large amounts of data, this was a necessity. The DynamoDB NoSQL cloud database solution was elected due to being flexible, free, fast, and lending itself well to the project schema. A NoSQL database was selected for its scalability and dynamic nature, a requirement for a project where it was not certain from the beginning how many entries would be required and which additional fields may have became necessary. DynamoDB, like many other NoSQL databases, operates using keys which query a particular entry in a table and allows for reading of values from the entry.

Each initial primary dataset was uploaded to the database, and then operations were performed on these datasets to create new working tables.[4] In order to efficiently access entries in the database, a JavaScript Object Notation (JSON) file containing all the keys in an array was saved. This allows for local iteration through objects in order to make requests to the database. While saving these files required storage space, it was significantly less than locally storing the entire contents of the database.

4.2 Primary Datasets

The primary datasets used, which provided dependent variables and means to collect independent variables, were roughly 4 years of digital logs from the Pacific Crest Trail Association (PCTA). The PCTA manages 4265 kilometers

[3] Elaborated in 8
[4] Elaborated in 4.2

of nearly continuous trail along the western coast of the United States, covering varying environment and human conditions, including rain, snow, wildfire, mountainsides, bike traffic, and hikers. This data was provided by Galen Keily, the Geographic Information System (GIS) Specialist at the PCTA. Two datasets were made available.

4.2.1 Projects Dataset

The first dataset is a list of trail maintenance projects performed by the PCTA. Entries contain a unique ID and numerous fields. The fields applicable to the project are those that contain the number of total hours spent on the project, across all volunteers and staff members, titled "hours", and the date range for the project, titled "date". The entries begin in the middle of 2021 and extend through early 2025. This dataset contains 11742 items, although many of the entries do not contain corresponding geographic data, and thus were not used. The dataset was provided in ShapeFile format and was converted to JSON before iterating through the object and uploading entries to the remote table *(fig. 1)*.[5]

ID *(String)*	date	hours
a1i7V000004Dna4	7/13/2022 - 7/20/2022	405
a1iUS000000AdZZ	4/5/2024 - 4/5/2024	19.5
a1i5w000007HvJq	5/1/2021 - 5/5/2021	372

Figure 1: Selected items in the projects table.

4.2.2 Lines Dataset

The second dataset contains 1621 lines on a map of the world, denoted as a list of points in Mercator Web Projection (EPSG:3857),[6] and the project ID which each of these lines is associated with *(fig. 2)*. To upload to the database, the ID was selected as the primary key, while the list of points (line) was uploaded in json format as a string *(fig. 3)*. However, since multiple lines can share an ID for a project that spans multiple areas, each item contains an array of lines, and IDs with only one line contain an array with a single line.[7]

[5]"Table" will be used to refer to data on the remote database, while "dataset" will be used for other forms.

[6]Mercator Web Projection is a coordinate system that uses the distance in meters from the latitude-longitude coordinate of (0,0)

[7]Note that each line is additionally a two dimensional array, holding points with a latitude and longitude coordinate.

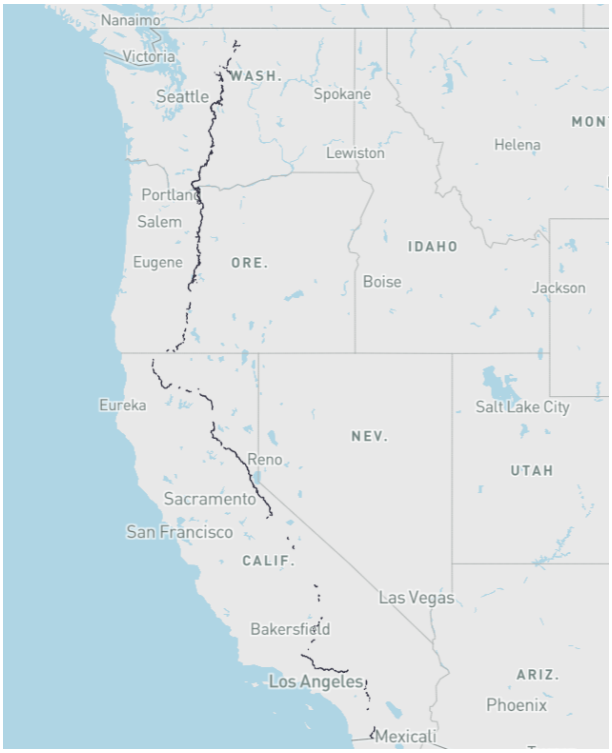


Figure 2: Project lines from the PCTA overlaid on a map.

ID - *Partition key*

a1i5w000007IwP0

points

[[[-13569194.9313, 5726321.7051], [-13569224.431, 5726319.634400003], [-13569230.2196, 5726323.138599999], [-13569237.121399999, 5726357.543200001], [-13569235.006299999, 5726405.964599997]], [[-13569207.955699999, 5726320.749399997], [-13569224.431, 5726319.634400003], [-13569230.2196, 5726323.138599999], [-13569234.0045, 5726342.0929000005]]]

Figure 3: Selected item in the lines table.

4.3 Creating Working Table

Using the data for any machine learning techniques or other analysis requires correlating the data with additional variables and creating a set of uniform items with fields to be correlated. The solution used was to create an additional table that used geographic points located on the PCTA as the primary key, and contained as values the hours spent per mile at the point for each year. Next, an average was calculated from the three years with complete data, 2022, 2023, and 2024, and appended as a value to the point. Secondary data was later added as additional values. Since the value for work done is in hours per mile per year, it is not necessary for the points to be equidistant, as the hours per mile in an area is intrinsic to a point.

This was accomplished by iterating through the list of ids and determining the hours spent on each project and the dates of work from the `projects` table. Next, the coordinates of the corresponding trail were queried from the `lines` table, and the length of trail worked on for the project was determined.[8] Next, each point from the line was assigned a value for hours per mile, calculated as $project\_hours/trail\_length$. The point was uploaded to the database, and the hours per mile value was appended to the point in a field titled as the year the project began, determined by parsing the date value. If a previous project had already updated the year field, the value was added to the existing value. This process was repeated for each project with associated lines, creating a table of 97,825 points. Next, an average was calculated using the three complete years. Finally, a field was added containing the latitude and longitude values in standard form for each point, converted from the previous Web Mercator Projection, to ease computation when correlating secondary data. Thus, the `points` table was fully prepared for independent variables *(fig. 4)*.

```
# Function to create the points table using the lines and projects table.
def create_points(ids): # pass in a list of project ids for indexing
    #iterate through each project
    for id in ids:
        #get project from database
        project = dynamodb.get_item(id) #function simplified for readability

        #get time (total hours by all people) and date
        time = project['hours']
        date = project['date']

        #get line from database
        line = dynamodb.get_item(id)
```

[8] Elaborated in 4.3.1

```
14
15          # get points
16          point_data = json.loads(line['points']) # needs to be
            ↪  loaded from a string
17
18          # lines is for distance calculation, points is just all
            ↪  the points involved
19          lines, points = combine_lines(point_data)
20
21          # get trail mileage
22          length = get_distance(lines)
23
24          # determine the year
25          date1,date2 = date.split(" - ")
26          date2=datetime.strptime(date2, date_format)
27          year=date2.year
28
29          # calculate hours per mile
30          hourspermile = time/length
31
32          # create or modify an entry for each point in the line
33          for point in points:
34              roundedpoint = [int(round(var, 0)) for var in point]
                ↪  #round the point so small changes in gps don't
                ↪  duplicate points
35
36              update_point(roundedpoint,hourspermile,year) # create
                ↪  or update the point in the database
```

point *(String)*	2021	2022	2023	2024	2025	avg	latlon
[-13236900, 4498584]				22.08983…		7.363277…	[37.42550625585259, -118.909095843…
[-13512701, 5989554]		10.75016…	4.757511…			5.169225…	[47.29009391971521, -121.386658380…
[-13557800, 5471932]	16.26259…			9.954065…		3.318021…	[44.041921980728425, -121.79178959…

Figure 4: Selected item in the points table before correlating independent variables. Hours represent the hours per mile at the location of the point.

4.3.1 Determining Mileage

In order to construct the points table and understand maintenance needs, trail length data was required. Calculating mileage of the lines dataset proved difficult, as multiple lines contributed to each project and often had overlap, yet did not share exact point values. This was remedied by combining lines where points were within one meter of each other using the following code, and finally calculating the entire distance of the lines associated with a project

using a simple multi-point distance algorithm. The unit of miles was chosen over kilometers because the outputs of the prediction model will likely be used primarily by United States residents outside of academia.

```
# Function to combine lines with tolerance
def combine_lines(lines):
    #familiar_points contains points seen before, and is used to generate the working table, while combined is the combined lines
    familiar_points = []
    combined = []

    # iterate through each of the lines in order to remove duplicated areas
    for line in lines:
        newline = []
        lastpoint = None
        for point in line:
            is_familiar = False #used to track familiarity of the point

            # if points are familiar, cut the line segment off, and restart when they stop being familiar
            for checkpoint in familiar_points:
                if close(point, checkpoint):
                    is_familiar = True
                    lastpoint = checkpoint
            if is_familiar:
                newline.append(point)
                if len(newline) > 1:
                    combined.append(newline)
                newline = []
            else:
                if len(newline) == 0 and lastpoint != None:
                    newline.append(lastpoint)
                newline.append(point)
                familiar_points.append(point)
                lastpoint = point

        # add the line to the combined set of lines
        if len(newline) > 1:
            combined.append(newline)

    # the returned values will be used to calculate mileage and create the working table
    return (combined,familiar_points)
```

4.4 Secondary Data

4.4.1 Precipitation

Precipitation data was collected using the Open Meteo Weather API, which provides historical weather data for the continental United States. Due to pricing barriers, weather data was collected using the last 90 days of data and yearly predictions were calculated, as earlier data is exclusive to a paid API key.[9] The API was accessed through the Python "requests" library. A python function, `get_average_rainfall(lat, lon)`, takes latitude and longitude as parameters and returns an estimated annual rainfall in millimeters. Another function calls `get_average_rainfall()` for each point in the `points` table, adding annual rainfall as an independent variable.

4.4.2 Slope

Slope data was obtained using the open Meteo Elevation API. This API provides elevation at a specific point with 90-meter resolution. Higher resolution elevation data is available from other APIs with a paid API key.[10] A python function, `find_slope(lat, lon)` takes 8 points in a circle with a radius of 100 meters around the given latitude and longitude and finds their elevation using the Python "requests" library. The function then takes the difference between the maximum and minimum elevations and the distance between the two points to calculate the slope of the ground surrounding the trail. Finally, the slope is converted to degrees using an inverse tangent formula.

4.4.3 Traffic

A relative unit of trail traffic is obtained via web scraping from an open source map project, freemap.sk. This map has a trail use heatmap layer that is scraped from trail use data on a popular athlete social media site called Strava. Trail traffic data is scraped from freemap.sk instead directly from the Strava heatmap due to a friendlier web scraping environment and background map layers that area easier to isolate from the heatmap data.[11] A Python function `trail_use_scraper` opens freemap.sk on chrome using the Python library "pyppeteer", a Python port to the Node.js library puppeteer. The function then calls another function `enter_coordinate()`. This function navigates to the entered coordinate and desired zoom level to take a screenshot *(fig. 5)*. It then isolates all pixels of the heatmap and calculates the average luminosity of the pixels, determining an indicative value for traffic. The returned value is representative of trail traffic because the method used effectively reverses the process used in the heatmap to visually represent trail use.

[9] Elaborated in 9.2.2
[10] Elaborated in 9.2.2
[11] Elaborated in 9.2.1

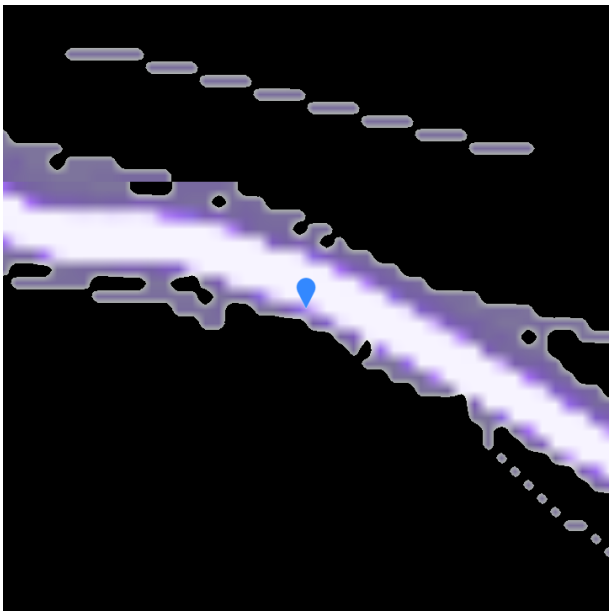


Figure 5: Example manipulated screenshot from webscraping for traffic determination.

Thus, the `points` table was finalized with independent variables and ready for analysis and constructing a model, with each point containing average hours per mile per year, latitude-longitude pairs in both Mercator Web Projection and standard form, annual precipitation, slope, and cumulative traffic *(fig. 6)*.

point *(String)*	avg	latlon	slope	traffic	weather
[-13236900, 4498584]	7.363277...	[37.425506...	16.172159...	12.681646...	1321.4759036144578
[-13512701, 5989554]	5.169225...	[47.290093...	20.052082...	11.760891...	1578.734939759036
[-13557800, 5471932]	3.318021...	[44.041921...	1.1457628...	12.994540...	2441.981927710843

Figure 6: Selected item in the points table after correlating independent variables. Yearly maintenance values are hidden for readability

4.5 Multithreading

Many of the operations performed in the generation of the `points` table involved a significant number of input-output requests to external databases or APIs. Additionally, other aspects of the project such as generating graphs and constructing the model involved many queries to the remote database. Thus, the runtime of many operations that were performed in relation to entire tables of multiple thousands of entries were limited not by compute speed, but rather

by internet latency. Thus, implementing multithreading across many parts of the code-base, including integration of all secondary variables, querying from the database, uploading to the database, and interfacing with APIs, reduced runtime drastically. Multithreading, in contrast to multiprocessing, does not utilize multiple processors, but rather rotates through many threads in quick succession on a single processor, creating the illusion of concurrency. This allows many input-output requests to be sent in quick succession without waiting for the previous request to complete.

5 Sub-segmenting Dataset

Many of the API's used had limitations on requests for the free tier of access.[12] Thus, we were forced to use only a sub-segment of the table to allow for integration of independent variables on all points of the dataset. 1000 points were randomly selected from the points dataset and used for correlation of secondary data, analysis, and construction of the model.

6 Machine Learning

Our data is visualized and the model is constructed using single variable and multivariate linear regression, a simple supervised machine learning technique. Supervised machine learning uses input data and correlated output data to develop a model that fits the provided data. Thus, if the input data follows a specific pattern, the machine can create predicted output data when given unfamiliar input data. Linear regression techniques develop a linear model that is designed to fit the data provided, and thus can predict outputs. Our dataset contains inputs of multiple variables and an output of maintenance hours per mile per year, making it a perfect candidate for machine learning to create a predictive model that can determine maintenance needed on unfamiliar data.

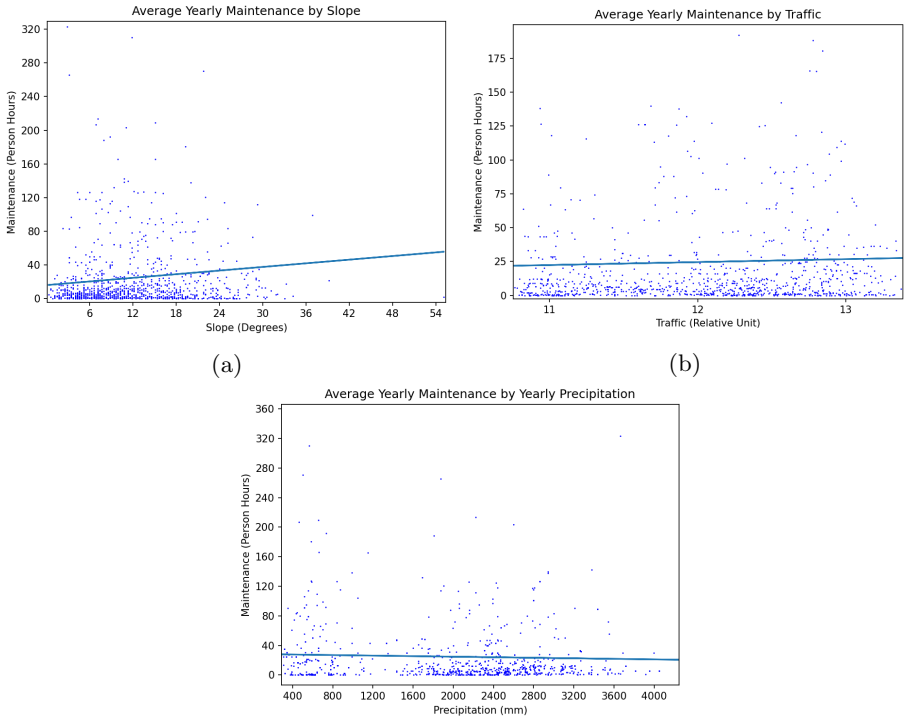
6.1 Linear Regression

To initially understand our data, three two dimensional graphs were produced, and linear regression was performed on the relationship between each of our secondary variables and the dependent variable of work hours per mile per year. This generated 3 graphs for analysis *(fig. 7)*.

6.2 Multivariate Linear Regression

To improve understanding of our data, graphs correlating two independent variables with work hours were produced. Thus, three graphs correlating each pair of independent variables with the dependent variable provide further insight into the working dataset.

[12] Elaborated in 9.2.2



(a) (b)

(c)

Figure 7: Graphs correlating independent variables with yearly maintenance requirements, with linear regression.

Multivariate linear regression was performed on each of these. In contrast with the single variable regressions, these models were developed using only 800 points of the 1000 point dataset, reserving the remaining 200 to understand the validity of the model. The graphs below show the prediction model overlaid with the 200 test points *(fig. 8)*.

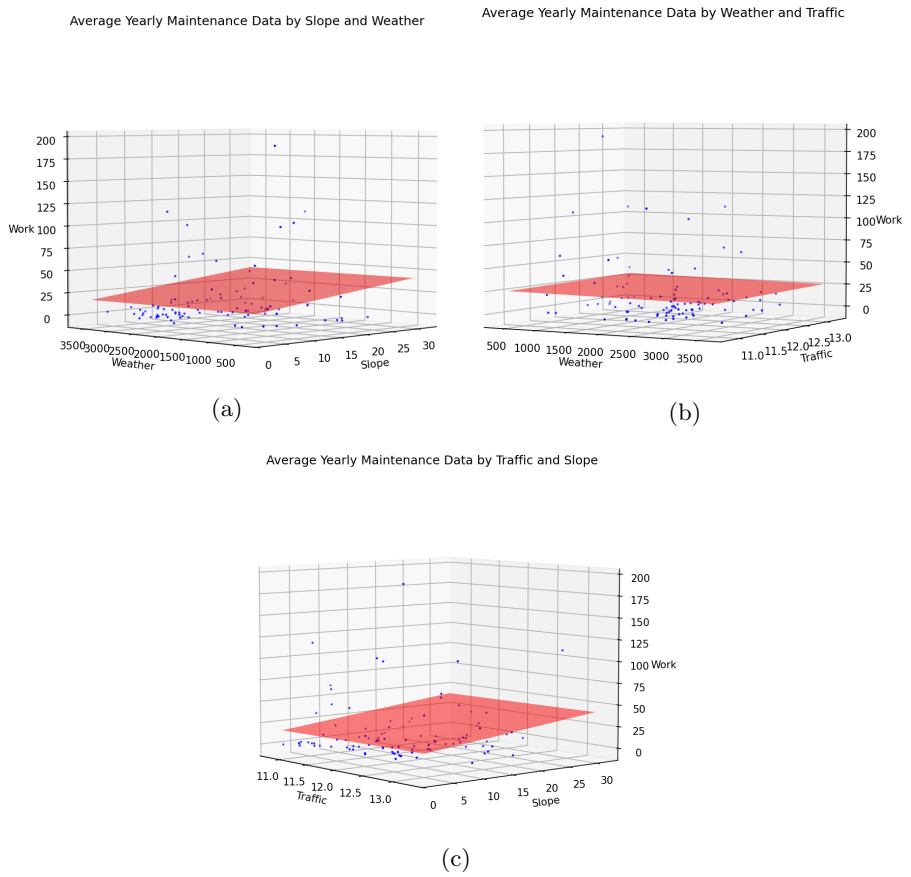


Figure 8: Graphs correlating two independent variables with yearly maintenance requirements, with multivariate linear regression.

6.3 Complete Model

Additionally, a linear regression model taking all three independent variables into account was produced, although it cannot be visually displayed and analyzed as a graph would occupy 4 dimensions. We can, however, understand its efficacy and reliability using the other graphs.

7 Analysis

Studying the first three graphs *(fig. 7)*, it is evident that both traffic and precipitation hold less significance than slope in determining the amount of maintenance performed by the PCTA.[13] Across the entire span of traffic data, from 10.82 to 13.33,[14] the projected change in yearly hours per mile is 5.444. Thus, traffic has some effect, with increased traffic implying increased maintenance. Across the spread of yearly precipitation, the projected hours decrease by 7.418, implying that precipitation causes slightly decreased trail maintenance hours. Slope appears the most significant influence on trail maintenance needs, with a positive difference of 27.52 hours per mile per year across the range of slopes. It is worth noting, however that slopes above roughly the halfway point are rare, so a spread of around 18 can be understood as more significant. It is additionally clear that none of the lines represent a strong fit, as many of the points are far from the line of best fit. This implies that the data does not hold a strong pattern.

The three dimensional graphs add little understanding, and share many of the same traits as the two dimensional graphs, including their lack of certainty due to poor fitting of the data. They do help us to understand that the highest projected maintenance occurs at low weather and high slope, low weather and high traffic, and high traffic and high slope, respectively, a fact that can be inferred from the two dimensional graphs.

The imprecision in the two dimensional and three dimensional regressions casts doubt on the validity of a predictive model constructed using this data. Thus, we can, at best, understand the four dimensional model constructed as a best guess or starting point that attempts to emulate the work of the PCTA.[15]

8 Results of the Model

8.1 Data Insights

Although the model is built on somewhat weak data, the model can still offer some interesting insights into trail maintenance necessity. Our model shows that more maintenance is needed at high slope, high traffic, and low weather area. An area with these characteristics is also one that is most likely to be susceptible to erosion. Large amounts of trail use can degrade trails overtime. Along with this, larger slopes create more damage from water runoff and, although it is counterintuitive, lower amounts of precipitation will likely increase erosion. Low

[13] It is important to note that this does not necessarily reflect optimal work hours, as will be elaborated in 9, 9.2.1 and 9.3

[14] These are relative values calculated from luminosity of heatmap pixels, and has no standalone meaning

[15] This uncertainty is a result of data, not procedure, as will be elaborated in 9

precipitation generally indicates lower vegetation density and thus less organic matter such as roots is present to hold together the topsoil. With no structure holding together the soil, major weather events will cause much more erosion.

Although we originally expected to see a larger correlation between trail traffic and maintenance need, the more uniform results offer an insight into the PCTA trail maintenance strategy. The slight upward trend in trail maintenance when traffic increases indicates that the PCTA spends slightly more time maintaining higher traffic areas but it also full fills its commitment to manage all 4265 kilometers of trail in its purview. A disproportionally large amount of work in only the high traffic areas could negatively affect the heath of the trail system in the long run.

8.2 Model Application

The model was applied to a trail in our local area, the Atalaya mountain trail *(fig. 9)*.

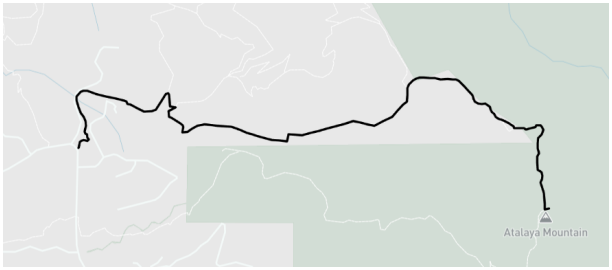


Figure 9: Atalaya trail Shapefile line overlaid on map.

The model was then applied to various points along the trail, which generated values of work hours per mile per year at each point. These values from along the length of the trail were averaged to get a final value of 27.31 hours per mile per year. Given a trail length of 2.2 miles, our model recommends that an organization maintaining this stretch of trail dedicates 60.09 work hours per year across all workers in order to maintain the trail in accordance with the trail maintenance standards outlined in the PCT comprehensive management plan. This comes out to roughly a standard day of trail work for a crew of 7 people.

```
#Return value for trail maintenance need in hours per mile per
↪  year at a specific point
def run_point(lat,lon):
    # download the presaved model
    with open('model.pkl', 'rb') as file:
        model = pickle.load(file)

```

```
7       # get the average precipitation at the point
8       weatherval = weather.get_average_rainfall(lat,lon)
9
10      #convert latlon to meters. This is necessary because the
        ↪  slope and traffic functions takes coordinates in Mercator
        ↪  format.
11      transformer = Transformer.from_crs("WGS84", "EPSG:3857")
12      lat,lon = transformer.transform(lat,lon)
13
14      #get slope at the point
15      slopeval = slope.find_slope(lat,lon)
16
17      # get the relative traffic value at the point. Called
        ↪  asyncronously
18      trafficval = asyncio.run(trailuse2.scrape_one(lat,lon))
19
20      # initialize the input data
21      ind = {"weather":[weatherval], "slope":[slopeval],
        ↪  "traffic":[trafficval]}
22      X = pd.DataFrame(ind)
23
24      #Run model using the values found for weather, slope, and
        ↪  traffic.
25      return model.predict(X)[0] #returns a single item array, so
        ↪  we take the first value
26
27  #This function takes an array of points along a trail and the
    ↪  length of trail in order to return suggested maintenance
    ↪  amount in # of work hours per year
28  def run_trail(trail_points, trail_miles):
29      #add together the values of hours per mile per year for each
        ↪  point
30      trail_vals = []
31      for point in trail_points:
32          #find value at each point
33          trail_vals.append(run_point(point))
34
35      #average values and return total number of hours
36      hours_per_mile_per_year = statistics.mean(trail_vals)
37      return(hours_per_mile_per_year*trail_miles)
```

9 Conclusion

9.1 Conclusions

Our model uses supervised machine learning to predict trail maintenance need based on training data from the Pacific Crest Trail Association and a variety of independent variables recommended by trail maintenance experts. The model provides a way for trail maintenance organizations to allocate their limited funding in a way that ensures all trails are up kept to a certain standard. This functionality offers exciting possibilities in trail maintenance and ensuring our trail remain a valuable resource for the community.

The data set for the model provides interesting insights into understanding the science of trial maintenance. The data collected demonstrates how the environmental variables that were recommended to us by experts in the field, do affect trail maintenance necessity. The correlations in the data between these environmental variables and trail maintenance need aligns with how experts predicted it would. Although the correlation between these environmental variables is undeniable, it is somewhat weak. This suggests that the PCTA does not only base its trail maintenance decisions on these variables. This limitation of the model is not based in its methodology, rather, it represents data set error. The same procedure could be expanded on with a stronger dataset to obtain more accurate results.[16]

Although the model is imperfect due the data it is built upon. It still offers exciting prospects for the field of trail maintenance. As demonstrated in 8.2 the model can easily and effectively be applied to trail systems around the country. If the model is applied to a complete trail system by an organization, it will completely revolutionize trail management. The ability to accurately predict trail maintenance need across an entire system will allow for easier resource budgeting and more effective resource allocation.

9.2 Limitations

9.2.1 Data Limitations

Data availability limitations somewhat handicapped the effectiveness of the model. The trail maintenance data provided by the Pacific Crest Trail Association (PCTA) is an incredible data set and invaluable to the functioning of the model. However, the PCTA has only been tracking their trail maintenance in this form for about 4 years so the data set is still relatively young. As data for more years becomes available the model will be able to produce more accurate insights into trail maintenance necessity.

[16] Expanded in 9.3

Our model also ran into limitations surrounding trail use data. As mentioned in 3.4.3, the trail use data originates from the athlete social media site Strava. Strava offers access to its raw trail use data through its program Strava Metro to urban planners, trail networks, and city governments to help improve mobility infrastructure. We reached out to Strava Metro in order to include their data in the model, however, Strava Metro denied our partnership request citing a large volume of requests and therefore only providing data to groups directly involved in active transportation infrastructure planning. Due to this limitation our model was forced to use a relative unit of trail use derived from heatmap web scraping. If a large trail maintenance organization such as the PCTA were to use our model in trail maintenance planning it is likely that Strava Metro would approve the partnership application and provide more accurate trail use data.

9.2.2 Financial Limitations

The model was forced to use less accurate approximations for both weather and slope data due to API paywalls. We were unwilling to pay for this data but a larger government or non profit institution using the model for trail maintenance planning would have the financial bandwidth to pay these relatively small fees that are used for API upkeep. An API key to the more accurate weather and slope data APIs only costs a total of 99 dollars per month putting the data well within reach of any organization planning on implementing the model for trail maintenance planning on a larger scale.

9.3 Solutions

As outlined above in 9, the correlation between the environmental variables and trail maintenance necessity is present but not perfect. This manifests in a less accurate model, however, the problem is not one of methodology and could be fixed with a stronger data set.

The weaker than expected correlation suggests a limit in the accuracy of the PCTA trail maintenance data. Although the PCTA is one of the leading trail maintenance organizations in the nation it is still probable that much of their trail maintenance resources are not allocated in the ideal way. One possible solution would be to work with the PCTA or a similar organization and segment off smaller sections of their trail to perform trail maintenance in that area truly as needed. This new data would serve as the ideal training data for our model and eliminate problems with the current trail maintenance dataset.

Strengthening the independent variable data would also improve the accuracy of the model. As outlined in 9.2.1, insufficiency with the current data set could be easily resolved by a larger organization using the model for trail maintenance planning. It is also possible that another interdependent variable needs to be

added to the model to improve its efficacy, which further research would aluminate. If the PCTA is making trail maintenance decisions based on an additional variable, this could throw off the model if is not included.

9.4 Foundation and Government Applicability

As mentioned in section 1.4, the United States Forest Service currently makes decisions around trail maintenance resource allocation using the Trail Assessment and Condition Surveys program (TRACS). The TRACS program currently is an organized framework for collecting data on trail conditions by agency personnel in order to better allocate resources towards trails in the worst state of disrepair. Although this program is has a large amount of value in ensuring trail health, it is purely a reactionary strategy that only responds to trail degradation when trail functionality is already limited. Our model has the potential to further develop this program by optimizing maintenance scheduling. The model will prioritize maintenance spending by predicting areas that are most likely to deteriorate soon, effectively switching the entire trail maintenance strategy from reactive to proactive.

The implementation of this model also has the potential to reduce the overall cost of trail maintenance. A more uniform and proactive approach to trail maintenance will allow personnel to identify and fix smaller problems with the trail before there are large problems that are costly to repair. This will also lead to a greater ability to predict trail maintenance costs giving managers a clearer picture of upcoming budgetary needs. This allows for better financial planning and reduces problems with overspending.

Additionally, an understanding of how the PCTA actually runs maintenance and how that interacts with environmental factors will allow them to improve their pipeline. Addressing these factors would not only improve PCTA maintenance, but allow for the training of a stronger model in future years, creating a feedback loop of improvement and benefiting the entire industry.

9.5 Acknowledgments

The Pacific Crest Trail Association made this project possible by kindly sharing the trail maintenance data that is the foundation of our project. We would like to specifically thank Galen Keily, the GIS specialist for the PCTA, for generously donating his time to explain the data set as well as allowing us to conduct an interview to better understand the trail maintenance process. Additionally, we would like to acknowledge our club sponsor Ms. Comstock for providing valuable guidance and a framework for achieving success. Lastly, we would like to thank the New Mexico Supercomputing Challenge for their feedback and aid across all stages of the project.

9.6 Generative AI Use

In the interest of using industry standards and utilizing available tools, the generative artificial intelligence Chat GPT was used in select cases to streamline the coding process and improve efficiency. Due to the limitations of the technology itself, it was used to write small snippets of code (database queries and API requests), or given pseudo code to transform for more simple functions whose work was heavily syntactical as opposed to logical (querying the weather API and calculating line distance). Lastly, in the interest of education and because Chat GPT is incapable of correctly programming complex functions, and often makes mistakes on simpler ones, the technology was utilized only for simple functions (those that could easily be logically planned and understood within seconds), and the human programmers carried out appropriate research and learning beforehand to fully understand the outputted code in order to debug. Essentially, Chat GPT was used to generate only code that the programmers could have written themselves with minimal cognitive stress, and was purely used to increase efficiency.

9.7 Additional Links

`https://github.com/lukerand/trails`

10 Works Cited

References

[1] "Benefits of Hiking." *National Park Service*, www.nps.gov/subjects/trails/benefits-of-hiking.htm. Accessed 19 Dec. 2024.

[2] Bevans, Rebecca. "Multiple Linear Regression — a Quick Guide (Examples)." *Scribbr*, 22 June 2023, www.scribbr.com/statistics/multiple-linear-regression.

[3] "Docs Open-Meteo.com." open-meteo.com/en/docs.

[4] "Elevation API Open-Meteo.com." open-meteo.com/en/docs/elevation-api.

[5] "Freemap Slovakia, Digital Map." *Freemap Slovakia*, www.freemap.sk.

[6] Kirvan, Paul. "Multithreading." *WhatIs*, 26 May 2022, www.techtarget.com/whatis/definition/multithreading.

[7] "Machine Learning, Explained — MIT Sloan." *MIT Sloan*, 21 Apr. 2021, mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained.

[8] Pacific Crest Trail Association. "Pacific Crest Trail Association - Preserving, Protecting and Promoting." *Pacific Crest Trail Association*, 20 Feb. 2018, www.pcta.org.

[9] Ray, Tyler. "Congress Passes Trail Inclusive Spending Bill." *American Hiking Society*, 6 Aug. 2021, americanhiking.org/congress-passes-trail-inclusive-spending-bill.

[10] Smallcombe, Mark. "SQL Vs NoSQL: 5 Critical Differences." *Integrate.io*, 15 Feb. 2024, www.integrate.io/blog/the-sql-vs-nosql-difference.

[11] "Trail Assessment and Condition Surveys (TRACS)." *US Forest Service* www.fs.usda.gov/managing-land/trails/trail-management-tools/tracs. Accessed 1 Apr. 2025.

[12] "COMPREHENSIVE MANAGEMENT PLAN for the PACIFIC CREST NATIONAL SCENIC TRAIL" *US Forest Service* www.fs.usda.gov/Internet/FSE_DOCUMENTS/stelprdb5311111.pdf. Accessed 2 Apr. 2025.

[13] "Why Trails Matter: Outdoor Learning." *American Trails*, www.americantrails.org/resources/why-trails-matter-outdoor-learning. Accessed 19 Dec. 2024.