Predictive Analysis of Hikes

BDAI Lab Project 2025

Lorenzo Dufour, 2133104

March 27, 2025

Introduction

The selected dataset is a collection of hikes from hikr.org, a platform where users can post reports about their hikes, scraped in Spring 2018 by "Rocco" and made available on Kaggle. The dataset contains 12,141 observations and 17 variables.

In this analysis, we will attempt to perform a predictive analysis of the hikes' duration based on the other variables in the dataset, and a classification analysis of the hikes' difficulty.

We will also perform a clustering analysis of the hikes' season.

Variables description

Variable name	Label
X_id	Unique identifier
length_3d	Hike length in meters considering elevation changes
user	Username of the hiker who recorded the hike
start_time	DateTime indicating when the hike started
max_elevation	Maximum altitude reached during the hike
bounds	Geographical coordinates defining the boundary of the hike region
uphill	Total elevation gained during the hike
moving_time	Duration of the hike in seconds, excluding breaks
end_time	DateTime indicating when the hike ended
max_speed	Maximum speed achieved during the hike
gpx	GPS Exchange Format file containing the hike's track, waypoints, and routes
difficulty	Hike difficulty rated on the SAC scale, from easiest T1 to most difficult T6
min_elevation	Minimum elevation reached during the hike
url	URL to the original post on hikr.org
downhill	Total elevation lost during the hike
name	Hike title, given by the user
length_2d	Hike length in meters without considering elevation changes

Data preparation

Import and formatting

Our first step is to validate that all variables are in the correct, most adapted format. We will use the `glimpse` function to get a summary of the dataset.

```
## Rows: 12,141
## Columns: 17
## $ X_id          <chr> "5afb229e8f80884aaad9c6ea", "5afb229e8f80884aaad9c6eb", ~
## $ length_3d     <dbl> 10832.953, 12259.376, 22980.168, 24903.503, 19581.274, 8~
## $ user          <chr> "Bergfritz", "Bergfritz", "igor", "rkroebl", "rkroebl", ~
## $ start_time    <chr> "2018-05-11 07:37:40", "2018-05-12 07:25:08", "2018-05-1~
## $ max_elevation <dbl> 1934.4700, 2186.2100, 2265.0000, 962.4200, 697.5700, 261~
## $ bounds        <chr> "{'min': {'type': 'Point', 'coordinates': [13.242523, 47~
## $ uphill        <dbl> 612.8800, 614.7530, 2255.9760, 882.3120, 310.6620, 922.8~
## $ moving_time   <dbl> 12155, 13876, 28971, 26726, 18197, 10905, 14660, 0, 0, 1~
## $ end_time      <chr> "2018-05-11 11:38:23", "2018-05-12 12:08:28", "2018-05-1~
## $ max_speed     <dbl> 1.595493, 1.394320, 1.503002, 1.516689, 1.542405, 3.8599~
## $ gpx           <chr> "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n<gpx xmlns:~
## $ difficulty    <chr> "T2 - Mountain hike", "T3 - Difficult Mountain hike", "T~
## $ min_elevation <dbl> 1322.9600, 1266.4000, 176.5400, 388.5100, 438.5000, 1685~
## $ url           <chr> "http://www.hikr.org/tour/post131855.html", "http://www.~
## $ downhill      <dbl> 609.6700, 1193.7330, 2177.6260, 901.0520, 305.3720, 927.~
## $ name          <chr> "Remsteinkopf, 1945 m", "Schuhflicker, 2214 m", " Cima d~
## $ length_2d     <dbl> 10832.953, 12259.376, 22980.168, 24903.503, 19581.274, 8~
```

Most variables are in the correct format, except for the `start_time` and `end_time` variables, which are in character format. The `difficulty` variable is also in character format, an ordered factor could be more appropriate if its unique values are consistent.

The `start_time` and `end_time` variables are in the format "YYYY-MM-DD HH:MM:SS". We will thus use the `lubridate` package to convert them to datetime format.

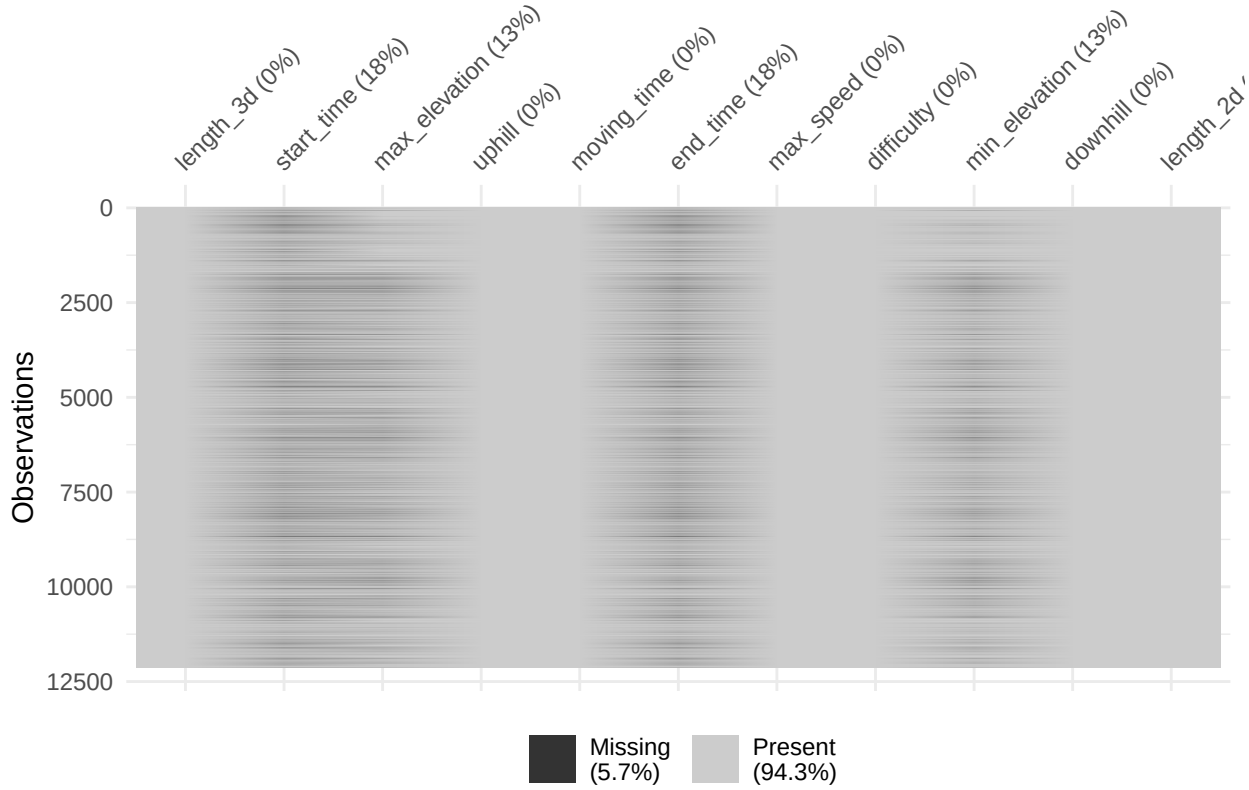
```
##  [1] "T2 - Mountain hike"
##  [2] "T3 - Difficult Mountain hike"
##  [3] "T1 - Valley hike"
##  [4] "T3+ - Difficult Mountain hike"
##  [5] "T4- - High-level Alpine hike"
##  [6] "T4 - High-level Alpine hike"
##  [7] "T5- - Challenging High-level Alpine hike"
##  [8] "T5+ - Challenging High-level Alpine hike"
##  [9] "T4+ - High-level Alpine hike"
## [10] "T6 - Difficult High-level Alpine hike"
## [11] "T5 - Challenging High-level Alpine hike"
## [12] "T6- - Difficult High-level Alpine hike"
## [13] "T6+ - Difficult High-level Alpine hike"
```

Using the base R `unique()` function, we see that the `difficulty` variable has 13 clean categories, fitted on the SAC alpine hiking scale. We can order them from easiest to most difficult using an ordered factor. We will also remove the hiking levels descriptions in the strings to make them more readable.

We also dropped columns that are not relevant for the analysis. These are the `X_id`, `bounds`, `gpx`, `url`, `name`, and `user` columns.

We can now proceed with the missing values management.

Missing values



Using the `naniar` package, we can visualize the missing values in the dataset. We see that the `min_elevation`, `max_elevation`, `start_time`, and `end_time` variables have missing values. Since these missing values represent more than 5% of the observations of each concerned column, we will not remove them. Instead, we will impute them with the most appropriate value of the respective column given its distribution skewness.

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## -32768.0    560.0    960.1   1003.3   1389.5   4180.0     1578
```

We can see in the hereabove summary that the `min_elevation` variable contains unrealistic negative values. We will impose missing values for these observations. Since negative elevations are possible, we arbitrarily take as a threshold the value of -500m corresponding to the approximate elevation of the Dead Sea. We will then impute the missing values with the median of the column.

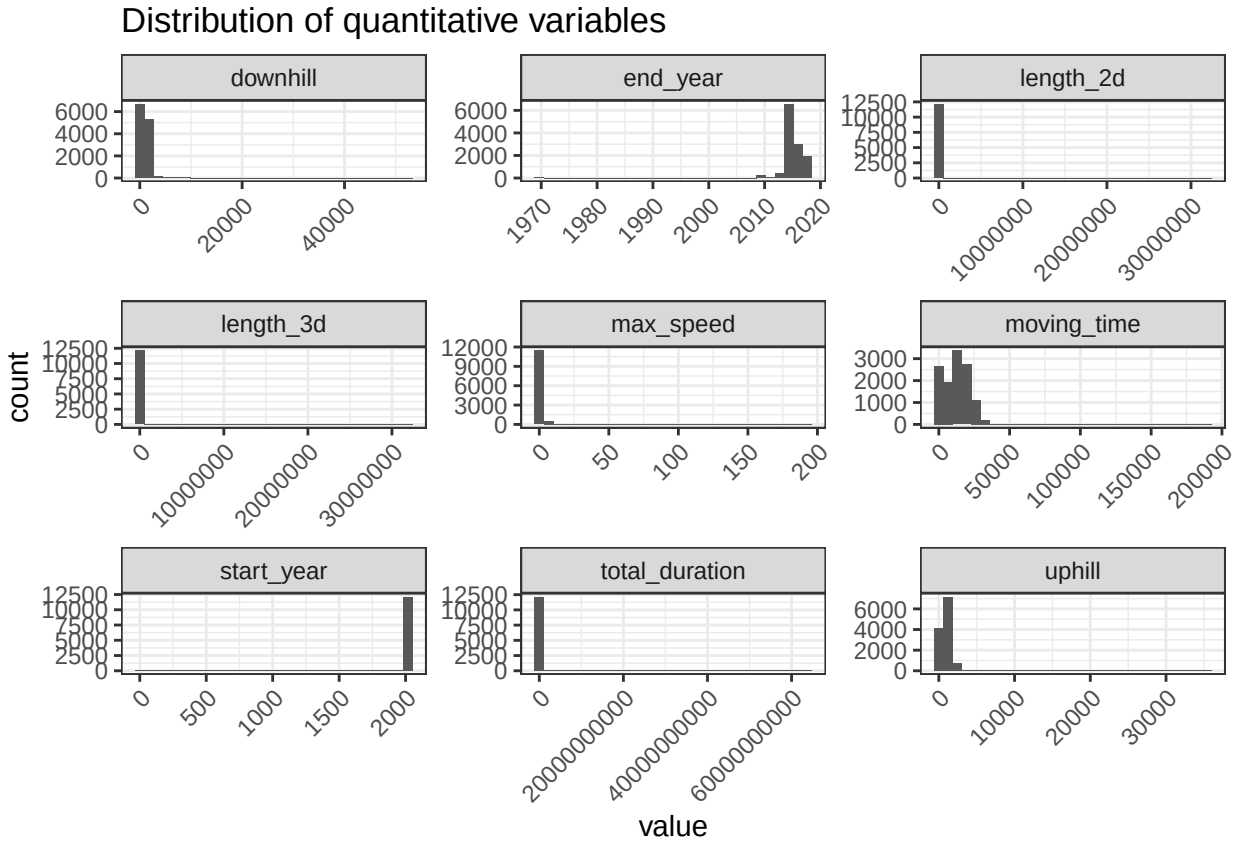
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##      -1    1382    1987    1934    2498    5633    1578
```

Since there is no unrealistic value to manage in the `max_elevation` variable, we will directly impute the missing values with the median of the column.

For the missing values in the time variables, we will create a new column for the total duration of the hike in seconds, to coincide with the unit of the `moving_time` column, then separate the year, month and hour components of the `start_time` and `end_time` variables, and impute the missing values with the median of the respective columns, rounded to the closest integer. The start and end time columns will be removed from the dataset.

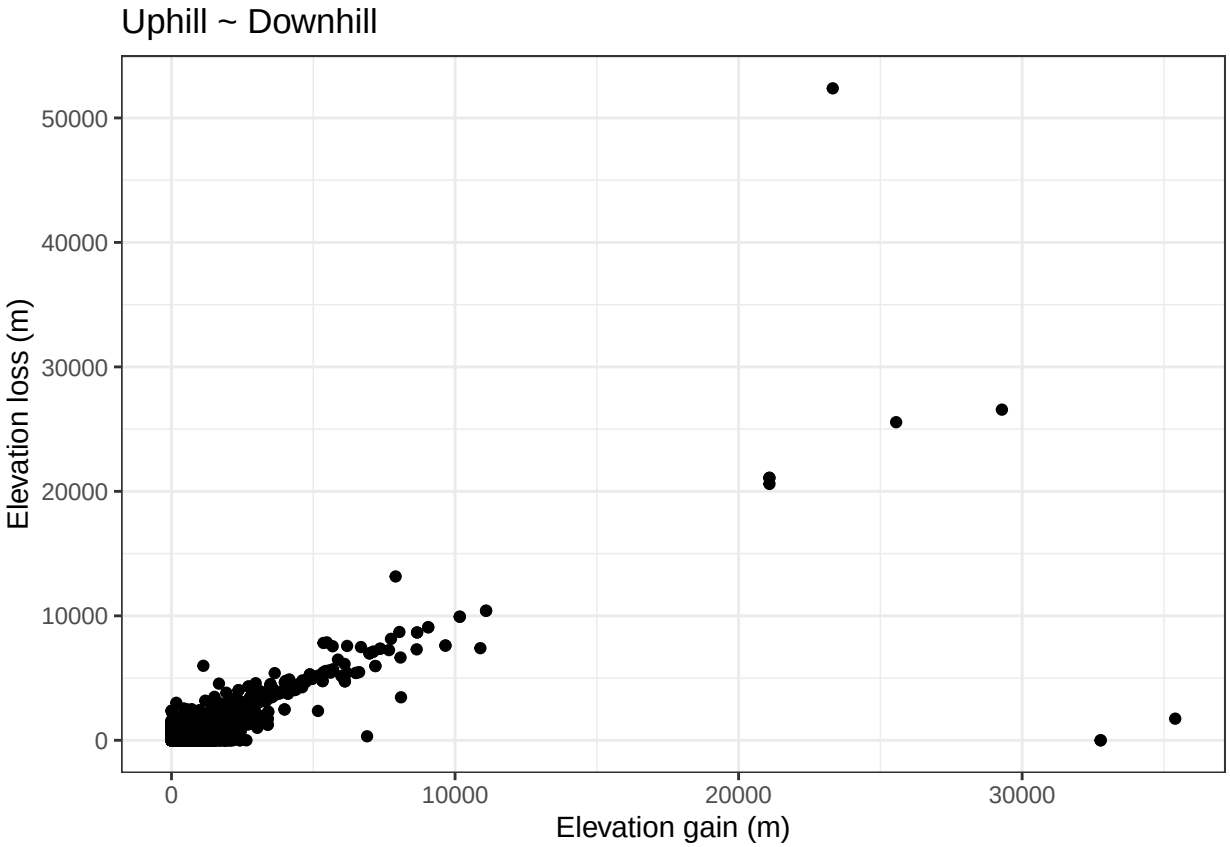
Unrealistic values and outliers

We have successfully imputed all missing values in the dataset. But some observations still contain unrealistic values. We will first check the distributions of the quantitative variables to identify outliers. We do not visualize the distributions of `max_elevation` and `min_elevation` as we already checked for unrealistic values.



Visualizing the distributions of the quantitative variables, we see that all are cramped on a side of the plot, suggesting the presence of extreme outliers.

In `uphill` and `downhill` We will use a scatterplot to visualize the relationship between the `uphill` and `downhill` variables.

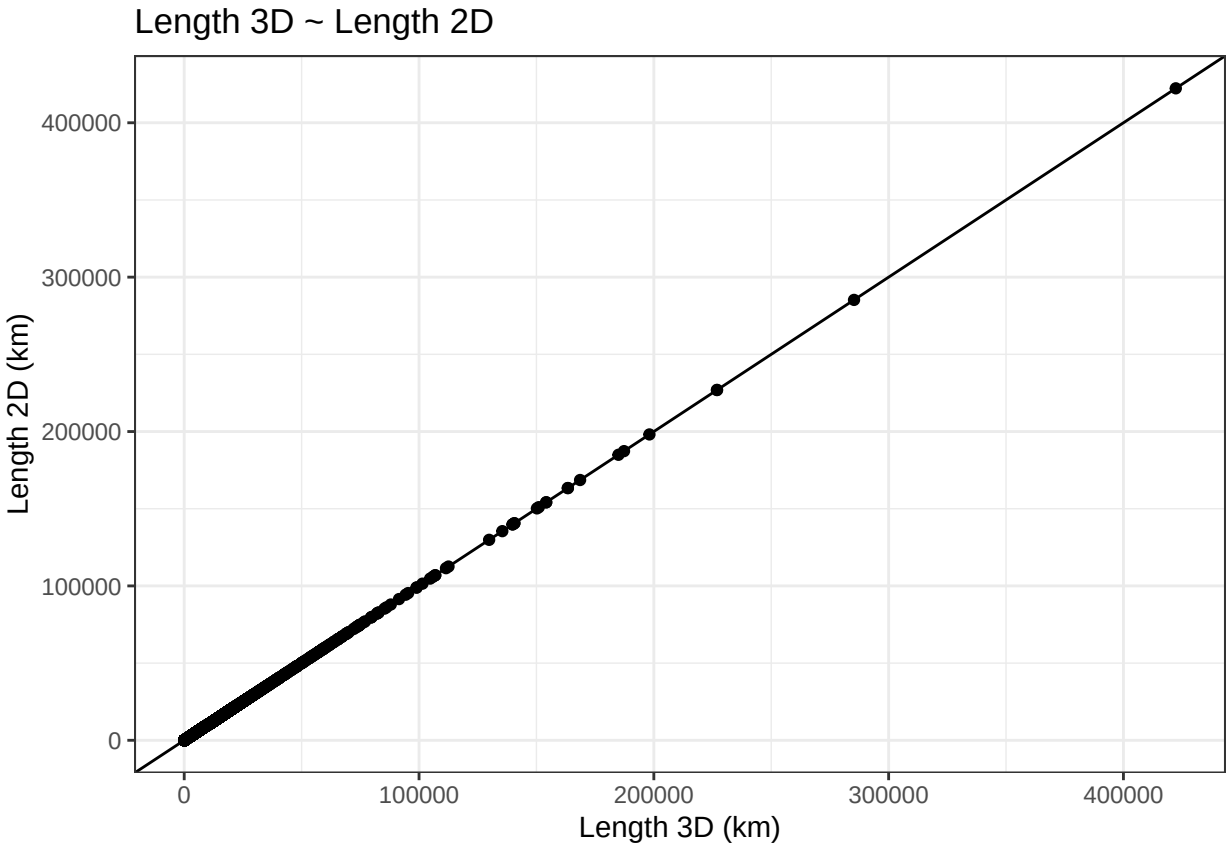


We can identify unrealistic values in the `uphill` and `downhill` variables. We see that the highest values are over 50,000 meters for the `downhill` variable, and over 30,000 meters for the `uphill` variable. We will remove observations with `uphill` over 10,000 meters, as this threshold removes 14 hikes, accounting for 0.1% of the dataset.

In time variables Some hikes have start and end year in the year 0, or 1970, which is unrealistic for the first, and the start of the unix time for the latter. Twenty-eight hikes have a `start_year` or `end_year` below 2000. Using the `table()` function, we can see that among those, 27 start and end years are 1970, and one starts in the year 0 but ends in 2015.

Because of this date issue, we can see that some other variables are impacted: `total_duration`, `moving_time`, and `max_speed`. We will not drop them since all other variables contain realistic values, and not like the `uphill` and `downhill` variables, these other values are not related to the time variables. We will thus impose missing values the variables stated above, for observations with a `start_date` below 2000, and then impute them with the column median.

In `length_3d` and `length_2d` Using the `summary()` function, we see that the `length_3d` and `length_2d` variables have the exact same summary statistics, with the highest value being over 30,000 km.



Plotting the `length_3d` against the `length_2d` variable, we see that the two variables are equal, perfectly following the x=y line. We will thus remove the `length_2d` variable and rename the `length_3d` variable to `length`. We will also drop the observations with a length above 100 km, which corresponds to 29 hikes. Removing these extra 29 observations would result in a total data loss from the original dataset of 0.35%, we will then do it.

In `max_speed` The `max_speed` variable contains unrealistic values, with the highest value being over 190 km/h. We will remove observations where the `max_speed` is above 25 km/h, which corresponds to 41 hikes, which brings the total amount of data loss to 0.7% of the original dataset. We acknowledge that 25 km/h is a high threshold for hiking, but we will use it to remove the most extreme outliers, and still account for some trail runners. Since the variable represents the maximum reached speed and not the average speed, a fast trail runner downhill could reach this level of speed.

In `total_duration` and `moving_time` We first notice that there is 98 observations with impossible values. That is if an observation complies with at least one of the following conditions:

- `total_duration` $<$ `moving_time`,
- `total_duration` < 0,
- `moving_time` < 0.

Since the other variables do not seem to be impacted by these unrealistic values, we will proceed as with unrealistic date values, by imposing missing values for observations with `total_duration` or `moving_time` below 0, and then impute them with the median of the respective columns.

The remaining observations with `total_duration` $<$ `moving_time` will be dropped. We now have a 0.9% data loss from the original dataset.

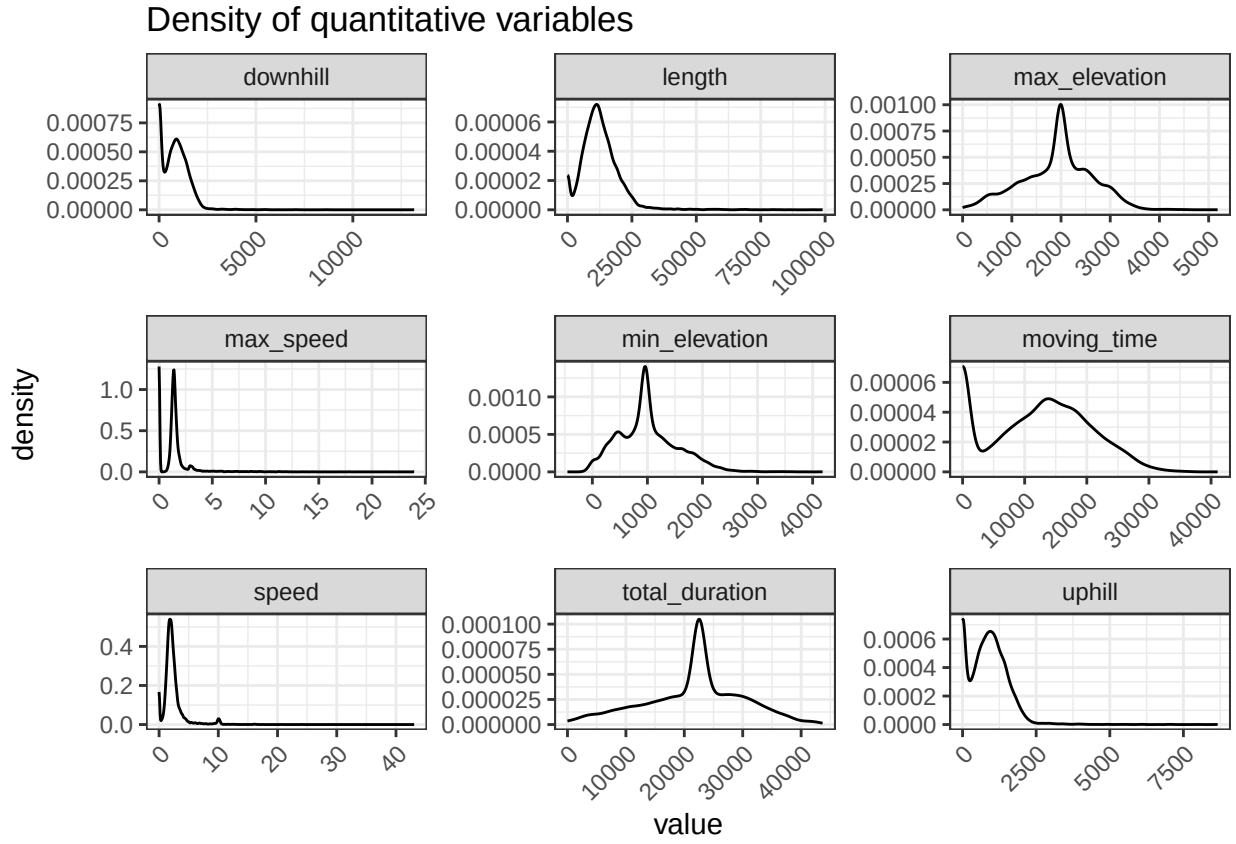
We still have some extreme outliers in the `total_duration` variable, with the highest being over 78 years long. We will remove observations with a `total_duration` above 1.5 times the interquartile range above the third quartile, which corresponds to 321 hikes, resulting in a total data loss of 3.6% of the original dataset.

Feature engineering

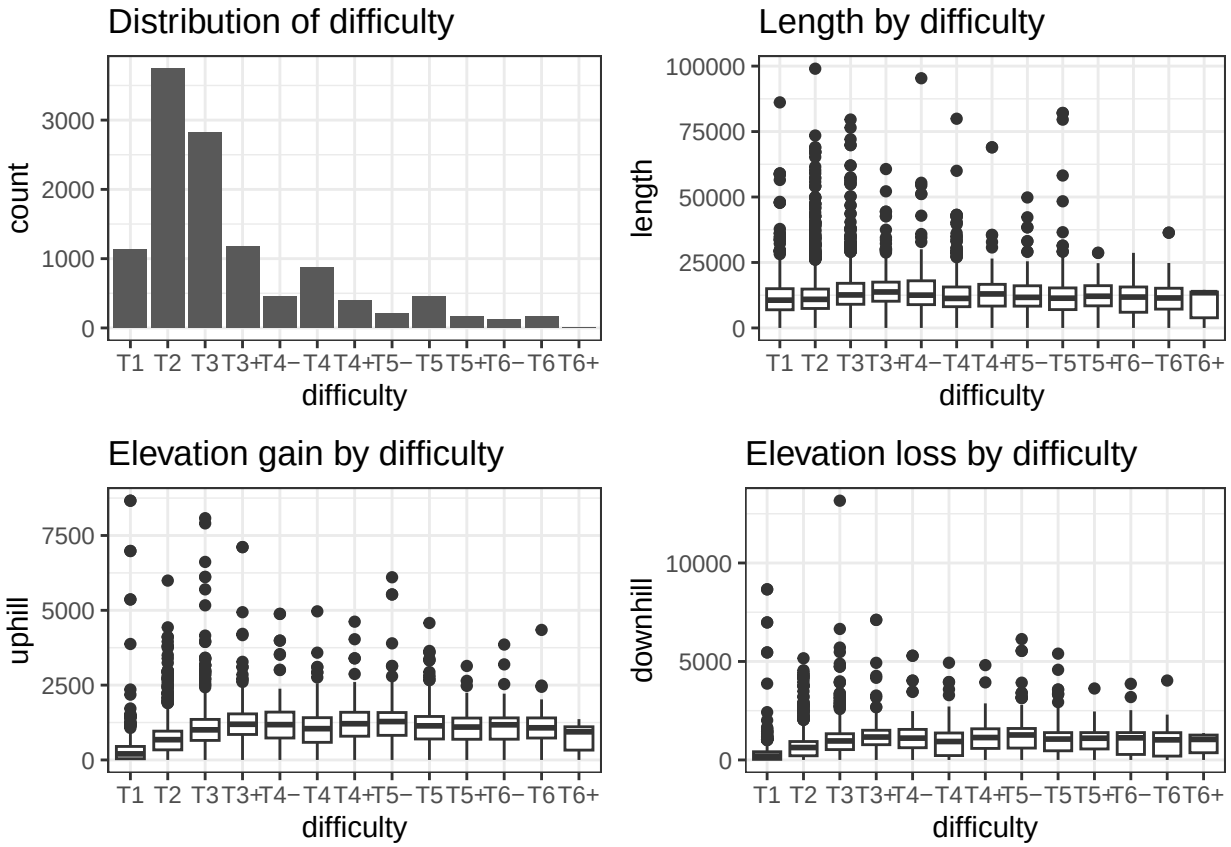
We will create a new variable `speed` representing the average speed of the hike in km/h, by dividing the `length` variable by the `total_duration` variable, then multiplying by 3.6 to convert the speed from m/s to km/h. If `total_duration` is 0, we will force a missing values, and then impute it with the median of the column.

Exploratory Data Analysis

After these data cleaning steps, we can now proceed with the exploratory data analysis. We will start by visualizing the density of the quantitative variables.



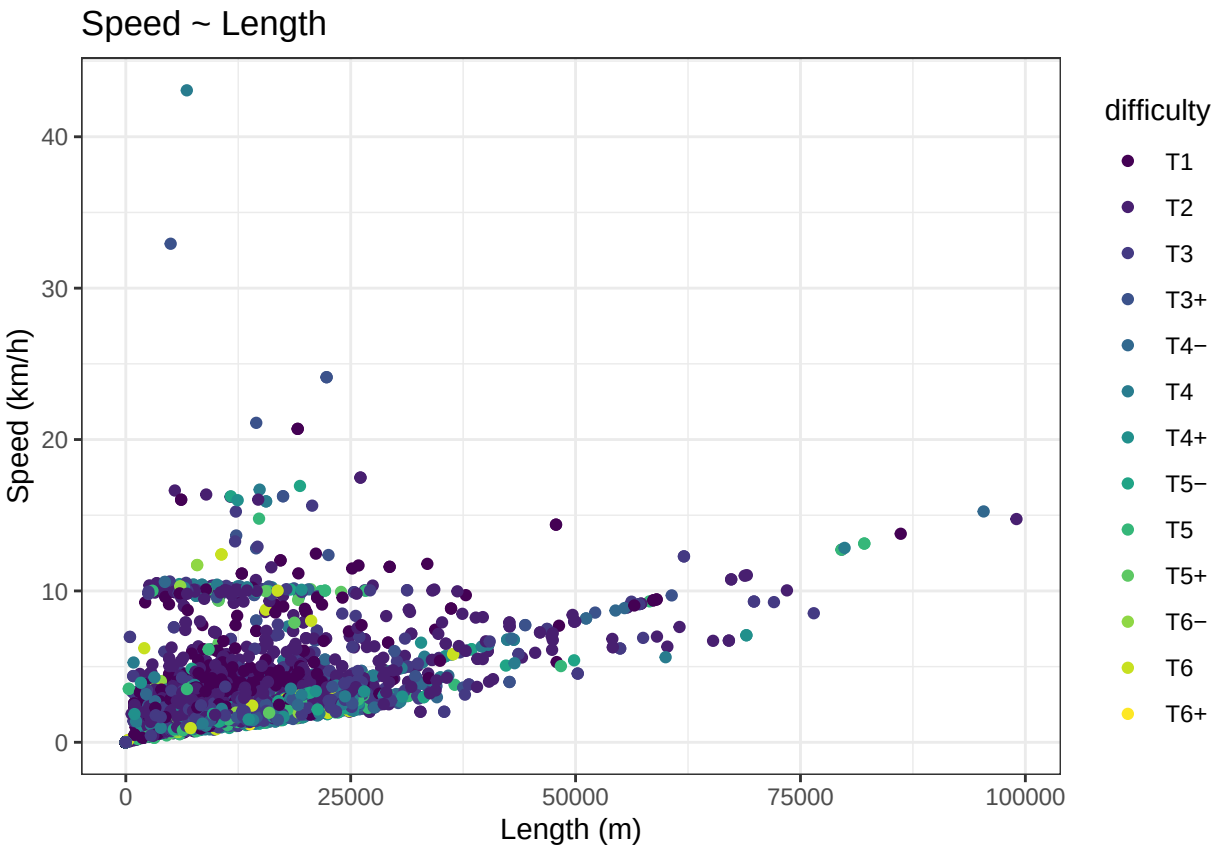
We can see that the variables are either right-skewed or similar to a normal distribution. We often have a peak in the density plot at the median, due to the imputation of missing values with the median of the column.



We can see that most of the hikes are rated T2 or T3. We have an increasingly lower number of hikes as the difficulty increases, with the most difficult hikes being the least represented in the dataset.

Surprisingly, we don't see a clear relationship between the hike's difficulty and its length. We can see that the hikes rated T6 have a higher median length than the hikes rated T5, but the hikes rated T4 have a lower median length than the hikes rated T3. This suggests that the difficulty rating is not solely based on the hike's length.

The same observation can be made for the elevation gain and loss. We see that the hikes rated T6 have a lower median elevation gain and loss than the hikes rated T5. This suggests that the difficulty rating is not solely based on the hike's elevation gain and loss.



Visualizing the speed against the length of the hike, we see a relashionship between the two variables.

But we can also identify outliers in the `speed` variable. We will remove observations with a `speed` above 15 km/h, which corresponds to 26 hikes, resulting in a total data loss of 3.8% of the original dataset.

Methodology

We will now proceed with the predictive analysis of the hikes' duration and difficulty. We will start by splitting the dataset into a training and a testing set, with 70% of the observations in the training set, and 30% in the testing set.

Then, we will run `regsubsets()` to select the most relevant variables using the forward method for a regression analysis of the `total_duration` variable based on the other variables in the dataset. We will then perform a 200 folds cross-validation to select the best model, and evaluate its performance on the testing set using the RMSE. We will start with only the following variables, that are the only ones that are known before the hike occurs: `length`, `uphill`, `downhill`, `difficulty`, `min_elevation`, `max_elevation`, `start_year`, `start_month`, `start_hour`.

For the classification analysis of the hikes' difficulty, we will create a new variable `is_diffucult` that will be a binary variable, with 0 for hikes rated T1 to T3 included, and 1 for hikes rated above T3. The T3 threshold was chosen as it was the best one to cut the dataset in two equal parts. We will then run a logistic regression analysis of the `is_difficult` variable based on the other variables in the dataset, excluding `difficulty`, and evaluate the model's performance on the testing set using the accuracy metric.

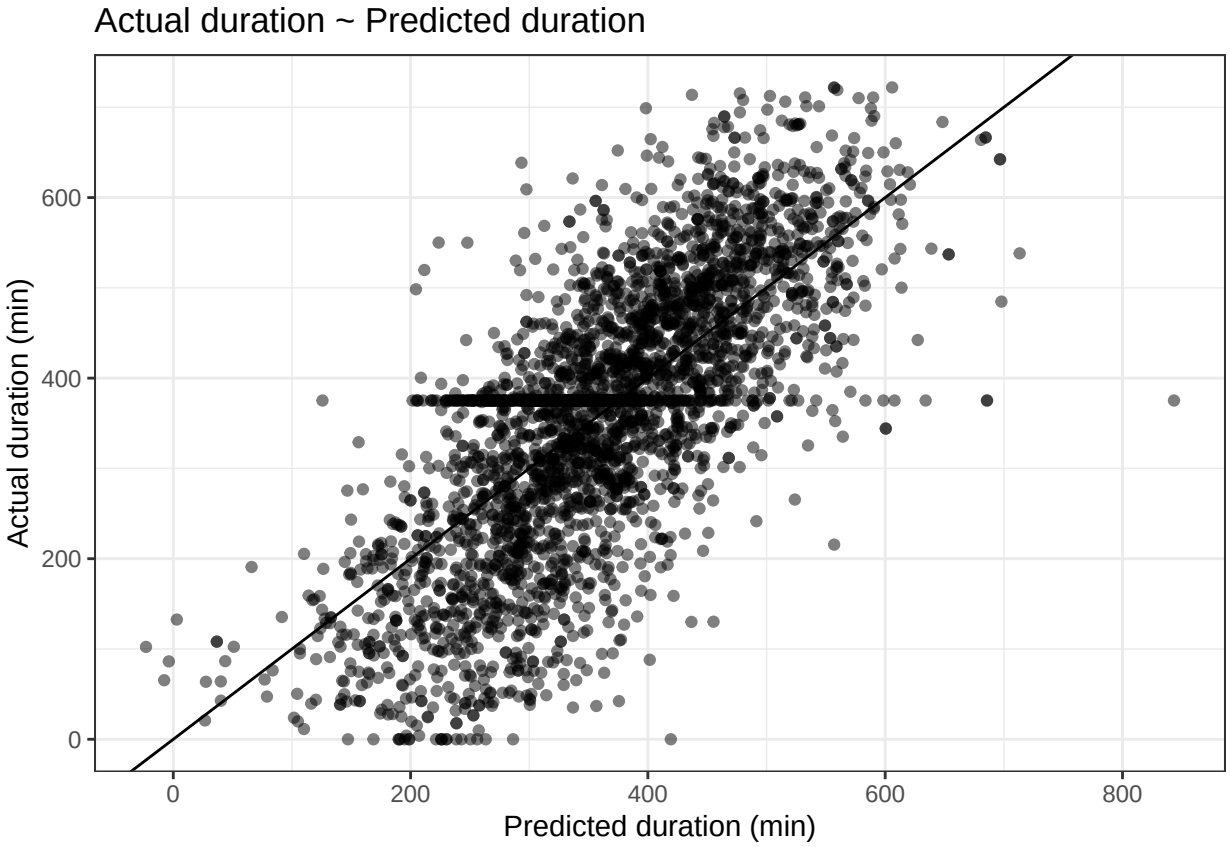
Then, we will perform a clustering analysis of the hikes' season based on the `speed` and `max_elevation` variables. We will use the k-means algorithm to cluster the hikes into two clusters - summer or not summer

(winter) - based on these variables and visualize the clusters. We define summer hikes as hikes starting in May, June, July, August, or September, and winter hikes as hikes starting in the other months.

Modelling and results

Predictive analysis of the hikes' duration

The best model has a root mean squared error of 5817.35, which means that the predicted duration of the hikes is on average 96.96 minutes away from the actual duration, with a median of the variable of 375.22 minutes.

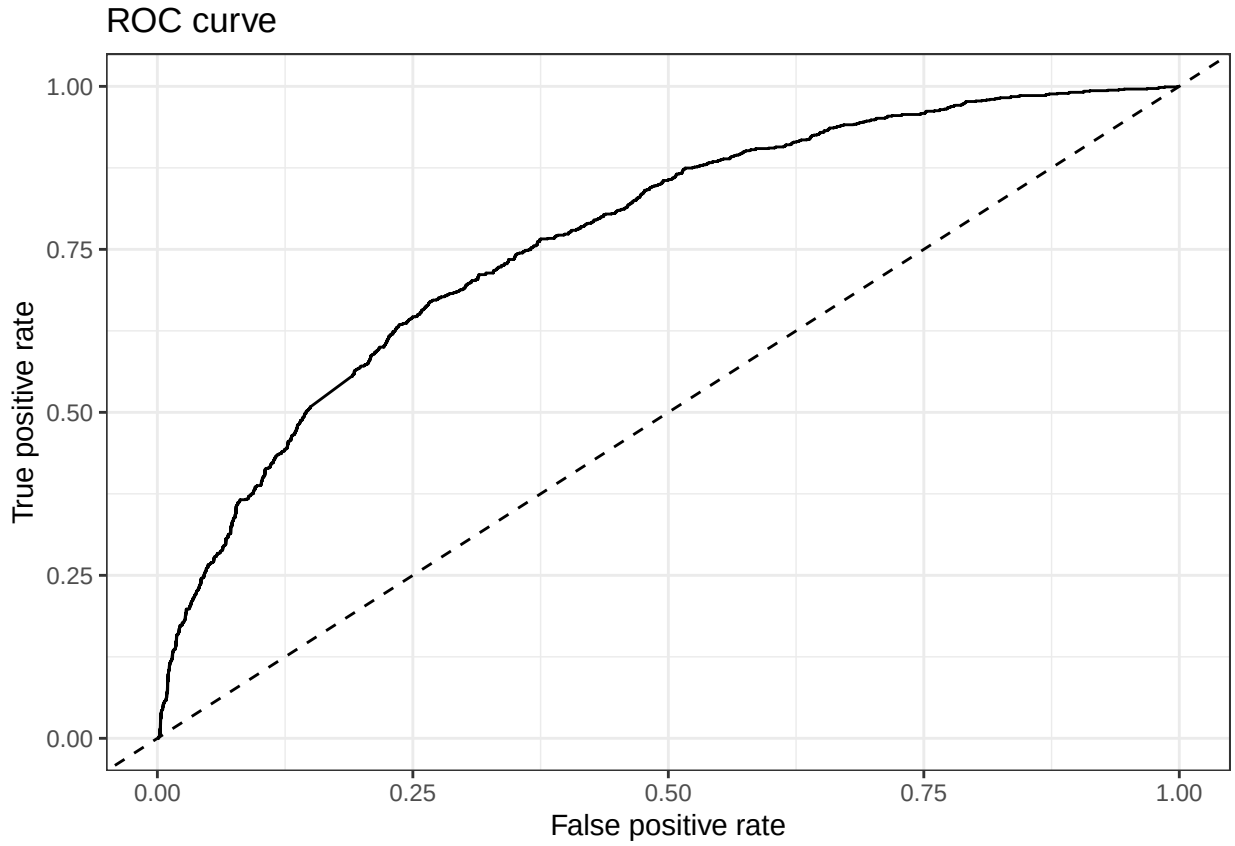


This plot shows that the predicted duration of the hikes is relatively close to the actual duration, with an overall linear relationship between the two variables following the x=y line.

Classification analysis of the hikes' difficulty

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

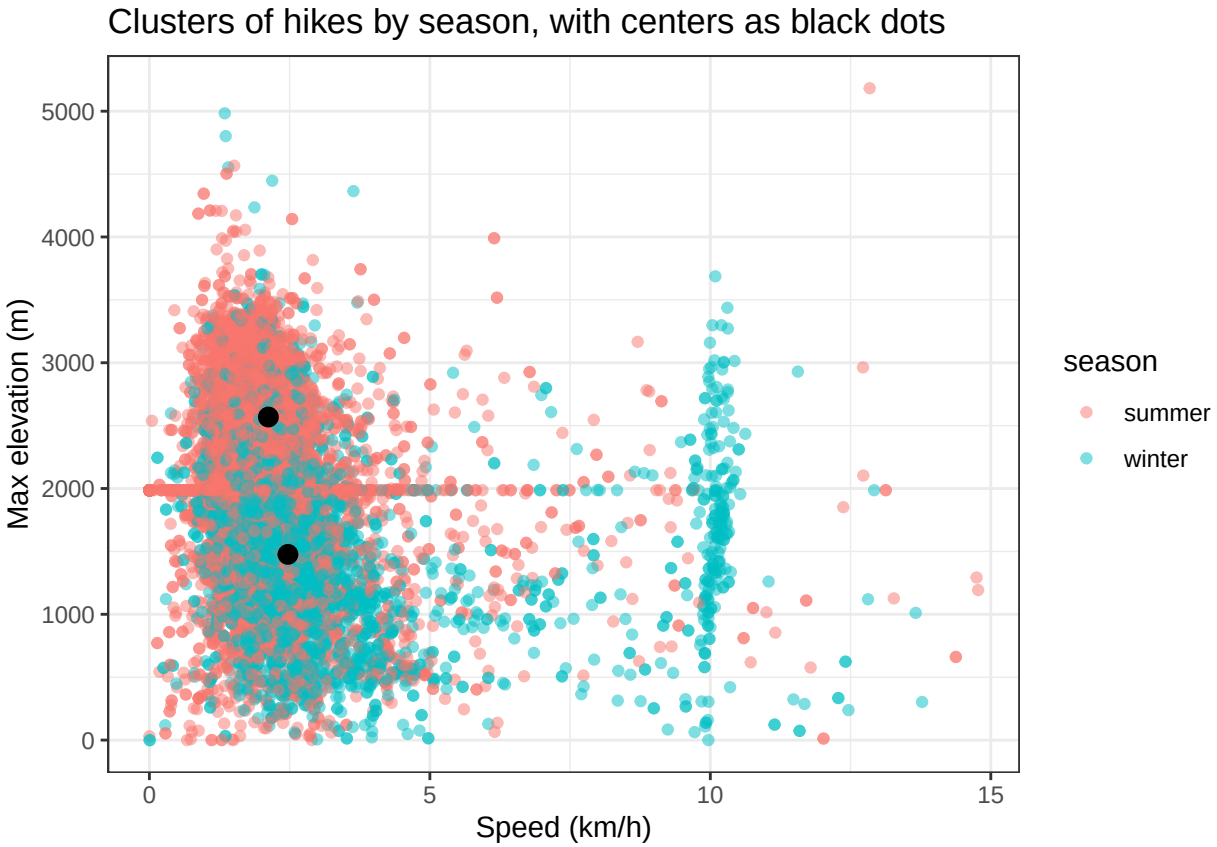
```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```



The logistic regression model has an accuracy of 0.73, which means that the model correctly predicts the difficulty of the hikes in the testing set in 72.75% of the cases. Given that the dataset is fairly balanced, with 65% of the hikes rated T1 to T3, and 35% rated above T3, this accuracy is quite good.

The ROC curve shows that the model has a good trade-off between sensitivity and specificity, with a high true positive rate and a low false positive rate. This performance is far from perfect but is still acceptable.

Clustering analysis of the hikes' season



The k-means algorithm identified two clusters of hikes based on the speed and maximum elevation of the hikes. The black dots represent the centers of the clusters. We can see that the algorithm successfully separated the hikes into two clusters based on the season, with the summer hikes having a lower speed and higher maximum elevation than the winter hikes.

Limitations

This analysis has several limitations:

The dataset contains missing and unrealistic values that were imputed or removed, which could have impacted the results of the analysis. This median imputation is well visible on most plots.

This analysis is based on a dataset that is not up-to-date, being scrapped in 2018, and may not be representative of current hikes.

Hikr.org is a platform where users can post reports about their hikes, and thus are likely to be more advanced hikers, or at least passionate ones, making the dataset likely biased and not representative of all hikes in the world.

References

- R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

- Wickham, H., François, R., Henry, L., & Müller, K. (2023). *dplyr: A Grammar of Data Manipulation*. R package version 1.1.0. https://CRAN.R-project.org/package=dplyr.
- Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org.
- Tierney, N., Cook, D., & Prvan, T. (2023). *naniar: Data Structures, Summaries, and Visualisations for Missing Data*. R package version 0.6.1. https://CRAN.R-project.org/package=naniar.
- Kuhn, M. (2023). *caret: Classification and Regression Training*. R package version 6.0-93. https://CRAN.R-project.org/package=caret.
- Lumley, T. (2023). *leaps: Regression Subset Selection*. R package version 3.1. https://CRAN.R-project.org/package=leaps.
- Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, 33(1), 1-22. URL https://www.jstatsoft.org/v33/i01/.
- Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J.-C., & Müller, M. (2011). pROC: An open-source package for R and S+ to analyze and compare ROC curves. *BMC Bioinformatics*, 12, 77. https://doi.org/10.1186/1471-2105-12-77.
- Rocco. (2018). *GPS recorded hikes from hikr.org*. Kaggle. https://www.kaggle.com/datasets/roccoli/gpx-hike-tracks (License: CC0)