My iTunes Dataset

Content Page

1. Introduction

1.1 Description

iTunes is where I get my music from. Using a paid family monthly subscription like that of Spotify, I am able to listen to unlimited music. On the iTunes app, I am able to retrieve very interesting pieces of information. Below is a snapshot of the dataset in its raw form.

Name		Time	Album by Artist	Artist	Plays	Year	Genre	Last Played	Release Date	Size
50 Ways to Say Goodbye		4:08	California 37	Train	19	2012	Pop	14/5/19, 7:46 AM	13/4/12	8.8 MB
25 Rasul		5:05	Brotherhood	Raihan	2	1997	Pop	6/8/19, 7:43 AM	10/12/97	11.1 MB
24K Magic		3:47	24K Magic	Bruno Mars	3	2016	Pop	4/11/17, 7:51 PM	7/10/16	7.7 MB
22		3:52	Red (Deluxe Versi...	Taylor Swift	10	2012	Country	15/10/18, 8:08 PM		8.2 MB
21 Guns		5:21	21st Century Brea...	Green Day	1	2009	Alternative	22/9/19, 3:25 PM	12/5/09	10.9 MB
20 ans		3:36	L'attente (Deluxe V...	Johnny Hallyday	1	2012	Rock	1/6/19, 11:01 AM	12/11/12	7.4 MB
7-й элемент (Седьмой эл...		4:05	Майский шашлын...	Vitas	2	2001	Pop	14/6/18, 8:42 PM	1/1/01	8.3 MB

Unfortunately, I don't know how to extract this as a csv. So I copy-pasted everything in to a txt file. It kinda looks like this:

```
50 Ways to Say Goodbye	4:08	California 37	Train	19	2012	Pop 14/5/19, 7:46 AM	13/4/12 8.8 MB
25 Rasul	5:05	Brotherhood Raihan	2	1997	Pop 6/8/19, 7:43 AM 10/12/97	11.1 MB
24K Magic	3:47	24K Magic	Bruno Mars	3	2016	Pop 4/11/17, 7:51 PM	7/10/16 7.7 MB
22	3:52	Red (Deluxe Version)	Taylor Swift	10	2012	Country 15/10/18, 8:08 PM		8.2 MB
21 Guns	5:21	21st Century Breakdown (Deluxe Version) Green Day	1	2009	Alternative 22/9/19, 3:25 PM	12/5/09 10.9 MB
20 ans	3:36	L'attente (Deluxe Version)	Johnny Hallyday 1	2012	Rock	1/6/19, 11:01 AM	12/11/12	7.4 MB
7-й элемент (Седьмой элемент)	4:05	Майский шашлындос. Любимые песни	Vitas	2	2001	Pop 14/6/18, 8:42 PM	1/1/01	8.3 MB
7 Years	3:57	Lukas Graham	Lukas Graham	3	2015	Pop 19/11/17, 5:15 PM	16/6/15 8.1 MB	1
7 Years	3:53	7 Years - Single	LittleTranscriber	2016	Pop	16/3/16 7.8 MB	1
6 at Best	3:38	Out of Time - EP	First to Eleven 1	2016	Pop 27/6/19, 7:33 AM	28/4/16 7.4 MB
2U (feat. Justin Bieber)	3:15	2U (feat. Justin Bieber) - Single	David Guetta	1	2017	Dance	1/9/17, 6:33 PM 9/6/17	6.7 MB
青花瓷	3:57	On the Run	Jay Chou	6	2007	Mandopop	12/4/19, 12:24 PM	2/11/07 8.2 MB
随它吧 (Chinese Mandarin Version)	3:45	Let It Go the Complete Set (From "Frozen")	Jalane Hu	12	2014	Soundtrack	11/9/19, 7:11 AM	15/4/14 7.6 MB
醉赤壁	4:38	林俊傑2003年-2010年作品精選集	JJ Lin	2008	Mandopop	2/10/08 9.3 MB
遇见 (feat. Jeremy Monteiro)	4:28	遇见 (feat. Jeremy Monteiro) - Single	Joanna Dong 2	2017	Contemporary Jazz	12/8/19, 10:43 AM	14/4/17 9 MB
追光者 (電視劇《夏至未至》插曲)	3:56	追光者 (電視劇《夏至未至》插曲) - Single	Yoyo Sham	2017	Mandopop	16/6/17 8 MB
轉眼 (2018自傳最終章)	6:01	轉眼 (2018自傳最終章) - Single	Mayday	1	2018	Mandopop	11/4/19, 6:46 PM	31/12/18	12.2 MB
說好的幸福呢	4:15	魔杰座	Jay Chou	1	2008	Mandopop	8/8/19, 10:11 AM	14/10/08	8.6 MB
```

After that, it was easy, since every entry is separated by tabs. So I used this code to convert the text into a dataset:

```r
raw_itunes <- read.delim('Arcturus/itunes.txt', header = FALSE, sep = "\t", dec = ",")
names(raw_itunes) <- c('title','downloaded','duration','album','artist','plays','year','genre','last_pla
raw_itunes$downloaded<-NULL
```

(I removed the 'downloaded' column because it is useless..)

1.2 Purpose

Honestly, I've always wanted to use this dataset for something. Looking at the variables, its hard to find a tangible quanitity to predict. But I wanted to know what songs are of my liking. So I ended up choosing to predict the 'Plays' variable. So, given other columns, I would like to predict how many times I've played the song. Hopefully, using this model, I am able to create a spider that finds songs that suit me. But that's a project for another time.

1.3 Variables

Printed below are the variables found in the dataset:

```r
names(raw_itunes)
```

```
##  [1] "title"        "duration"     "album"        "artist"       "plays"
##  [6] "year"         "genre"        "last_played"  "release_date" "size"
## [11] "skips"
```

title (note: LaTeX doesn't support chinese characters so I am unable to show all the titles..)

Self-explanatory, contains the title of the song. The important thing to note here is that the title sometimes contains the names of supporting artists. After much exploring, I found out that the only places where artists can be found in the title is after the 'feat.' word. Extracting these names is for another section:

```r
as.vector(raw_itunes$title[1:5])
```

```
## [1] "À la plus haute branche"
## [2] "À peu près"
## [3] "The A Team"
## [4] "The a Team"
## [5] "Abe Lincoln vs Chuck Norris (feat. Nice Peter & Epiclloyd)"
```

duration

This represents the duration, in minutes, of the song:

```r
as.vector(raw_itunes$duration[1:10])
```

```
##  [1] "4:50" "3:26" "4:42" "4:18" "2:08" "2:01" "2:28" "4:24" "3:44" "4:56"
```

album

The name of the album that the song is from. Ended up not using it:

```r
as.vector(raw_itunes$album[1:5])
```

```
## [1] "Encore un soir"
## [2] "À peu près"
## [3] "Cover Sessions, Vol. 2"
## [4] "+"
## [5] "Abe Lincoln vs Chuck Norris (feat. Nice Peter & Epiclloyd) - Single"
```

artist

The name of the artist(s) that created / performed the song. Sometimes contains multiple artistes, separated by either commas or the ampersand(&) sign. Same as title in this regard:

```r
as.data.frame(raw_itunes %>% group_by(artist) %>%
            summarize(n=n()) %>% arrange(desc(n)))[1:4,]
```

```
##                        artist   n
## 1               Boyce Avenue 118
## 2           LittleTranscriber 100
## 3             Rucka Rucka Ali  89
## 4 Epic Rap Battles of History  81
```

plays

Our objective. This records the number of times I've listened to this song:

```
as.data.frame(raw_itunes %>% arrange(desc(plays)) %>% select(title, plays))[1:4,]
```

```
##                    title plays
## 1             The Nights   106
## 2 Down (feat. Lil Wayne)    98
## 3      Rewrite the Stars    96
## 4     Alexander Hamilton    95
```

genre

This records the genre that the song belongs in. Note that there are many different types of genres recorded, with some values missing:

```
as.data.frame(raw_itunes %>% group_by(genre) %>%
            summarize(n=n()) %>% arrange(desc(n)))[1:4,]
```

```
##                genre   n
## 1                Pop 784
## 2             Comedy 253
## 3 Singer/Songwriter 202
## 4        Alternative 167
```

last_played

This records the last time I've played this song. It comes as a string. If it is blank, it means I haven't played the song before:

```
as.vector(raw_itunes$last_played)[1:10]
```

```
##  [1] "4/10/18, 1:10 PM"  "14/1/20, 6:37 PM"  "2/3/19, 10:37 AM"
##  [4] "29/11/17, 4:44 AM" "4/11/19, 6:47 PM"  "3/9/19, 7:13 AM"
##  [7] "9/8/19, 12:36 PM"  "24/6/19, 7:21 AM"  "24/7/19, 6:51 PM"
## [10] "3/12/19, 8:37 PM"
```

release_date

This records the date that the song is released. It also comes as a string. Similar to last_played, there are missing values:

```
as.vector(raw_itunes$release_date)[1:10]
```

```
##  [1] "26/8/16"  "6/10/17"  "2/1/12"   "7/2/10"   "15/12/11" "8/2/13"
##  [7] "17/9/13"  "6/11/15"  "17/5/18"  "8/4/82"
```

size

This records the file size of the song, aka how much space it takes up on my phone:

```
as.vector(raw_itunes$size)[1:10]
```

```
##  [1] "9.7 MB"  "7.1 MB"  "9.3 MB"  "8.7 MB"  "4.7 MB"  "4.4 MB"  "5.3 MB"
##  [8] "8.9 MB"  "8.5 MB"  "10.6 MB"
```

skips

This records the number of times I've skipped the song, aka my annoyance with the song.

```
as.data.frame(raw_itunes %>% arrange(desc(skips)) %>% select(title, skips))[1:4,]
```

```
##       title skips
## 1     Hello     9
## 2 All of Me     7
## 3     Sorry     7
## 4 Good Time     6
```

1.4 Key Steps

- Things I need to do:
  - Convert 'last_played' and 'release_date' into datetime strings
  - Extract artist names from 'title' and 'artist
  - Convert 'duration' and 'size' into their appropriate units
  - Clean data
  - Replace empty data with fake ones

2. Methods / Analysis

2.1 Cleaning the data

This data has a few issues. Firstly, there are quite a few missing values.

Missing values in skips, plays, genre

For the numeric ones like 'skips' and 'plays', missing values are due to zeroes. For 'genre', after some searching, I've found that missing values are all 'Instrumental' in nature. So using this code, I am able to replace them:

```
raw_itunes$plays[is.na(raw_itunes$plays)] <- 0
raw_itunes$skips[is.na(raw_itunes$skips)] <- 0
raw_itunes$genre[raw_itunes$genre==''] <- 'Instrumental'
```

Reassigning genre

There are an unnecessarily large number of genres in this dataset, including the low-count ones like:

```
  as.data.frame(raw_itunes %>% group_by(genre) %>%
            summarize(n=n()) %>% arrange(n))[1:4,]
```

```
##                genre n
## 1 Adult Contemporary 1
## 2          Bollywood 1
## 3          Brazilian 1
## 4       Chinese Rock 1
```

As such, I needed a way to reassign the genres to encompass a smaller variety. I used this code to map the old to the new genres:

```
unique_genres <- unique(raw_itunes$genre)
genres_that_i_want <- c('Non-Asian','Asian','Pop','Jazz','Dance-ish','Rock-ish','For Kids',
                        'Soundtrack','Rap','Covers','Instrumental','Alternative','Comedy','Misc')
assigned_genres <- genres_that_i_want[c(1,10,13,5,12,11,6,1,7,3,
                                        5,9,8,11,3,12,3,12,2,14,
                                        11,5,6,11,14,1,4,14,9,3,
                                        9,6,3,2,1,1,5,11,14,6,
                                        4,9,2,4,2,11,2,2,1,8,
                                        3,1,2,4)]

genre_map <- setNames(assigned_genres,unique_genres)
raw_itunes$genre <- genre_map[as.vector(raw_itunes$genre)]
```

So now, the least common genres have the following numbers of songs:

```
as.data.frame(raw_itunes %>% group_by(genre) %>%
            summarize(n=n()) %>% arrange(n))[1:4,]
```

```
##        genre  n
## 1       Jazz  4
## 2      Asian 10
## 3 Soundtrack 33
## 4   For Kids 41
```

Reformatting duration

The 'duration' column needs to be reformatted. It is currently of the format 'MM:SS', where the song is MM minutes and SS seconds long. So, I created a new column, labelled 'time_in_seconds' which records the duration in terms of seconds:

```
raw_itunes$time_in_seconds <- sapply(as.vector(raw_itunes$duration), function(x){
  as.numeric(strsplit(x,':')[[1]][1])*60+as.numeric(strsplit(x,':')[[1]][2])
})
# Remove invalid / zero length durations
raw_itunes$time_in_seconds[is.na(raw_itunes$time_in_seconds)] <- 0
raw_itunes <- subset(raw_itunes, time_in_seconds!=0)
```

So now, we can see the longest songs:

```
as.data.frame(raw_itunes %>% arrange(desc(time_in_seconds)) %>% select(title, time_in_seconds))[1:4,]
```

```
##                                                                                 title
## 1 Four_Chord_Mega-Medley_Alan_Walker_Imagine_Dragons_Avril_Lavigne_The_Script__more
## 2                                                                              H20901
## 3                                                                         Albuquerque
## 4           Trapped In the Drive-Thru (Parody of Trapped In the Closet By R. Kelly)
##   time_in_seconds
## 1            1627
## 2            1560
## 3             683
## 4             651
```

Reformatting size

The 'size' column also needs to be reformatted. Currently, it is either of the form 'x KB' or 'x MB'. I want to convert everything into a standard format. So I used the following code to convert everything to KB as a new column:

```
library(tidyr)
raw_itunes$kb <- sapply(as.vector(raw_itunes$size),function(x){
  if (grepl('MB',x)) {
    as.numeric(gsub(' MB','',x))*1000
  } else {
    as.numeric(gsub(' KB','',x))
  }
})
```

Reformatting dates

'release_date' and 'last_played' are still in string format, which is not very useful. So, I used this code to convert them into datetime formats:

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```
raw_itunes$clean_release_date <- as.Date(raw_itunes$release_date, format='%d/%m/%y')
raw_itunes$clean_last_played <-as.Date(raw_itunes$last_played, format='%d/%m/%y, %I:%M %p')
```

Faking the data

I found that the two dates columns mentioned have quite alot of missing data. At first I thought that this wasn't an issue. But later on I found that this causes alot of problems as I will have very few predicting variables. As such, I decided to create fake data. I first retrieve the means and sd of each column (non-NAs) and then I use rtruncnorm to generate new dates that follow the distribution. Somehow, it manages to work. So for example, the average and sd of 'clean_release_date' are shown below:

```
print(mean(raw_itunes$clean_release_date[!is.na(raw_itunes$clean_release_date)]))
```

```
## [1] "2013-01-28"
```

```
print(sd(raw_itunes$clean_release_date[!is.na(raw_itunes$clean_release_date)]))
```

```
## [1] 2469.313
```

Further cleaning of release date

I also found out another problem. For the release date, the format was %d/%m/%y, which meant that the year was reported as double digits. As such, some songs early in the 20th century were wrongly reported as being in the 21st century:

```
raw_itunes %>% filter(clean_release_date > Sys.Date()) %>% select(title, clean_release_date)
```

```
##           title clean_release_date
## 1 The Elements         2053-01-01
## 2 La Cucaracha         2062-01-01
```

So that is fixed with this code:

```
year(raw_itunes$clean_release_date[raw_itunes$clean_release_date>current_date]) <-
+year(raw_itunes$clean_release_date[raw_itunes$clean_release_date>current_date])-100
```

Extracting all artists

The 'artists' column is far from complete. Multiple artists in the same song are separated by commas. Same goes for the title, where the word 'feat.' appears in many songs. As such, I decided that I want to create a separate row for each artist in the song. Firstly, I created the column 'songId' in order to not lose track of the song in question:

```
raw_itunes$songId<-c(1:length(raw_itunes$title))
```

Next is just a sequence of str_split, str_replace_all, gsub, and sapply to get what I want (too long to show). So now my top artist list looks like this:

```
as.data.frame(clean_itunes %>% group_by(artist) %>%
            summarize(n=n()) %>% arrange(desc(n)))[1:4,]
```

```
##                        artist   n
## 1                Boyce Avenue 120
## 2           LittleTranscriber 100
## 3             Rucka Rucka Ali  89
## 4 Epic Rap Battles of History  81
```

(Not much difference, I know, but trust me on this. BTW since this is a major change, I renamed the dataset 'clean_itunes')

2.2 Some theories

I have some theories on what variables affect the number of plays in a song.

info_density

First one is info_density, which is how dense the song is. Maybe the denser the music, the more I like it?? So I used the 'kb' and 'time_in_seconds' to create this column.

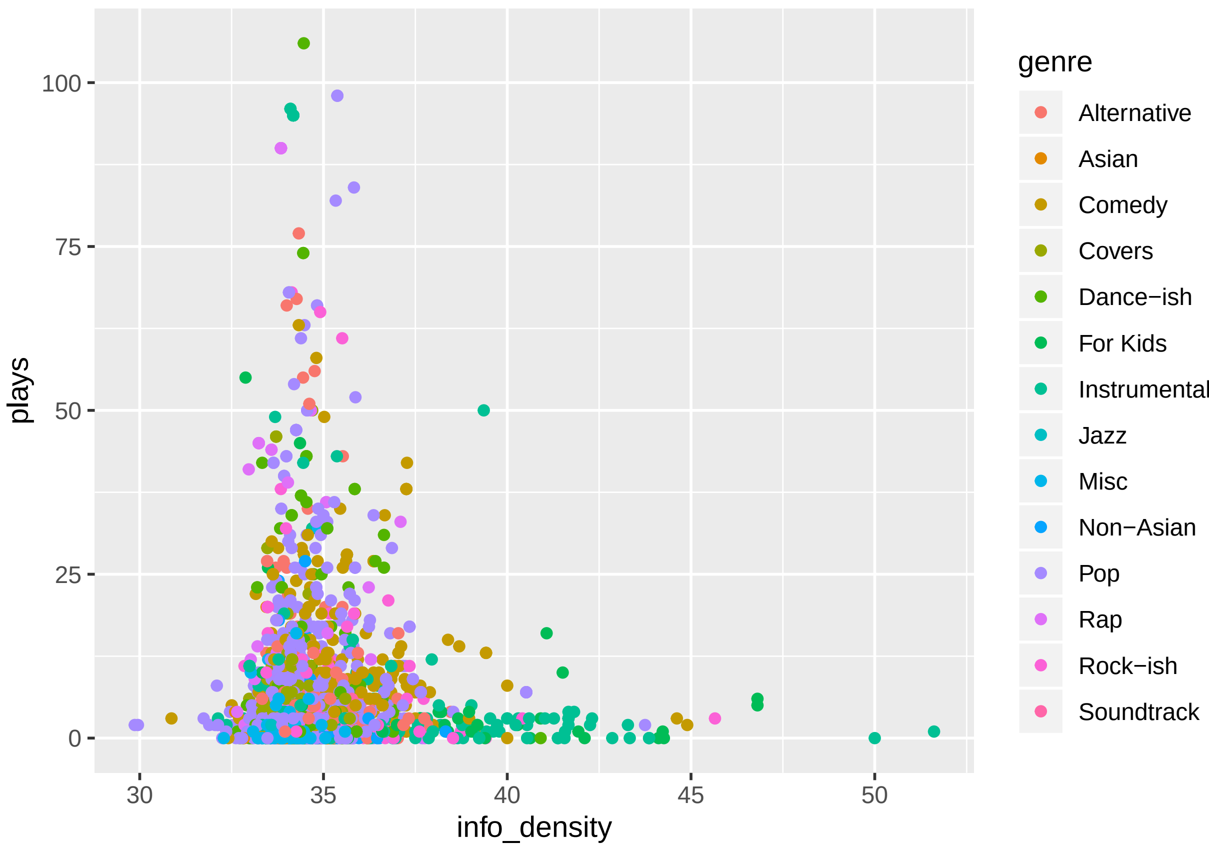
```
clean_itunes <- clean_itunes%>%mutate(info_density = kb/time_in_seconds)
```

And now, let's look at the correlation coefficient and the distribution of the info_density:

```
library(ggplot2)
print(cor(clean_itunes$plays,clean_itunes$info_density))
```

```
## [1] -0.01827507
```

```
clean_itunes %>% group_by(genre) %>% filter(info_density<100) %>%
ggplot(aes(x=info_density,y=plays,color=genre))+geom_point()
```



Doesn't quite work, but I'll still include it.

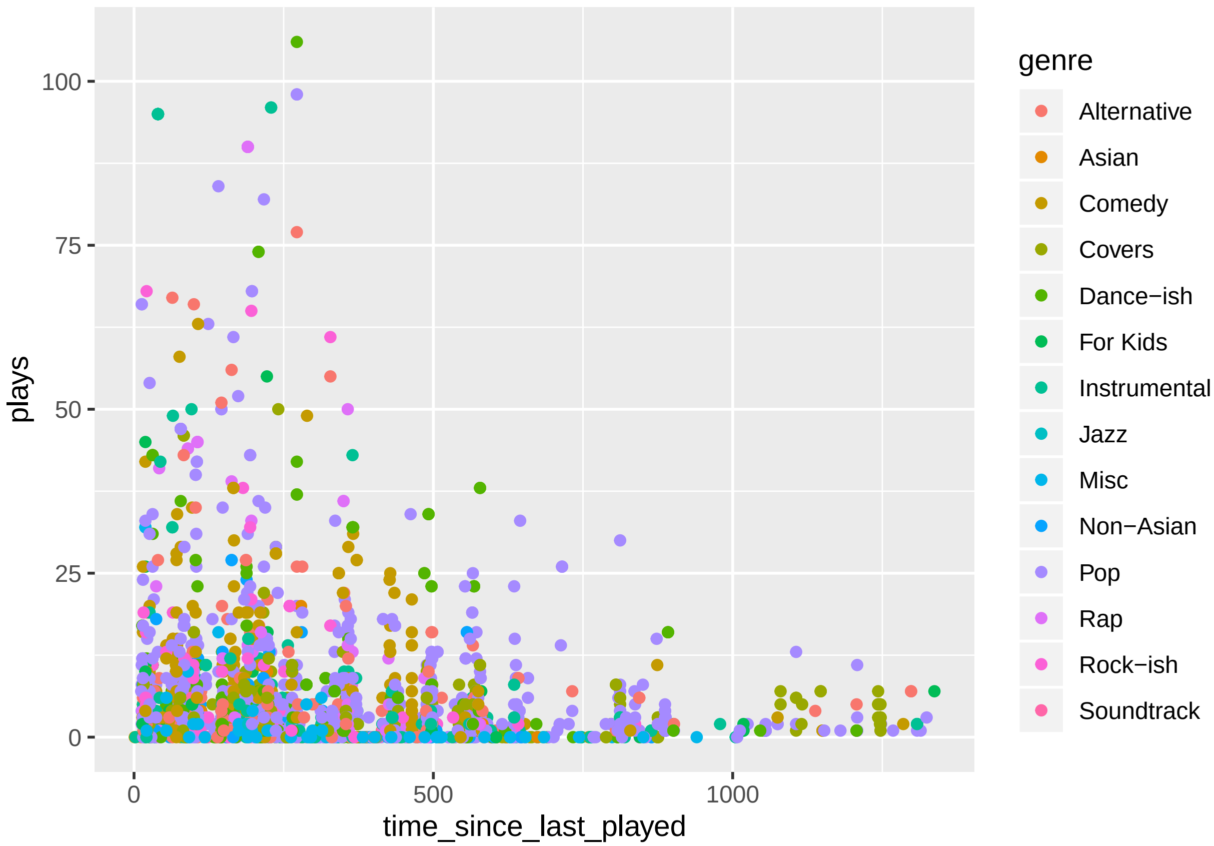
time_since_last_played

If it's been a long time since I played a song, it usually means that i rarely play it:

```
clean_itunes$time_since_last_played <-
  as.numeric(current_date) - as.numeric(clean_itunes$clean_last_played)
print(cor(clean_itunes$time_since_last_played,clean_itunes$plays))
```

```
## [1] -0.2007519
```

```
clean_itunes %>% group_by(genre) %>%
ggplot(aes(x=time_since_last_played,y=plays,color=genre))+geom_point()
```



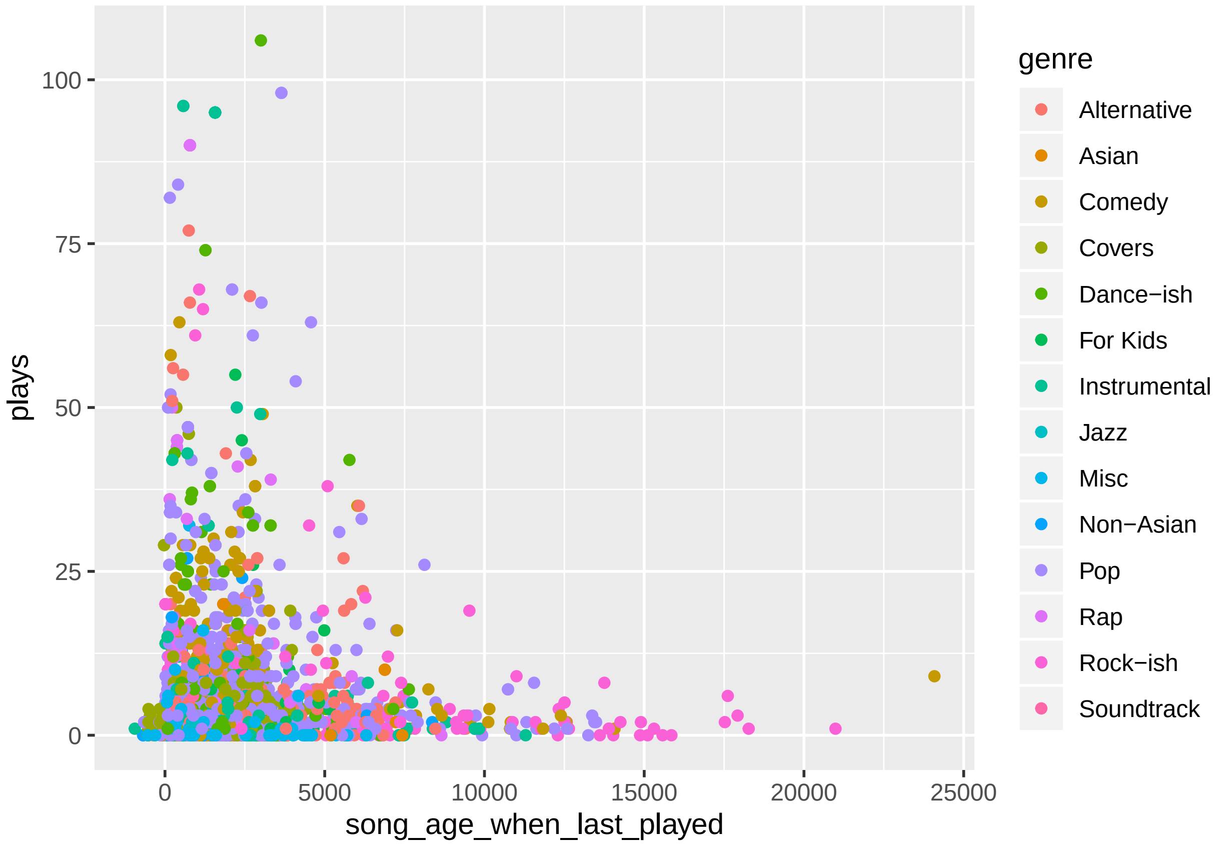
song_age_when_last_played

As the name suggests:

```
clean_itunes$song_age <- as.numeric(current_date) - as.numeric(clean_itunes$clean_release_date)
clean_itunes$song_age_when_last_played <- clean_itunes$song_age - clean_itunes$time_since_last_played
print(cor(clean_itunes$song_age_when_last_played,clean_itunes$plays))
```

```
## [1] -0.06286705
```

```
clean_itunes %>% group_by(genre) %>%
ggplot(aes(x=song_age_when_last_played,y=plays,color=genre))+geom_point()
```



Variables bound to artist

These are variables I can only define within the train set because I'm not supposed to use test data as predictors to avoid overfitting. They are created by first grouping by artist and then do some other operations:

artist_time_spent

Finds out how much time in total I've spent listening to the artist. sum(plays*time_in_seconds)

artist_plays

How many times I've played one of their songs. sum(plays)

artist_skips

How many times I've skipped one of their songs. sum(skips)

artist_total_songs

How many of their songs I have. n()

artist_avg_time_spent

Average time spent on their songs. mean(plays*time_in_seconds)

artist_avg_plays

Average number of times I've played their songs. mean(plays)

artist_avg_skips

Average number of times I've skipped their songs. mean(skips)

Variables bound to genre

Exactly the same as that for artist.

2.3 Modelling approach

After doing the cleaning, I separated the data into an initial 0.9,0.1 split using createDataPartition. A train_set and a validation set. I used a sapply loop to loop through a vector 1:100. Within that loop, from the train_set, I separated that further into a 0.9,0.1 split. An itunes set and a test_set. I performed the artist-wise and the genre-wise operations on the itunes set. Similar to the cor() functions I've done earlier, I then found out what variables are most correlated to the number of plays, choosing the best 10 as predictors:

```
df_of_cors <- as.data.frame(cor(itunes[sapply(itunes,is.numeric)],itunes$plays))
names(df_of_cors) <- c('R')
df_of_cors$variable <- row.names(df_of_cors)
variables_to_train <- df_of_cors[order(-abs(df_of_cors$R)),][2:11,]$variable
```

I then used three different training models from the caret package:

```
TrainData <- itunes %>% select(variables_to_train)
TrainClasses <- itunes$plays
model_1 <- train(TrainData,TrainClasses,method='glm',family='gaussian',
  tuneLength=10,trControl=trainControl(method='cv'))
model_2 <- train(TrainData,TrainClasses,method='glm',family='quasipoisson',
  tuneLength=10,trControl=trainControl(method='cv'))
model_3 <- train(TrainData,TrainClasses,method='gamLoess',
tuneLength=10,trControl=trainControl(method='cv'))
```

After that, I merged the artist-wise and genre-wise data from the itunes dataset into the test_set. So now, the test_set has the necessary predictors. After some cleaning, I predicted the test_set plays. I picked the best model out of all of them (best RMSE) and if the RMSE is less than 10, I would use that model to predict the validation plays. And out of all these validation predictions, I averaged them to get my final prediction. This process took like 2 hours to complete so I'm not going to show it here.

3. Results

3.1 RMSE

I experimented with many regression models. Some had issues, other didn't. Some churned out abyssmal RMSEs, others gave adequate ones. Out of the 100 times I've split the data, only 18 of them reported RMSEs less than 10. Quite sad, but I'll have to deal with it. In the end, the final RMSE I got was quite bad, at 12.06523.

3.2 The outliers

I decided to find out what were causing the values to be way off. Using this code, I managed to find out which rows gave the lowest errors:

```
> as.data.frame(validation %>% arrange(error))[1:10,]
                                          title duration                               album          artist plays year        genre
1                               Tunak Tunak Tun     5:03                     Tunak Tunak Tun    Daler Mehndi     2 1998        Asian
2                                  Hotline Bling     2:58          Hotline Bling - Single LittleTranscriber     2 2015          Pop
3                 Lonely Together (feat. Rita Ora)     3:02                       Avīci (01) - EP          Avicii     7 2017    Dance-ish
4                          Don't You Worry Child     4:06                             Wonders  The Piano Guys     4 2014 Instrumental
5                                        Shallow     3:36            A Star Is Born Soundtrack       Lady Gaga     3 2018 Instrumental
6                             You Are the Reason     3:24                 Only Human (Deluxe)     Calum Scott     3 2017          Pop
7  This Is What It Feels Like (feat. Trevor Guthrie)     3:23 Intense (Bonus Track Version)  Trevor Guthrie     1 2013    Dance-ish
8                                         Jumper     1:43                 Community Favorites      Waterflame     4 2015    Dance-ish
9                             Collide (Original)     4:09                 Stop All the World Now       Howie Day     4 2003          Pop
10                                   Wavin' Flag     3:41     Troubadour (Champion Edition)          K'naan     5 2009          Rap
        last_played release_date    size skips time_in_seconds    kb info_density clean_release_date clean_last_played time_since_last_played
1   17/7/18, 5:52 PM     30/9/98 10.4 MB     0             303 10400     34.32343         1998-09-30        2018-07-17                    579
2   15/9/17, 2:19 AM    30/11/15  6.1 MB     0             178  6100     34.26966         2015-11-30        2017-09-15                    884
3   25/7/19, 6:57 AM     10/8/17  6.5 MB     0             182  6500     35.71429         2017-08-10        2019-07-25                    206
4   24/9/19, 7:08 AM     6/10/14  8.4 MB     0             246  8400     34.14634         2014-10-06        2019-09-24                    145
5   12/3/19, 7:00 AM     5/10/18  7.4 MB     0             216  7400     34.25926         2018-10-05        2019-03-12                    341
6  9/10/18, 10:54 AM    17/11/17  6.9 MB     0             204  6900     33.82353         2017-11-17        2018-10-09                    495
7  30/10/16, 9:07 AM      8/4/13  7.2 MB     0             203  7200     35.46798         2013-04-08        2016-10-30                   1204
8    2/3/19, 9:53 AM     14/2/15  3.8 MB     0             103  3800     36.89320         2015-02-14        2019-03-02                    351
9  28/11/19, 9:23 PM     7/10/03  8.7 MB     0             249  8700     34.93976         2003-10-07        2019-11-28                     80
10 14/11/19, 7:25 AM     24/2/09  7.5 MB     0             221  7500     33.93665         2009-02-24        2019-11-14                     94
   song_age song_age_when_last_played songId prediction      error
1      7809                      7230   1934   1.982214 0.01778602
2      1539                       655    776   1.981429 0.01857054
3       920                       714   1089   6.975913 0.02408744
4      1959                      1814    473   4.029127 0.02912650
5       499                       158   1605   3.098127 0.09812660
6       821                       326   2141   3.146440 0.14643966
7      2505                      1301   1849   1.152513 0.15251295
8      1828                      1477    961   3.833532 0.16646756
9      5976                      5896    343   4.174189 0.17418885
10     4009                      3915   1995   5.175586 0.17558569
```

Out of this, I noticed one thing: All the play values are small

Which kinda makes sense I guess, as it is easier to guess a small number more accurately than a large number.

Also, I found out which rows gave the highest errors:

```
> as.data.frame(validation %>% arrange(desc(error)))[1:10,]
                                              title duration                                                  album
1  The Way I Are (Dance With Somebody) [feat. Lil Wayne]     3:08 The Way I Are (Dance With Somebody) [feat. Lil Wayne] - Single
2                                        Last Hurrah     2:30                                     Last Hurrah - Single
3                                           Riff Off     4:49           Pitch Perfect 3 (Original Motion Picture Soundtrack)
4                                      Tour the World     8:34                                             Brain Beats 2
5                                  Scared to Be Lonely     3:41                             Scared to Be Lonely - Single
6                                       Just a Dream     3:58                                                       5.0
7                                          On My Way     3:14                                       On My Way - Single
8                                         Love Story     3:55                              Fearless (Platinum Edition)
9                  Journey Back to You (feat. NerdOut)     3:56                           Young Unprofessionals (Acoustic)
10                 Journey Back to You (feat. NerdOut)     3:56                           Young Unprofessionals (Acoustic)
            artist plays year     genre        last_played release_date    size skips time_in_seconds    kb info_density clean_release_date
1        Lil Wayne     3 2017       Pop    4/2/20, 5:26 PM      19/5/17  6.7 MB     0             188  6700     35.63830         2017-05-19
2      Bebe Rexha    82 2019       Pop   17/7/19, 6:44 PM      15/2/19  5.3 MB     0             150  5300     35.33333         2019-02-15
3        Evermoist    47 2017       Pop   3/12/19, 6:10 PM     15/12/17  9.9 MB     1             289  9900     34.25606         2017-12-15
4  Renald Francoeur    55 2013  For Kids   12/7/19, 6:57 AM       2/7/13 16.9 MB     3             514 16900     32.87938         2013-07-02
5   Martin Garrix    37 2017 Dance-ish   23/5/19, 1:39 PM      27/1/17  7.6 MB     0             221  7600     34.38914         2017-01-27
6            Nelly    39 2010       Rap    9/9/19, 6:58 AM      17/8/10  8.1 MB     1             238  8100     34.03361         2010-08-17
7      Alan Walker    43 2019 Dance-ish   19/1/20, 9:48 PM      21/3/19  6.7 MB     2             194  6700     34.53608         2019-03-21
8     Taylor Swift    30 2009       Pop 29/11/17, 11:44 AM                  8 MB     2             235  8000     34.04255         2013-01-05
9     Ben Schuller    29 2019    Covers   27/6/19, 9:39 AM      26/7/19  7.9 MB     0             236  7900     33.47458         2019-07-26
10         NerdOut    29 2019    Covers   27/6/19, 9:39 AM      26/7/19  7.9 MB     0             236  7900     33.47458         2019-07-26
   clean_last_played time_since_last_played song_age song_age_when_last_played songId prediction     error
1         2020-02-04                     12 1003.000                   991.000   1996 139.370310 136.37031
2         2019-07-17                    214  366.000                   152.000   1007   4.742559  77.25744
3         2019-12-03                     75  793.000                   718.000   1499   6.403177  40.59682
4         2019-07-12                    219 2420.000                  2201.000   1907  17.018560  37.98144
5         2019-05-23                    269 1115.000                   846.000   1567   4.103928  32.89607
6         2019-09-09                    160 3470.000                  3310.000    963   6.169941  32.83006
7         2020-01-19                     28  332.000                   304.000   1318  13.862669  29.13733
8         2017-11-29                    809 2597.333                  1788.333   1121   4.730965  25.26904
9         2019-06-27                    234  205.000                   -29.000    959   3.821549  25.17845
10        2019-06-27                    234  205.000                   -29.000    959   5.025656  23.97434
```

Out of this, I noticed two things: 1. (Almost) All the play values are large 2. I rarely listen to these artists So I guess it kinda makes sense that the model would underestimate most of these songs.

And then I thought to myself: "Only the sith deals in absolutes" so I turned this around and decided to measure errors percentage-wise. I used the following metric as my new error column:

```
validation <- validation %>% mutate(error=abs(plays/prediction-1))
```

And I got these results for the best rows:

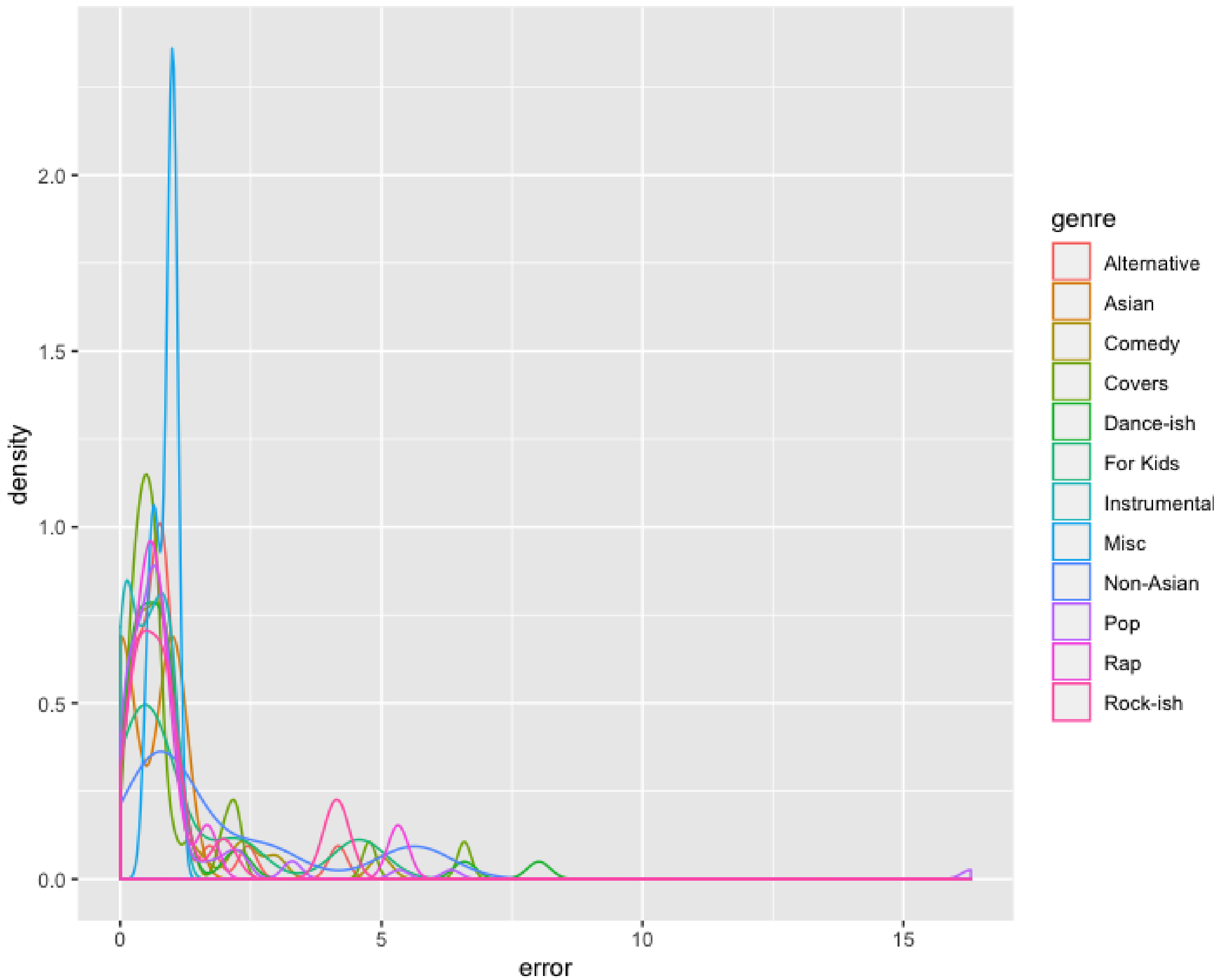
```
> as.data.frame(validation %>% arrange(error))[1:10,]
                          title duration                        album          artist plays year        genre       last_played release_date    size skips
1  Lonely Together (feat. Rita Ora)     3:02              Avici (01) - EP          Avicii     7 2017    Dance-ish  25/7/19, 6:57 AM      10/8/17  6.5 MB     0
2             Don't You Worry Child     4:06                      Wonders   The Piano Guys     4 2014 Instrumental  24/9/19, 7:08 AM      6/10/14  8.4 MB     0
3                   Tunak Tunak Tun     5:03              Tunak Tunak Tun    Daler Mehndi     2 1998        Asian  17/7/18, 5:52 PM      30/9/98 10.4 MB     0
4                     Hotline Bling     2:58       Hotline Bling - Single LittleTranscriber     2 2015          Pop  15/9/17, 2:19 AM     30/11/15  6.1 MB     0
5                           Shallow     3:36     A Star Is Born Soundtrack       Lady Gaga     3 2018 Instrumental  12/3/19, 7:00 AM      5/10/18  7.4 MB     0
6                        Wavin' Flag     3:41 Troubadour (Champion Edition)          K'naan     5 2009          Rap 14/11/19, 7:25 AM      24/2/09  7.5 MB     0
7                 Collide (Original)     4:09          Stop All the World Now       Howie Day     4 2003          Pop 28/11/19, 9:23 PM      7/10/03  8.7 MB     0
8                            Jumper     1:43          Community Favorites      Waterflame     4 2015    Dance-ish   2/3/19, 9:53 AM      14/2/15  3.8 MB     0
9                  You Are the Reason     3:24          Only Human (Deluxe)     Calum Scott     3 2017          Pop 9/10/18, 10:54 AM     17/11/17  6.9 MB     0
10                  My Happy Ending     4:02                 Under My Skin    Avril Lavigne     8 2004          Pop  21/5/19, 6:07 PM      25/5/04  8.6 MB     1
   time_in_seconds    kb info_density clean_release_date clean_last_played time_since_last_played song_age song_age_when_last_played songId prediction       error
1              182  6500     35.71429         2017-08-10        2019-07-25                    206      920                        714   1089   6.975913 0.003452944
2              246  8400     34.14634         2014-10-06        2019-09-24                    145     1959                       1814    473   4.029127 0.007228987
3              303 10400     34.32343         1998-09-30        2018-07-17                    579     7809                       7230   1934   1.982214 0.008972806
4              178  6100     34.26966         2015-11-30        2017-09-15                    884     1539                        655    776   1.981429 0.009372295
5              216  7400     34.25926         2018-10-05        2019-03-12                    341      499                        158   1605   3.098127 0.031672884
6              221  7500     33.93665         2009-02-24        2019-11-14                     94     4009                       3915   1995   5.175586 0.033925763
7              249  8700     34.93976         2003-10-07        2019-11-28                     80     5976                       5896    343   4.174189 0.041729988
8              103  3800     36.89320         2015-02-14        2019-03-02                    351     1828                       1477    961   3.833532 0.043424064
9              204  6900     33.82353         2017-11-17        2018-10-09                    495      821                        326   2141   3.146440 0.046541385
10             242  8600     35.53719         2004-05-25        2019-05-21                    271     5745                       5474   1227   8.411011 0.048865846
```

And these for the worst rows:

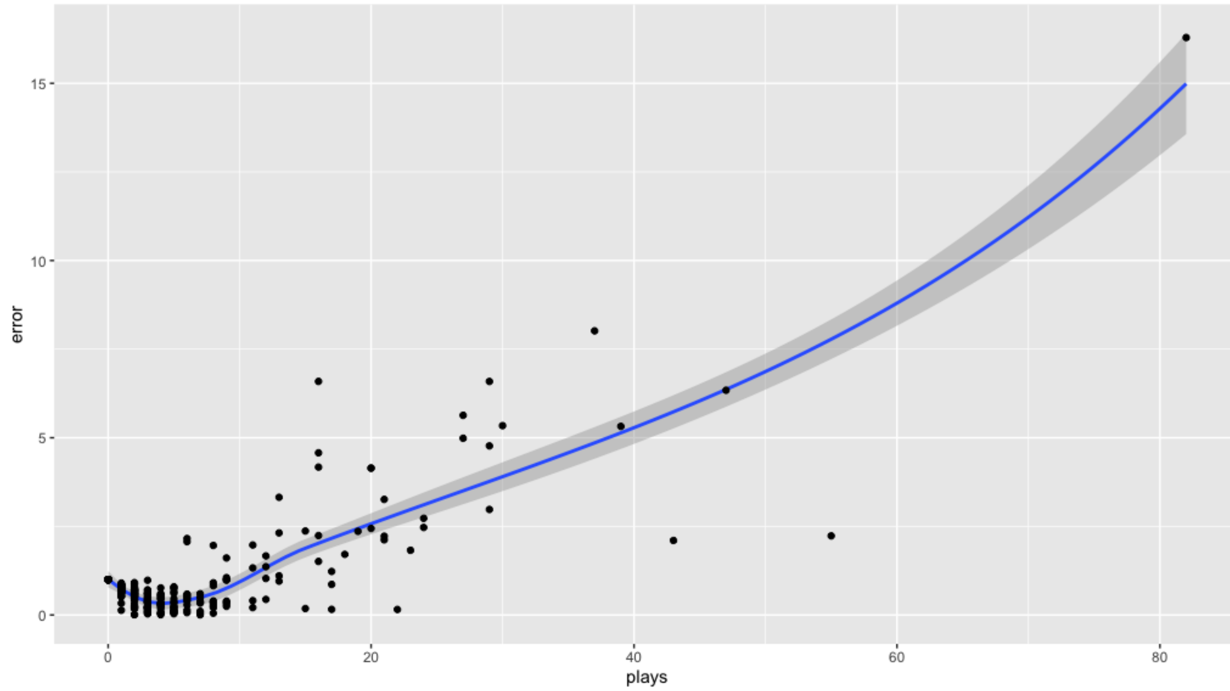
```
> as.data.frame(validation %>% arrange(desc(error)))[1:10,]
                                                  title duration                                                          album          artist plays year
1                                           Last Hurrah     2:30                                            Last Hurrah - Single       Bebe Rexha    82 2019
2                                    Scared to Be Lonely     3:41                                    Scared to Be Lonely - Single    Martin Garrix    37 2017
3                                     Lean On (feat. MØ)     2:57                                          Peace Is the Mission        DJ Snake    16 2015
4                         Journey Back to You (feat. NerdOut)     3:56                                 Young Unprofessionals (Acoustic)     Ben Schuller    29 2019
5                                               Riff Off     4:49               Pitch Perfect 3 (Original Motion Picture Soundtrack)        Evermoist    47 2017
6  The Hardest Karaoke Song in the World (feat. Steindi Jr.)     3:20 The Hardest Karaoke Song in the World (feat. Steindi Jr.) - Single Inspired by Iceland    27 2017
7                                             Love Story     3:55                                      Fearless (Platinum Edition)     Taylor Swift    30 2009
8                                           Just a Dream     3:58                                                             5.0            Nelly    39 2010
9                      Down (feat. DJ Not Nice & Lil Wang)     4:02                                                    Rucka's World      DJ Not Nice    27 2012
10                        Journey Back to You (feat. NerdOut)     3:56                                 Young Unprofessionals (Acoustic)          NerdOut    29 2019
       genre        last_played release_date   size skips time_in_seconds   kb info_density clean_release_date clean_last_played time_since_last_played song_age
1        Pop   17/7/19, 6:44 PM      15/2/19 5.3 MB     0             150 5300     35.33333         2019-02-15        2019-07-17                    214   366.000
2  Dance-ish   23/5/19, 1:39 PM      27/1/17 7.6 MB     0             221 7600     34.38914         2017-01-27        2019-05-23                    269  1115.000
3  Dance-ish  10/9/17, 12:42 PM       2/3/15 6.3 MB     0             177 6300     35.59322         2015-03-02        2017-09-10                    889  1812.000
4     Covers   27/6/19, 9:39 AM      26/7/19 7.9 MB     0             236 7900     33.47458         2019-07-26        2019-06-27                    234   205.000
5        Pop   3/12/19, 6:10 PM     15/12/17 9.9 MB     1             289 9900     34.25606         2017-12-15        2019-12-03                     75   793.000
6  Non-Asian    9/9/19, 7:21 PM     14/10/17 6.9 MB     0             200 6900     34.50000         2017-10-14        2019-09-09                    160   855.000
7        Pop 29/11/17, 11:44 AM                 8 MB     2             235 8000     34.04255         2013-01-05        2017-11-29                    809  2597.333
8        Rap    9/9/19, 6:58 AM      17/8/10 8.1 MB     1             238 8100     34.03361         2010-08-17        2019-09-09                    160  3470.000
9     Comedy   12/2/19, 8:05 PM      11/9/12 8.8 MB     0             242 8800     36.36364         2012-09-11        2019-02-12                    369  2714.000
10    Covers   27/6/19, 9:39 AM      26/7/19 7.9 MB     0             236 7900     33.47458         2019-07-26        2019-06-27                    234   205.000
   song_age_when_last_played songId prediction     error
1                    152.000   1007   4.742559 16.290243
2                    846.000   1567   4.103928  8.015753
3                    923.000   1022   2.108041  6.589984
4                    -29.000    959   3.821549  6.588546
5                    718.000   1499   6.403177  6.340106
6                    695.000    712   4.070530  5.633043
7                   1788.333   1121   4.730965  5.341202
8                   3310.000    963   6.169941  5.320968
9                   2345.000    483   4.511004  4.985364
10                   -29.000    959   5.025656  4.770391
```

3.3 General trend

Here is the distribution of the errors:



,which shows that the majority of the play values were estimated quite well, especially for the miscellaneous genres. However, the Non-Asian songs showed much greater errors, probably because I don't listen to it often. Here is another plot:



,which shows the general trend that the more I listen to a song, the higher the error is for the song. This is also shown by the correlation coefficient of 0.7911089 between the error and the number of plays.

4. Conclusion

4.1 Summary

I wanted to find out if there was a way to predict the number of times I listen to a song based on all the other variables in my itunes dataset. I found out that while it was possible, the predictions are way off. This effect is severely compounded for one-hit wonder songs, aka when I only have one song from a particular artist. Other than that, in hindsight, many of the variables that I have been using are not optimal. What I intended at the start was to create a spider program that will use my model to pick out potential songs that I would listen to. However, when I use the 'skips' and 'last_played' columns, the big assumption would be that I have already listened to the song, beating the whole purpose of the model. ## 4.2 Potential Impact My original intention flawed, I can't really think of any other use for this model, other than to compare it with my friends'. Kinda shameful that songs appearing in the dataset are 'Tunak Tunak Tun' and 'Big and Chunky'. ## 4.3 Limitations Already mentioned ## 4.4 Future work I'll try to find a more universal dataset that will improve my prediction model. Reduce my reliance on historical data ('skips','last_played', etc..). And once I re-run this model on this newfound dataset, I will create my spider program. Apart from this I guess I can just compile as many friends' datasets as I can to find out what songs of theirs I would like, but seeing as though people rarely use Apple Music, this is not a viable option.