Final Machine Learning Project

Wanderlust Bundle

Jayasri Maditati

MIS64060:Fundamentals of Machine Learning
Professor Murali Shanker
Kent State University

12 December 2021

Abstract

The objective of this final exam is to apply the appropriate machine learning technique to the business problem, and then present the solution to top-level management.

Contents

I. Introduction:

Trail Segmentation and Trail Recommendation from the user specified trail information.

This dataset (**AllTrails**) is from Kaggle that provides information on the top/best trails in the United States and Hawaii. It comprises of both numerical and categorical variables. The variables in the dataset are:

- **Trail ID** - A Unique Id given for each trail
- **Name** - Name of the trail
- **Area Name** – Name of the National Park in which the trail is located
- **City Name** – City in which the trail is located
- **State Name** – State in which the trail is located
- **Country Name**- Name of the Country
- **X_geoloc** – Details about Latitude and Longitude of the trail
- **Popularity** – Details about how popular the trail is
- **Difficulty Rating** (Easy, Moderate, Hard) – 1,3,5,7
- **Average Rating** – It is given by Reviewers (0 to 5)
- **Number of Reviews** – Number of reviews for the trail
- **Length (Meters)** – Length of the Trail
- **Elevation Gain (Meters)** – Elevation Gain of the Trail
- **Route Type** (Loop, Out & Back, Point to Point)
- **Visitor Usage** – Usage of the trail by visitors – 1,2,3,4
- **Features** - Trail features such as paved,river,wildlife,views etc..
- **Activities** - Activities designated mainly for the trails - Camping,birding,biking,walking,hiking etc..

II. Data Exploration

```
Warning: package 'rattle' was built under R version 4.1.2
```

```
#Importing the dataset.
AllTrails_data <- read.csv("AllTrails_new.csv", header = TRUE)
#Check for dimensions
dim(AllTrails_data)
```

```
## [1] 3284   18
```

This All trails dataset has 3284 observations and 18 features.

III. Data Preparation

This Dataset consists of trails from US and Hawaii. I am going to focus only on the United States Trails.

```
#Filter only US Trails
AllTrails_US_NA<-AllTrails_data %>% filter(country_name == "United States")
```

After filtering only the US Trails, we have 3240 observations and now lets add the test data frame to the dataset

```
df9<-data.frame(trail_id=10259853,name="Cades Cove Loop Road",area_name="Great Smoky Mountains National
,state_name="Tennessee",country_name="United States",X_geoloc="{'lat': 35.60651, 'lng': -83.77419}",popu
```

```
#To check if NA Values exist
colMeans(is.na(AllTrails_US_NA))
```

```
##          trail_id              name         area_name         city_name
##        0.00000000        0.00000000        0.00000000        0.00000000
##        state_name      country_name          X_geoloc        popularity
##        0.00000000        0.00000000        0.00000000        0.00000000
##            length    elevation_gain difficulty_rating        route_type
##        0.00000000        0.00000000        0.00000000        0.00000000
##     visitor_usage        avg_rating       num_reviews          features
##        0.07528541        0.00000000        0.00000000        0.00000000
##        activities             units
##        0.00000000        0.00000000
```

We can see that 'visitor_usage' column has the missing values in the dataset. So we have removed the NA values.

```
#Omitting the NA Values from the dataset
AllTrails_US<-na.omit(AllTrails_US_NA)

dim(AllTrails_US)
```

```
## [1] 2997   18
```

```
#Remove some of the qualitative variables which is not necessary
Trail_Data <- AllTrails_US[,-c(2,3,4,5,6,7,18)]
```

After removing NA Values, The dataset has 2996 observations and 18 features.

Now, let's create few tags by doing some feature engineering. The remaining features would be categorically tagged with a value of 0 for "No" or 1 for "Yes" depending on if the feature described a given hike. I would be representing them as Trail Tags

Recoding the Trail_Tags:

```
# Catgorically tagging the variables with 1/0

for(i in 1:length(Trail_Data$activities)){

  if(length(grep("driving",Trail_Data$activities[[i]],perl = TRUE, value=FALSE)) > 0 |
     length(grep("bike",Trail_Data$activities[[i]],perl = TRUE, value=FALSE)) > 0){
    Trail_Data$driving[i] <- "1"

  }else{
    Trail_Data$driving[i] <- "0"
  }
}



for(i in 1:length(Trail_Data$activities)){

  if(length(grep("canoeing",Trail_Data$activities[[i]],fixed = TRUE, value=FALSE)) > 0 |
     length(grep("sports",Trail_Data$activities[[i]],perl = TRUE, value=FALSE)) > 0 |
     length(grep("skiing",Trail_Data$activities[[i]],perl = TRUE, value=FALSE)) > 0 |
     (length(grep("snow",Trail_Data$activities[[i]],perl = TRUE, value=FALSE)) > 0) |
     length(grep("sports",Trail_Data$activities[[i]],perl = TRUE, value=FALSE)) > 0){

    Trail_Data$water_sports[i] <- "1"

  }else{
    Trail_Data$water_sports[i] <- "0"
  }
}

#View(Trail_Data)

# Convert the variables into numeric
Trail_Data$trail_id<- as.integer(Trail_Data$trail_id)
Trail_Data$driving <- as.numeric(Trail_Data$driving)
Trail_Data$water_sports <- as.numeric(Trail_Data$water_sports)

# Lets select the features which is required to achieve our objective
Trail_Data_new <- Trail_Data[,-c(1,2,5,6,7,8,9,10,11)]
colnames(Trail_Data_new)
```

```
## [1] "length"         "elevation_gain" "driving"        "water_sports"
```

Data Normalization

Since we are dealing with variables of very different nature that are expressed in different units, we have to normalize the data before applying any algorithm.

```r
#Normalize the data using scale function
Trail_data_norm<- scale(Trail_Data_new)

#Top rows of the dataset after preprocessing
head(Trail_data_norm)
```

```
##        length elevation_gain    driving water_sports
## 1 -0.05475518     -0.4897691  4.6035776   -0.2848539
## 2 -0.08252660      0.5917558 -0.2171499   -0.2848539
## 3 -0.45744068     -0.1521392 -0.2171499   -0.2848539
## 4 -0.63101201     -0.6363989 -0.2171499   -0.2848539
## 5 -0.61018345     -0.5934152 -0.2171499   -0.2848539
## 6  0.52844450      0.5494654 -0.2171499   -0.2848539
```

Distance Measure

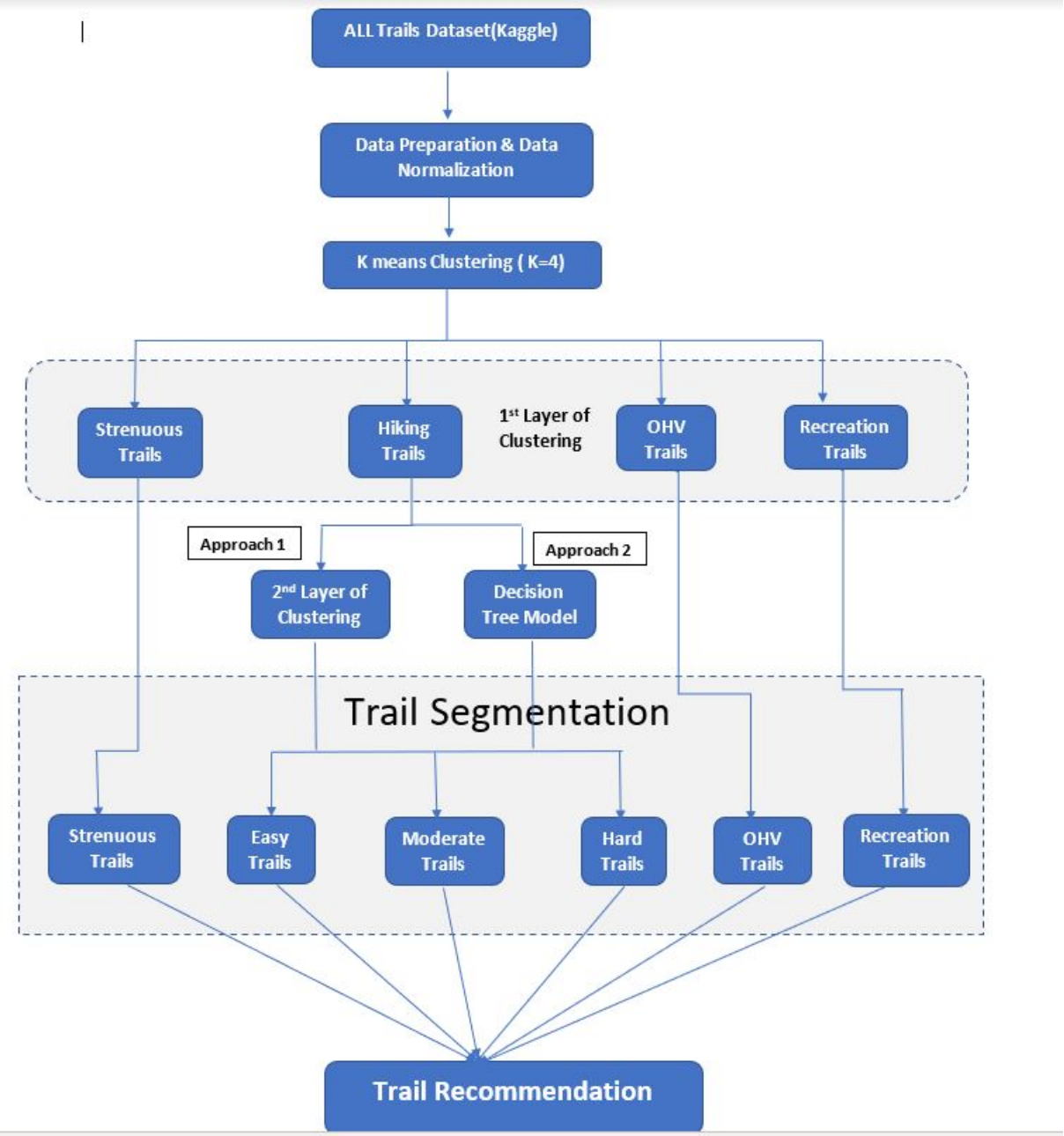
For computing distance, we are going to use the **get__distance** function.It uses the Euclidean distance as default metric and **fviz__clust** is to visualize the distance matrix.

```r
#Computing distance. Euclidean distance as a default metric
distance<- get_dist(Trail_data_norm)
head(distance)
```

```
## [1] 4.940636 4.849285 4.857261 4.853726 4.965838 4.949525
```

Solution

Flowchart of the approach followed



IV.I Trail Segmentation

To achieve my objective ie, Trail recommendation which is a use case of Trail segmentation. We need to perform Clustering technique as it fits my objective which is to segment the trails.

The main objective of the Clustering is: It is the process of grouping a set of objects into the classes of similar objects. Clustering can be used as a stand – alone tool to gain insight into data distribution and as a preprocessing step of other algorithms in intelligent systems.

Speaking of which, I chose K-means model to be my ML algorithm for this project as it is compatible with high dimensional and large dataset,whereas DB Scan Algorithm doesn't work well with high dimensional data and Hierarchical clustering wasn't able to handle this dataset computationally. So Lets run k-means model.

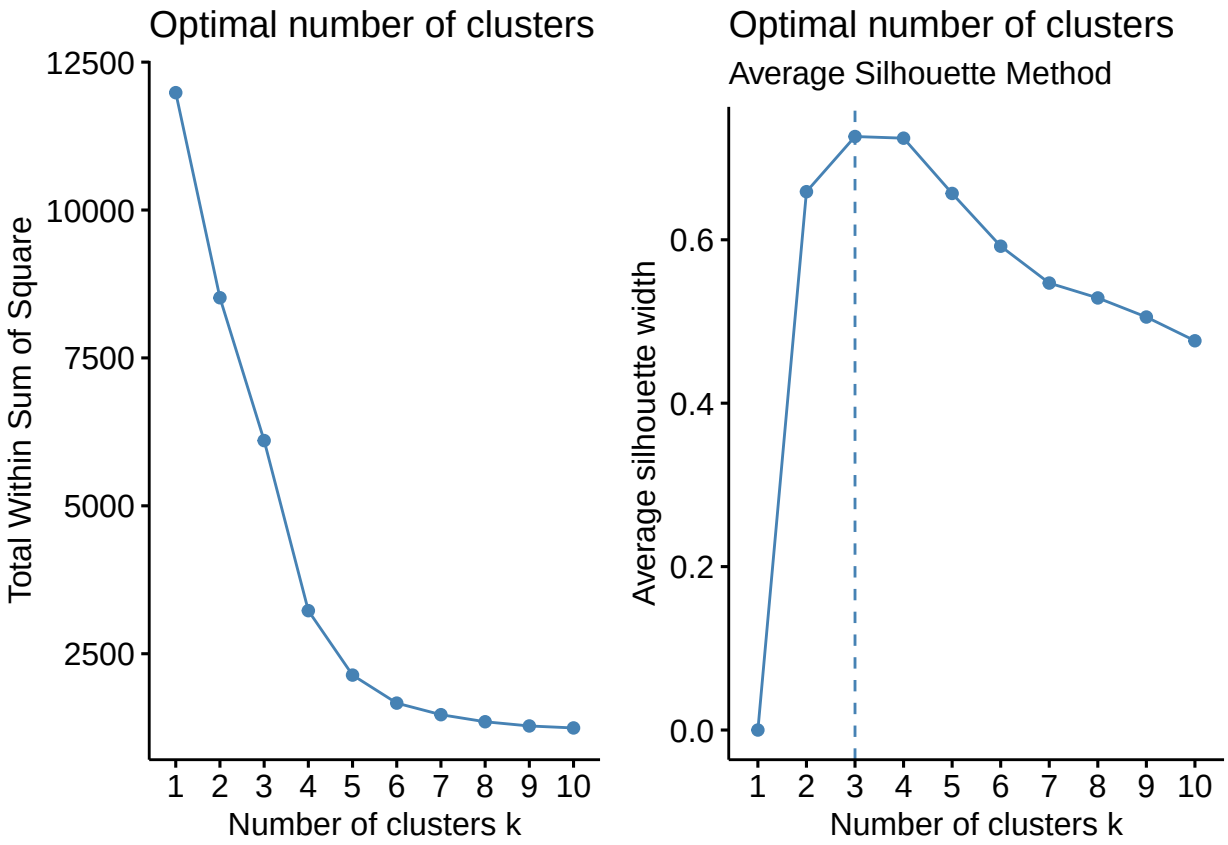
To find the Optimal K

The choice of the number of clusters can either be driven by:

- **External Considerations**(e.g., previous knowledge, practical constraints, etc.)
- **The Elbow Method**
- **The Average Silhouette Method**

First, Lets use the two data driven methods to check for the value of K.

```
wss<-fviz_nbclust(Trail_data_norm,kmeans,method = "wss")
Avg<-fviz_nbclust(Trail_data_norm,kmeans,method = "silhouette")+
labs(subtitle = "Average Silhouette Method")
plot_grid(wss,Avg)
```



From the two methods plotted, we can see that the Elbow Method gives us the optimal value of k is 4, While the Silhouette Method gives us k=3 as a result.

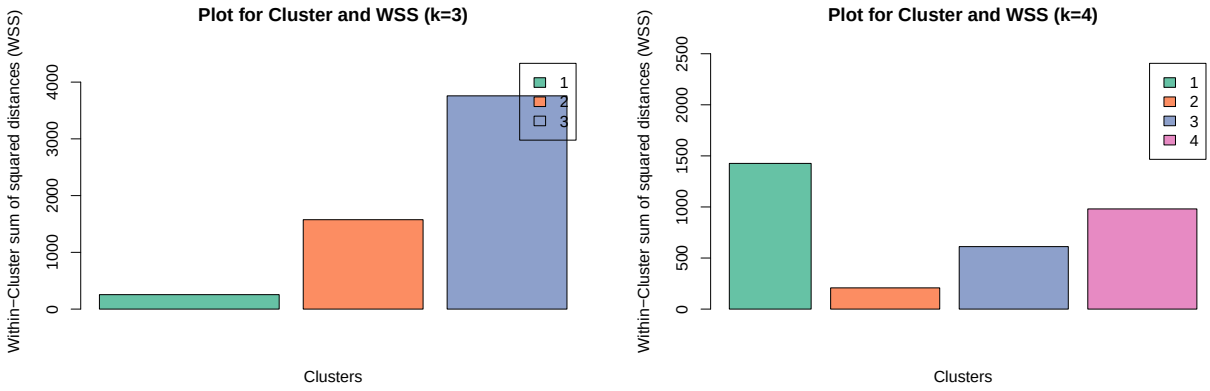
I tried to run k-means Model with both k=3 and 4 and decided to consider withinss to find the optimal k among these two

Running the K-Means Model (using K=3 & 4)

```
set.seed(123)
# K means model when k=3
k3<- kmeans(Trail_data_norm,centers = 3,nstart = 25)
# K means model when k=4
k4<- kmeans(Trail_data_norm,centers = 4,nstart = 25)
```

Plot for Cluster and WithinSS

These are the plots for clusters with their corresponding Within cluster sum of squared distances when k=3 and k=4



Observations:

WithinSS is a measure of dispersion of the data within the cluster.

- When we take look at the above plots, It is clearly evident that the clusters in the plot where k=3 looks less homogeneous compared to the clusters in the plot where k=4.
- The goal here isn't just to make clusters, but to make good, meaningful clusters.Moreover when k=4, it certainly satisfies our objective.

I chose k=4 as the optimal value.

```
# To see the centers of the 4 clusters
k4$centers
```

```
##        length elevation_gain    driving water_sports
## 1  1.98993003      2.4421919 -0.1964601   -0.2848539
## 2 -0.04343395     -0.2014920 -0.2171499    3.5094001
## 3  1.27272874      0.1851130  4.6035776   -0.1999079
## 4 -0.25936753     -0.2280335 -0.2171499   -0.2848539
```

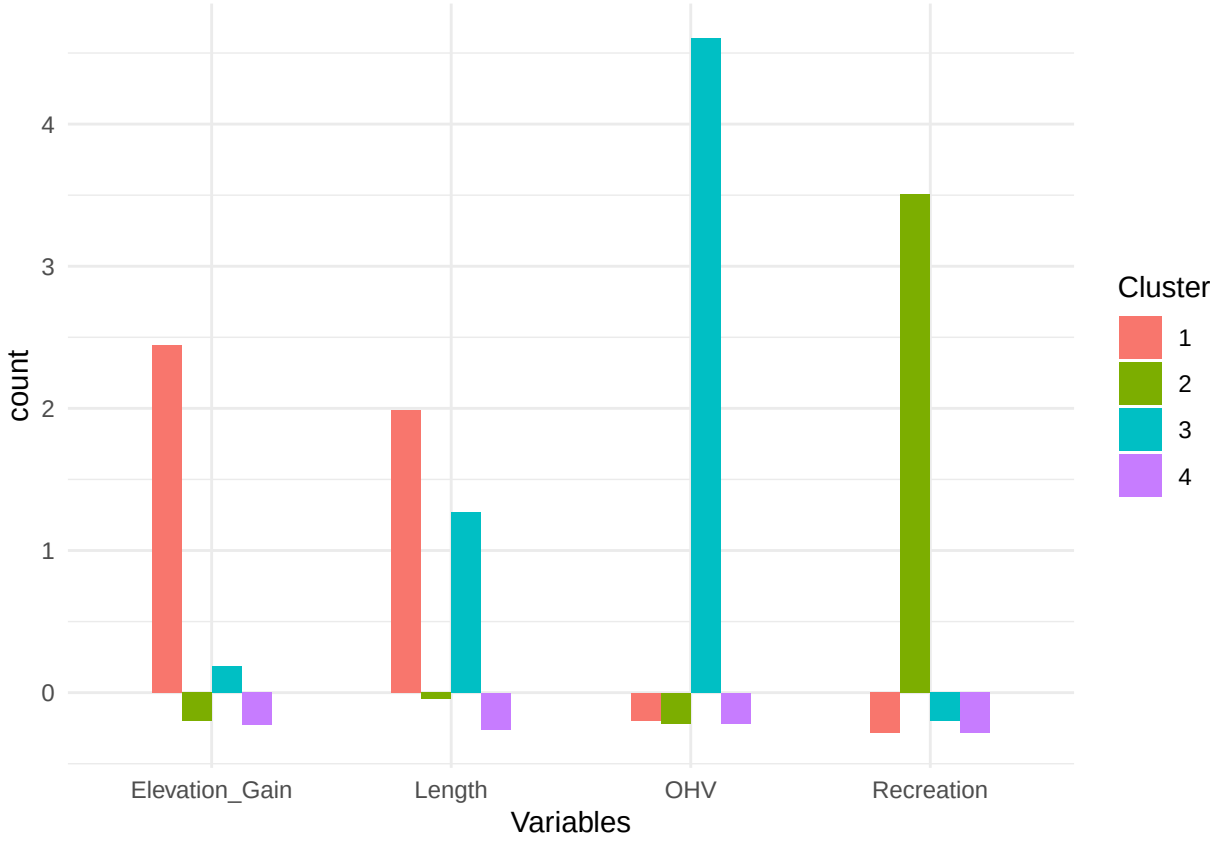
```
# To check the size of clusters
k4$size
```

```
## [1]  233  222  134 2408
```

```
#To identify to which cluster sizes belong
table(k4$cluster)
```

```
##
##    1    2    3    4
##  233  222  134 2408
```

Cluster Interpretation (1st layer of Clustering):



We can see that 4 clusters with sizes of different characterizations:

- **Cluster1:** Strenuous Trails
- **Cluster2:** Recreation Trails
- **Cluster3:** OHV Trails
- **Cluster4:** Hiking Trails

We are mainly focused on hiking trails and to cluster those trails into Easy, Moderate, and Hard trails

After the first layer of clustering, we are down to 4 clusters. Now,lets consider Cluster1 which has only the hiking trails.

Approach 1: Using K-Means Model (Second Layer of Clustering)

Further clustering only the Hiking trails which are in Cluster 1

```
# Considering only Elevation Gain
Trail_clust_Data_new <- Trail_clust4[,c(10)]


# Data Normalization

#Scale the dataset
Trail_clust_data_norm<- scale(Trail_clust_Data_new)
#Distance Measure
distance1<- get_dist(Trail_clust_data_norm)
```

Running K-Means Model

As my main objective of this doing this second layer of clustering is to divide the data into 3 segments namely, Easy, moderate, and Hard Trails.

So,I chose k =3 and lets run the model with k=3

```
set.seed(123)

k3_clust<- kmeans(Trail_clust_data_norm,centers = 3,nstart = 25)
#size of the cluster
k3_clust$size
```

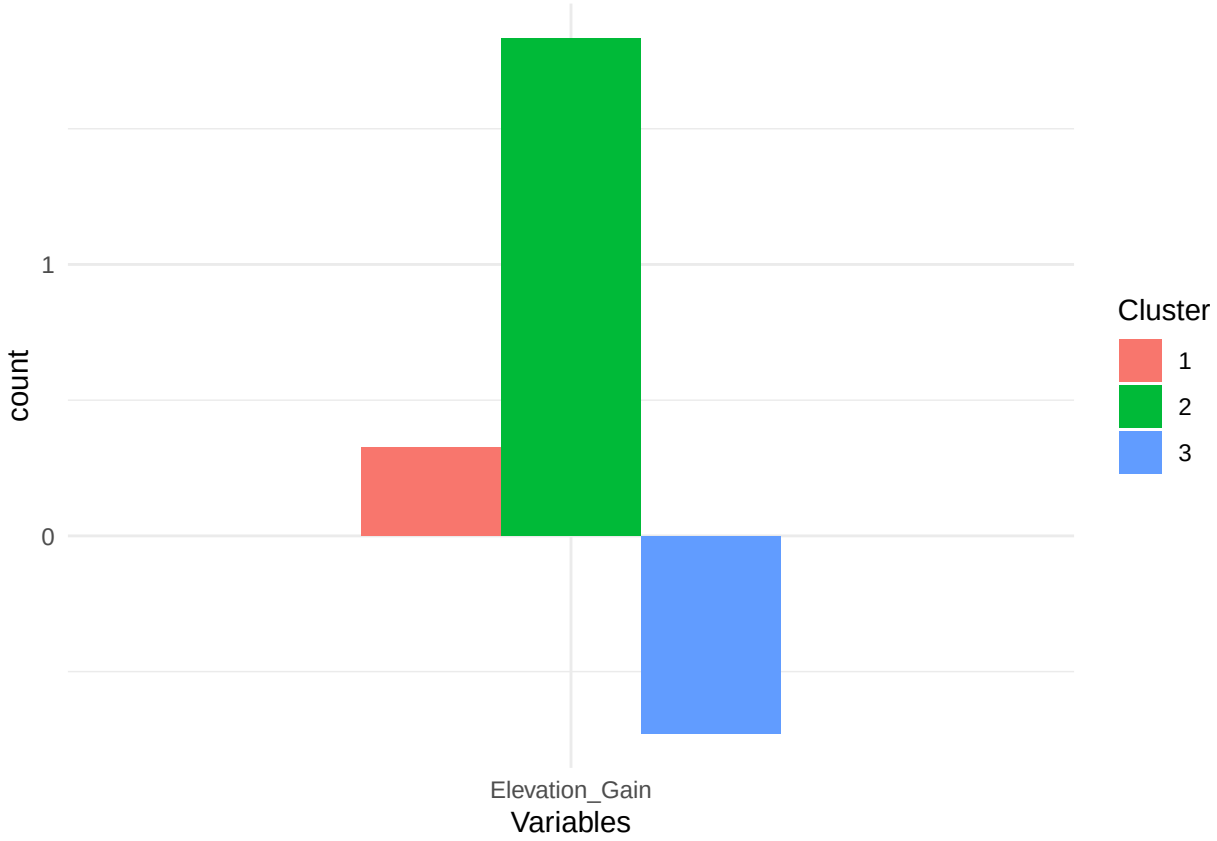
```
## [1]  699  396 1313
```

```
table(k3_clust$cluster)
```

```
##
##    1    2    3
##  699  396 1313
```

```
### Descriptive Statistics of 3 clusters
#Descriptive Statistics of 3 clusters
#describeBy(x=Trail_cluster_new,group="cluster",skew=FALSE)
```

Cluster Interpretation (2nd layer of clustering)



We can see that 3 clusters of Hiking Trails with different characterizations:

- **Cluster1:** Moderate Trails
- **Cluster2:** Hard Trails
- **Cluster3:** Easy Trails

Combining two layers of clustering

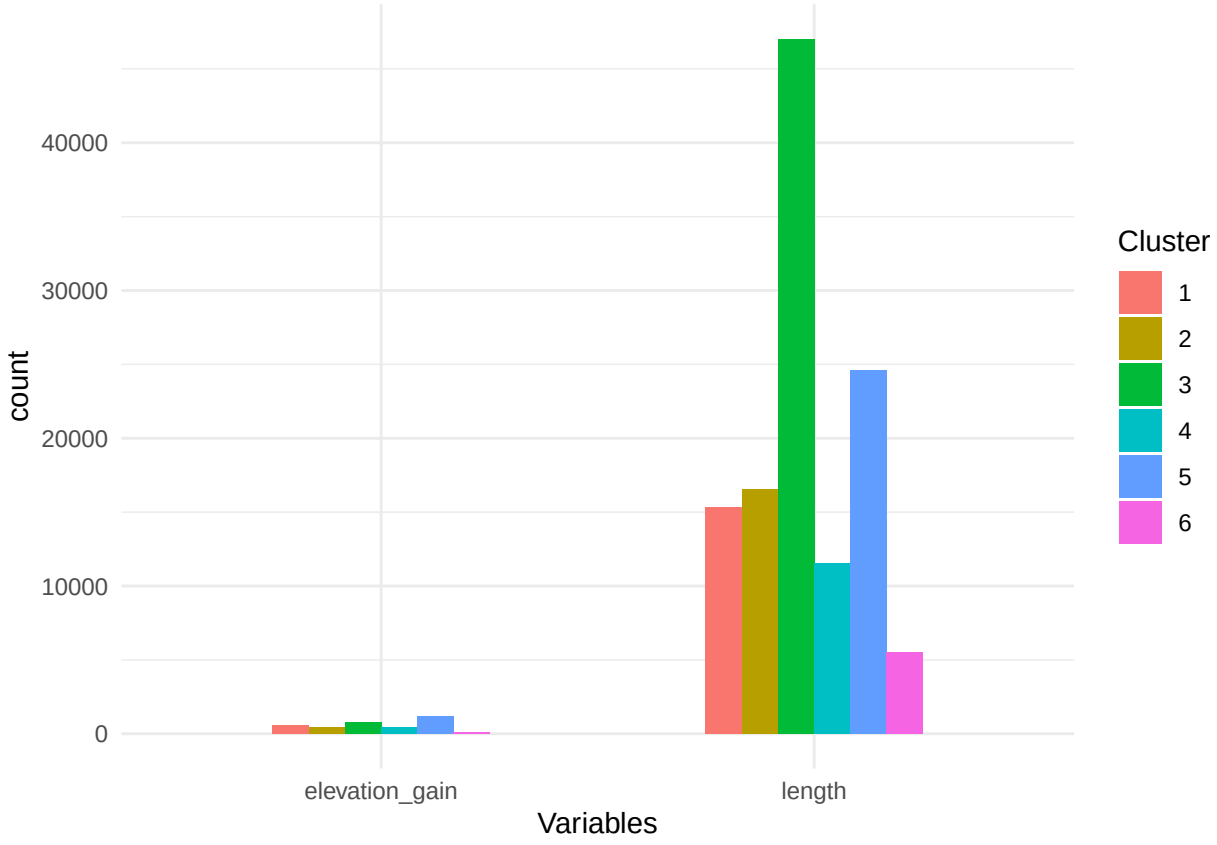
Size of each cluster

```
##
##    1    2    3    4    5    6
##  699  222  134 2408  396 1313
```

Cluster Interpretation of 6 Clusters:

```
##   Group.1    length elevation_gain
## 1       1 15353.196       579.2975
## 2       2 16516.758       464.4012
## 3       3 47025.155       804.3399
## 4       4 11511.459       441.0636
```

```
## 5       5 24600.956      1217.3142
## 6       6  5518.455       133.3555
```



We can see that 6 clusters with sizes of different characterizations:

- **Cluster1:** Strenuous Trails
- **Cluster2:** Recreation Trails
- **Cluster3:** OHV Trails
- **Cluster4:** Moderate Trails
- **Cluster5:** Hard Trails
- **Cluster6:** Easy Trails

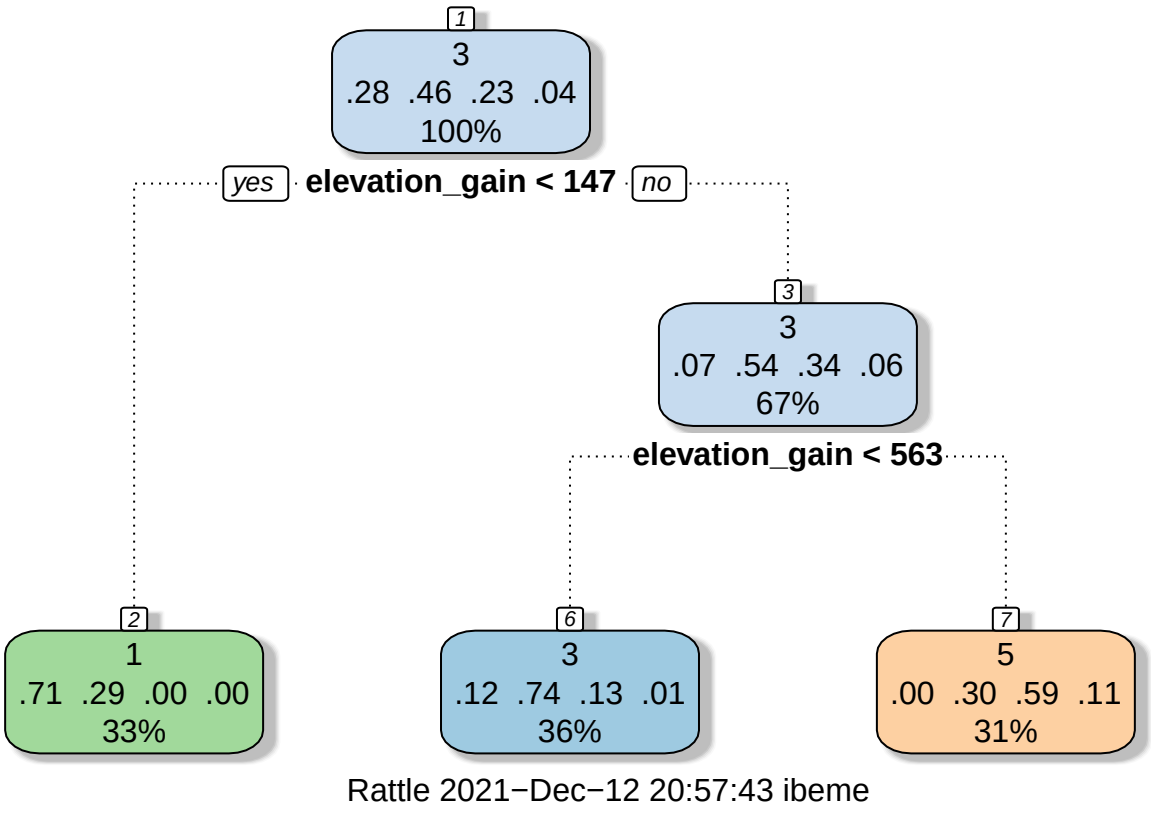
Approach 2: Using Decision Tree Model

To find the threshold values of length or Elevation gain in that hiking trails cluster, we are using Decision tree model.

```
#install.packages("rattle")


Model1<-rpart(difficulty_rating ~ length+elevation_gain,data=Trail_clust4,method='class')

fancyRpartPlot(Model1)
```



It divided into 3 leaf nodes considering elevation gain as a parameter and this values can be used during trail recommendation

IV.II Trail Recommendation:

It is one of the use case of Trail segmentation and lets recommend the similar trails using the threshold values from the decision tree.

Here, a distance matrix is created using the cluster centroids and the user inputs. Once the cluster that is most similar is chosen, another distance matrix is used to determine which hikes in the cluster are closest. The recommended hike is a random choice of the top three most similar hikes.

```
    #Extracted Elevation_gain from the user input
    test_gain<-AllTrails_US_NA[1,10]

    pt<-Trail_data_norm[,1:4]
    #user input
    pt[1,]
```

```
##         length elevation_gain        driving   water_sports
##    -0.05475518    -0.48976906     4.60357755    -0.28485391
```

```
    # Calculate a distance matrix using cluster centroids and the user input

    closest = as.matrix(get_dist(rbind(pt[1,],k4$centers)))[-1,1]

    # Choosing the most similar cluster

    clust_num = which(closest %in% min(closest))


    # Calculate  a distance matrix to determine which hikes in the cluster are closest.

    similar = pt[k4$cluster == clust_num, ]
    likely = as.matrix(get_dist(rbind(pt[1,], similar)))[-1,1]
    names(likely) = row.names(similar)

    # Extracting only top 5 to 6 columns
    top<-head(sort(likely))
    output<-AllTrails_US_NA[as.numeric(names(top)), ]

    final_result<- output[,c(2,5,9,10)]
    colnames(final_result)<- c("Trail_Name","State","Length(mts)","Elevation(mts)")

    if(clust_num==4){
      if(test_gain <= 147.00){
      cat("These are Easy Trails.\nJust relax with a small stroll and enjoy bird watching.\nHere are so
        final_result
    }else if( test_gain >= 147.00 & test_gain<=563.00){
       cat("You have selected the Moderate Trails.\nThese are mainly for hiking,bird watching and to en
      final_result
    }else
cat("Hard and Challenging Trails.Here are some of the Hard trails.\nDont forget to backpack and carry l
      final_result
    }else if(clust_num==2){
        cat("These are Recreation Trails.\nThese are primarily used for fun activities such as paddle s
      final_result

    }else if(clust_num==3){
          cat("These are OHV trails.\nThese are mainly for Off-Road Driving,Road-Biking,Scenic Driving
      final_result

    }else
cat("These are Strenous Trails and Here are some Similar trails.\nThese are designed for hiking,backpac
```

```
## These are OHV trails.
## These are mainly for Off-Road Driving,Road-Biking,Scenic Driving and Bike Touring.
## Here are some of the similar trails you would like to try.
```

```
##                     Trail_Name      State Length(mts) Elevation(mts)
## 1         Cades Cove Loop Road  Tennessee    16254.33       210.9216
## 1890      Cades Cove Loop Road  Tennessee    16254.33       210.9216
## 92    Cactus Forest Loop Drive    Arizona    16576.20       213.9696
## 600    Brooklyn Mine OHV Trail California    16737.14       268.8336
## 2425        Sovereign OHV Loop       Utah    16898.07       282.8544
## 2243 Capitol Reef Scenic Drive       Utah    12713.79       215.7984
```

```
      final_result
```

```
##                     Trail_Name      State Length(mts) Elevation(mts)
## 1         Cades Cove Loop Road  Tennessee    16254.33       210.9216
## 1890      Cades Cove Loop Road  Tennessee    16254.33       210.9216
## 92    Cactus Forest Loop Drive    Arizona    16576.20       213.9696
## 600    Brooklyn Mine OHV Trail California    16737.14       268.8336
## 2425        Sovereign OHV Loop       Utah    16898.07       282.8544
## 2243 Capitol Reef Scenic Drive       Utah    12713.79       215.7984
```

V. Conclusion:

In a nutshell, I would like to conclude that Wanderlust Bundle will give two scenarios

- **Trail Segmentation:** in which the entire dataset has been segmented into 6 clusters which can be used by the business based on the use cases.
- **Trail Recommendation:** is one such use case of the segmentation where similar trails are recommended along with their designated activities based on the user specified trails.

VI. Final Thoughts:

So many of our daily decisions are made on the basis of recommendations. The list may go on and on, whether it's food, restaurants, movies, or retail merchandise. It's also crucial to know that what is being recommended to us will be something that we will have a high confidence that we know we will like or enjoy. Wanderlust Bundle was created to help people identify user-friendly paths and to save lot of time. If I had more time and data to develop Wanderlust Bundle, I could have put some additional data into the project that could have been useful. Trail user information may have been integrated during the data collection procedure. Additional types of recommender models, such as an Item Similarity Recommender, may have been explored if user ratings had been available. Additionally, weather data points for each trail may have been collected and added as model elements and more interested in creating interactive website for the user input.