STAT228 Final Project

Anna Tedeschi

2024-05-04

Predicting My Roommates Skips per Spotify Listening Session Based on Listening Length

Background Information

My roommate , Sydney , has a passion for music and the emotions that they convey. So much so that she requested her Spotify data for the past 6 years. For this project specifically we will be looking at the years of 2021-2023. During these years it spans the end of her senior year to end of her junior year fall semester. I chose to work with this data set because there was the ability to research and do many visualizations with this data set. I also liked that idea of being able to give my roommate a better insight of her spotify data because of how much music means to her.

Research Question : Can I predict how many skips will be listening session based on its length?

Lets start by loading in our data

```
show_col_types = FALSE
sydspotify <- read_csv("~/Desktop/finalproject/Streaming_History_Audio_2021-2023_0.csv")
```

```
## Rows: 16241 Columns: 21
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (14): username, platform, conn_country, ip_addr_decrypted, user_agent_d...
## dbl   (2): ms_played, offline_timestamp
## lgl   (4): shuffle, skipped, offline, incognito_mode
## dttm  (1): ts
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Then we need to clean our data.

```
# impute missing values in the "skipped" column with the mode
mode_skipped <- names(sort(table(sydspotify$skipped), decreasing = TRUE))[1]
sydspotify$skipped[is.na(sydspotify$skipped)] <- mode_skipped
```

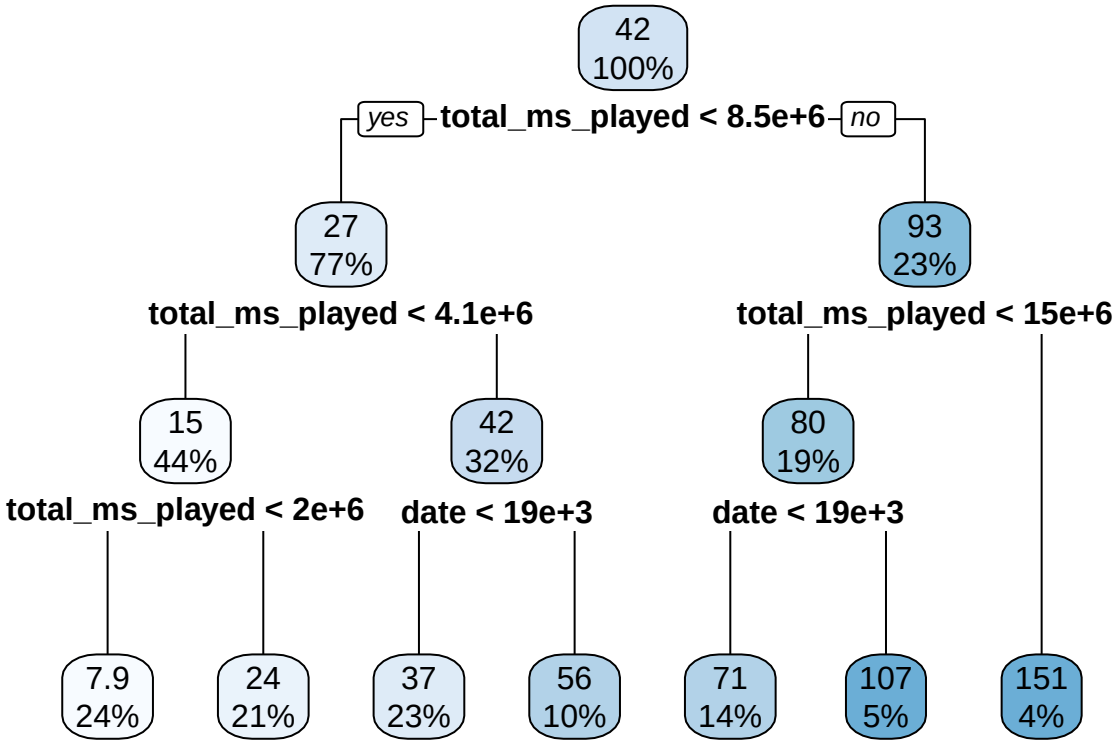
```r
# Convert "skipped" into a binary factor so we can use it
sydspotify$skipped <- factor(sydspotify$skipped == "TRUE", levels = c(FALSE, TRUE))

# convert "skipped" to numeric so we can do math with it
sydspotify$skipped <- as.numeric(sydspotify$skipped)

# get date from "ts" variable, this will be turned into milliseconds so we will be able to compare the
sydspotify$ts <- as.POSIXct(sydspotify$ts)  # convert to POSIXct format
sydspotify$date <- as.Date(sydspotify$ts)   # getting the  date component

# aggregating time data by day
daily_aggregated_data <- sydspotify %>%
  group_by(date) %>%
  summarise(
    total_ms_played = sum(ms_played),
    total_skips = sum(skipped)
    #this is summing the skipped and milliseconds played
  )

# Create a training index
set.seed(123) # for reproducibility
train_index <- sample(1:nrow(daily_aggregated_data), 0.8 * nrow(daily_aggregated_data)) # 80% for train
train_data <- daily_aggregated_data[train_index, ]
test_data <- daily_aggregated_data[-train_index, ]
# here we are test and training our index for our model
```

Interpreting our model and outcome

```r
#creating a model to use
tree_model <- rpart(total_skips~., data = train_data, method = "anova")
tree_model
```

```
## n= 336
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 336 580274.000  42.154760
##    2) total_ms_played< 8532544 258  90470.780  26.872090
##      4) total_ms_played< 4060990 149  17023.250  15.496640
##        8) total_ms_played< 2023485 79   2590.734   7.873418 *
##        9) total_ms_played>=2023485 70   4660.300  24.100000 *
##      5) total_ms_played>=4060990 109  27810.590  42.422020
##       10) date< 19278.5 77   8120.675  36.935060 *
##       11) date>=19278.5 32  11793.500  55.625000 *
##    3) total_ms_played>=8532544 78 230228.200  92.705130
##      6) total_ms_played< 1.455035e+07 64  60908.860  79.953120
##       12) date< 19343.5 48  19214.980  70.979170 *
##       13) date>=19343.5 16  26231.750 106.875000 *
##      7) total_ms_played>=1.455035e+07 14 111336.000 151.000000 *
```

```
# Plot the decision tree
rpart.plot(tree_model)
```




```
#data provided by sydney gonyea and spotify
```

The decision tree model splits the data based on the length of the song, represented in milliseconds. The model starts by splits the data set based on whether the length of the song is less than 75.498 minutes(after conversion from milliseconds). For listening sessions shorter than this threshold, the model predicts an average response variable value of approximately 42 skips for that session. For sessions longer than 75.498 minutes, the average skips for that sessions value increases significantly to approximately 93 ,and so on and so forth.

From this interpretation, we learn that listening session length plays a significant role in predicting the predicted skips value. Shorter sessions tend to have less skips, while longer sessions tend to have more skips per sessions. This suggests that Sydney may have different listening behaviors or preferences based on the length of the sessions they are playing.Therefor, understanding this relationship can help in tailoring recommendations or personalized experiences for users based on her song length preferences.