Capstone_MachineLearning

Jennifer Klein

3/29/2018

Applying Machine Learning to Capstone Project

Now that my data has been cleaned, wrangled and explored, it's time to perform some analysis using machine learning.

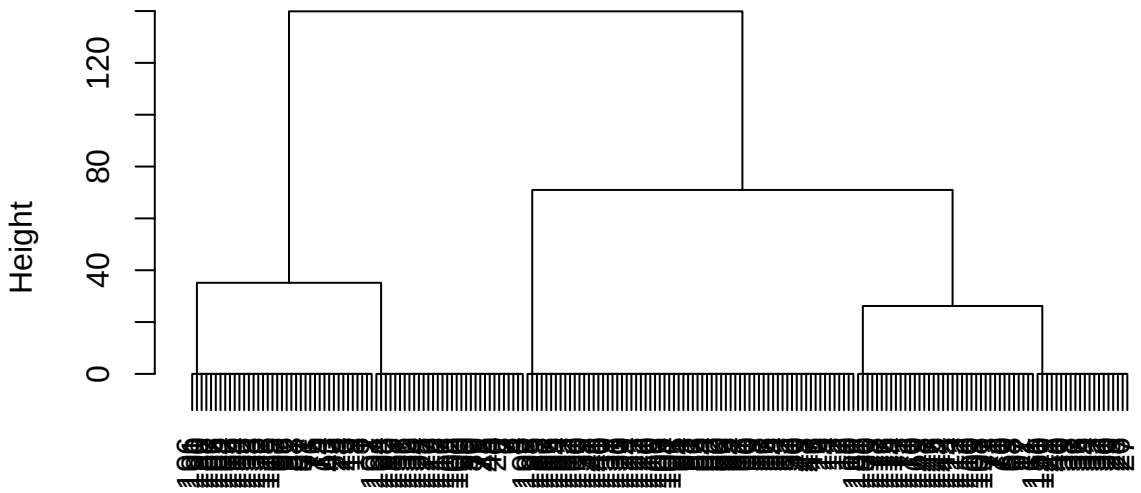
Using the Popularity Rating system I devised earlier, I decided to use those rankings as a way to make a loose recommendation system for podcast episodes. Figuring that a listener would be eager to find more episodes that received the same rating as the episode they just enjoyed, it looks like both collaborative filtering and clustering could work well here.

Clustering is an unsupervised learning method, where I'll try to segment the data into similar groups, instead of trying to predict an outcome. Clustering can also be used to improve predictive methods. And by using Hierarchical clustering, I created a dendrogram to decide how many clusters to use in my final model.

```
library(readr)
Podcast_Dataset <- read_csv("~/Documents/Podcast_Dataset.csv")
```

```
## Parsed with column specification:
## cols(
##   Podcast = col_character(),
##   `Episode Name` = col_character(),
##   `Running Time` = col_integer(),
##   `Release Date` = col_integer(),
##   `Episode Description` = col_character(),
##   `Popularity Rating` = col_integer()
## )
```

```
distances <- dist(Podcast_Dataset[ ,6], method = "euclidean")
clusterPodcasts <- hclust(distances, method = "ward.D")
plot(clusterPodcasts)
```



According to the dendrogram, it looks like 4 clusters would be a good choice when making episode recommendations.

```
clusterGroups <- cutree(clusterPodcasts, k = 4)
tapply(Podcast_Dataset$`Popularity Rating`, clusterGroups)
```

```
##   [1] 1 1 1 2 2 2 2 2 3 3 3 3 3 3 3 3 4 1 1 2 4 4 1 1 1 4 4 4 4 4 4 4 4 4 4
##  [36] 4 4 4 4 4 4 4 1 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 4 4 4 4
##  [71] 4 4 4 1 1 1 1 2 2 3 1 3 3 3 3 3 1 1 2 2 3 1 3 3 3 1 1 1 2 2 3 2 2 3 3
## [106] 3 4 4 4 4 4 4 4 4 4 1 2 2 2 3 3 3 3 3 3 1 2 2 2 3 3 3 1 1 2 2 2 3 1 3 3
## [141] 3 3 4 4 4 4 4 4 4 4 4 4 4 1 1 1 1 2 2 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4
## [176] 4 4 4 4 4 4 4 4 1 1 1 1 1 1 2 1 1 2 2 2 3 1 2 3 3
```

```
cluster1 <- subset(Podcast_Dataset, clusterGroups == 1)
cluster1[1:6]
```

```
## # A tibble: 39 x 6
##    Podcast  `Episode Name` `Running Time` `Release Date` `Episode Descrip~
##    <chr>    <chr>                  <int>          <int> <chr>
##  1 Oprah's~ "ALANIS MORIS~            33              8 "Grammy award-wi~
##  2 Oprah's~ Shauna Niequi~            39              8 Does the life yo~
##  3 Oprah's~ Brene Brown P~            35              8 What is the real~
##  4 Oprah's~ Mitch Albom: ~            38              8 It's been twenty~
##  5 This Am~ DIY                       60              8 After four lawye~
##  6 Stuff Y~ How Personali~            60              8 For millennia, w~
##  7 Stuff Y~ SYSK Selects:~            37              8 In this week's S~
##  8 Stuff Y~ A Lip-Smackin~            54              8 Barbecue, or for~
##  9 Revisio~ The  Basement~            30              8 What is a son's ~
## 10 Dan. Ca~ The Celtic Ho~           359              8 Julius Caesar is~
## # ... with 29 more rows, and 1 more variable: `Popularity Rating` <int>
```

We can see that cluster 1 has the episodes rated 5, cluster 2 has the episodes rated 4, cluster 3 has the episodes rated 3 and cluster 4 has the episodes rated 2 or 1. We can then take a subset of episodes that are only rated 5, make a new dataset with only those episodes and pull recommendations just from that set.

Recommendation systems can definitely be an advantage to companies trying to reach consumers in an increasingly fractured entertainment marketplace. An excellent recommendation system can provide a company with the leg up they need. Clustering algorithms can certainly go a long way to help reach your intended audience.