CIS361 Report

Nick Sheehan

5/5/2020

## SUMMARY

The problem I am trying to address is the equality of pay across the different genres of YouTube. Is each genre generating the same kind of revenue for the highest viewed/subscribed channels? If there is a difference, is it because of the nature of the genre, the demographic, or something else? These are the questions I will be asking myself and finding the answer for with this project. This is important because equality of pay is an issue throughout any industry, but I think especially in YouTube as it becomes more of a known source of a way to make income. I would like to see that there is relatively equal pay and no gross differences but these are important to point out if found. That way people can start to think of ways to improve the inequality.

There aren’t too many articles on my topic but the few I found had some interesting points. One being that the videos in overall more popular genres will get more views and in turn make more money. Since YouTube pays mostly by CPM (Clicks per Mille), which basically means an amount for every thousand views the ads on their videos get. Another good point that was brought up was video types. Some video types tend to get more views than others, thus meaning more money.

A few research questions I am asking myself are: - Which genres are the most viewed? - Does higher views have a direct impact on interactions? (Likes, Dislikes, and Comments) - Based off the two above, which genres are generating the most revenue

I believe there will be some pretty large differences in the salaries among the genres. I also would expect to see genres tailored towards the younger demographic having the highest salaries. I would also expect to see the genres with more family friendly content will generate more revenue compared to genres with 18+ material.

## DATA

First thing I needed to do was enable all the packages i will be using.

#Enable Packages  
library(tuber)  
library(purrr)  
library(stringr)  
library(dplyr)  
library(shiny)  
library(rmarkdown)  
library(ggplot2)  
library(knitr)

Then I needed to source my data. Since i was using information from YouTube, I needed to be able to hit their API’s to collect data. Luckily, they offer some public API’s that you can access after setting up an account with Google Developers Console. I was able to activate these APIs from my project and get my token using Auth 2.0. Tuber offers the function yt\_oauth that allows me to do this.

#Authorization for Youtube APIs  
client\_id <- "871403720599-cjqncpljsnghb9las2sr0s87rdtm62ko.apps.googleusercontent.com"  
client\_secret <- "UV1qXjK9TmVB9k6MzQLoJsjT"  
yt\_oauth(app\_id = client\_id,  
 app\_secret = client\_secret,  
 token = '')

Next, I needed to figure out the best way to collect the data from YouTube to fit my needs. What i decided to do, was to create a playlist for each main genre on YouTube and put the trending videos from each category in it. Then I was able to use Tuber to pull my playlist, and strip out the video Ids. I had to re-use this for each genre, and I could not place into a loop because I needed to save to different variables every iteration.

#Get playlist Id for Music playlist  
NickFBPlaylistId <- str\_split(  
 string = "https://www.youtube.com/playlist?list=PL1SoOwjbBbc-rh4v7ciKTtIcOyOpT11ov",   
 pattern = "=",   
 n = 2,  
 simplify = TRUE)[ , 2]  
#Put videos in playlist into data frame  
NickFBPlaylist <- get\_playlist\_items(filter =   
 c(playlist\_id = NickFBPlaylistId),   
 part = "contentDetails",  
 max\_results = 10)  
#Save list of video ids  
FB\_ids <- base::as.vector(NickFBPlaylist$contentDetails.videoId)

I then proceeded to collect both the stats and details for all videos within these playlists. I did this through the Tuber functions get\_stats and get\_video\_details. I then took those results and put them into appropriate lists and data frames. All the Data I really needed was in the stats of the videos, except for the video Title. So that is why I had to get the details for each video too.

#Function for get all stats on vids  
get\_all\_stats <- function(id) {  
 get\_stats(video\_id = id)  
}   
#Get all stats for vid ids in that playlist  
NickFBStats <- map\_df(.x = FB\_ids, .f = get\_all\_stats)  
#get all details on vids  
NickFBDetails <- lapply(FB\_ids, get\_video\_details)  
NickMusicDetails <- lapply(music\_ids, get\_video\_details)  
NickGamingDetails <- lapply(gaming\_ids, get\_video\_details)  
NickMoviesDetails <- lapply(movies\_ids, get\_video\_details)

Next, I looped through each details list to get the Id and Title for each video. I placed this into its own data frame as I plan to merge this to the stats data frames later. I had to loop through each details list for each genre this way.

#Pull Id and Title from details  
FBDetailsDF <- data.frame(id=NA, Title=NA)  
for (p in 1:length(NickFBDetails)) {  
 id <- NickFBDetails[[p]]$items[[1]]$id  
 Title <- NickFBDetails[[p]]$items[[1]]$snippet$title  
   
 FBdetail <- data\_frame(id = id, Title = Title)  
 FBDetailsDF <- rbind(FBdetail, FBDetailsDF)  
}

Finally, I merged my Titles with the stats data frames to make a complete data frame for each genre with the data I need.

#Merge details to stats data frame  
NickMusicAllData <- merge(NickMusicStats,MusicDetailsDF,by = "id")  
NickGamingAllData <- merge(NickGamingStats,GamingDetailsDF,by = "id")  
NickMoviesAllData <- merge(NickMovieStats,MoviesDetailsDF,by = "id")  
NickFBAllData <- merge(NickFBStats,FBDetailsDF,by = "id")

## METHODOLOGY/RESULTS

Here is where I start to graph some of my data to get an idea of how it is looking. I made a simple bar plot representing Total view counts per genre.

#Graph showing differences in views by main genres  
options(scipen=999)  
MusicGenreViewCount <- sum(as.numeric(NickMusicAllData$viewCount), na.rm = TRUE)  
GamingGenreViewCount <- sum(as.numeric(NickGamingAllData$viewCount), na.rm = TRUE)  
MoviesGenreViewCount <- sum(as.numeric(NickMoviesAllData$viewCount), na.rm = TRUE)  
FBGenreViewCount <- sum(as.numeric(NickFBAllData$viewCount), na.rm = TRUE)  
Genres <- c("Movies/Entertainment","Music","Gaming","Fashion/Beauty")  
ViewCounts <- c(MoviesGenreViewCount,MusicGenreViewCount,GamingGenreViewCount,FBGenreViewCount)  
BarGraphDF <- data.frame(Genres,ViewCounts)  
ggplot(data=BarGraphDF, aes(x=Genres,y=ViewCounts))+  
 geom\_bar(stat="identity", fill="steelblue")+  
 geom\_text(aes(label=ViewCounts), vjust=1.6, color="white",size=3.5)+  
 theme\_minimal()

Using the same format, I created three more bar graphs to investigate the interactions the videos in each genre recieve. Interactions are Likes, Dislikes, and comments. These data points help judge how active a community is.

#Generate Bar Plot for Likes and Disklikes for each genre  
MusicLikeCount <- sum(as.numeric(NickMusicAllData$likeCount), na.rm=TRUE)  
GamingLikeCount <- sum(as.numeric(NickGamingAllData$likeCount), na.rm=TRUE)  
MoviesLikeCount <- sum(as.numeric(NickMoviesAllData$likeCount), na.rm=TRUE)  
FBLikeCount <- sum(as.numeric(NickFBAllData$likeCount), na.rm=TRUE)  
LikeCounts <- c(MoviesLikeCount,MusicLikeCount,GamingLikeCount,FBLikeCount)  
LikesBarGraphDF <- data.frame(Genres,LikeCounts)  
ggplot(data=LikesBarGraphDF, aes(x=Genres,y=LikeCounts))+  
 geom\_bar(stat="identity", fill="steelblue")+  
 geom\_text(aes(label=LikeCounts), vjust=1.6, color="white",size=3.5)+  
 theme\_minimal()  
MusicDislikeCount <- sum(as.numeric(NickMusicAllData$dislikeCount), na.rm=TRUE)  
GamingDislikeCount <- sum(as.numeric(NickGamingAllData$dislikeCount), na.rm=TRUE)  
MoviesDislikeCount <- sum(as.numeric(NickMoviesAllData$dislikeCount), na.rm=TRUE)  
FBDislikeCount <- sum(as.numeric(NickFBAllData$dislikeCount), na.rm=TRUE)  
DislikeCounts <- c(MoviesDislikeCount,MusicDislikeCount,GamingDislikeCount,FBDislikeCount)  
DislikesBarGraphDF <- data.frame(Genres,DislikeCounts)  
ggplot(data=DislikesBarGraphDF, aes(x=Genres,y=DislikeCounts))+  
 geom\_bar(stat="identity", fill="steelblue")+  
 geom\_text(aes(label=DislikeCounts), vjust=1.6, color="white",size=3.5)+  
 theme\_minimal()  
# Bar graph for comments per genre  
MusicCommentCount <- sum(as.numeric(NickMusicAllData$commentCount), na.rm=TRUE)  
GamingCommentCount <- sum(as.numeric(NickGamingAllData$commentCount), na.rm=TRUE)  
MoviesCommentCount <- sum(as.numeric(NickMoviesAllData$commentCount), na.rm=TRUE)  
FBCommentCount <- sum(as.numeric(NickFBAllData$commentCount), na.rm=TRUE)  
CommentCounts <- c(MoviesCommentCount,MusicCommentCount,GamingCommentCount,FBCommentCount)  
CommentsBarGraphDF <- data.frame(Genres,CommentCounts)  
ggplot(data=CommentsBarGraphDF, aes(x=Genres,y=CommentCounts))+  
 geom\_bar(stat="identity", fill="steelblue")+  
 geom\_text(aes(label=CommentCounts), vjust=1.6, color="white",size=3.5)+  
 theme\_minimal()

## CONCLUSION

The results seem to show that the more views you have, the more interactions you have. When you you are talking about how much money people are making off videos, you have to take a couple things into account. The genre, views, and the interactions on the videos. Interactions are important because the more ineractions you get the more noticable your video is on YouTube. Music is by far in the lead in all categories but what was suprising to me was how strong gaming is on Youtube. It is easily second, but produces a larging interaction percentage then even the music genre. By these numbers, it definately seems like what you get paid will be quite different between genres. There is not much equality when it comes to pay on YouTube at this point, but hopefully strides toward this will be made soon.

## Resources

Packages I used:

Tuber: <https://cran.r-project.org/web/packages/tuber/index.html>  
Purrr: <https://cran.r-project.org/web/packages/purrr/index.html>  
Stringr: <https://cran.r-project.org/web/packages/stringr/index.html>  
Shiny: <https://cran.r-project.org/web/packages/shiny/index.html>  
RMarkdown: <https://cran.r-project.org/web/packages/rmarkdown/index.html>  
Dplyr: <https://cran.r-project.org/web/packages/dplyr/index.html>  
ggplot2: <https://cran.r-project.org/web/packages/ggplot2/index.html>  
knitr: <https://cran.r-project.org/web/packages/knitr/index.html>