STA141A - Project

Trending Youtube Data Analysis

12/10/2021

Nathan Liu (njlliu@ucdavis.edu)
Contribtuion: Data Cleaning, Elementary Statistics, Hierarchical Clustering, Function

Minh-Tu Nguyen (minngu@ucdavis.edu)
Contribution: Data Description, Data Visualization, PCA, RMD Editor

Vici Thahir (vthahir@ucdavis.edu)
Contribution: Introduction, Research Questions, Linear Analysis, Interpretation, Conclusion

Introduction

Youtube is a free online video-sharing platform by Google. It is the second most visited website after Google, with more than one billion users monthly who collectively watch more than one billion hours worth of videos every day [1]. With that huge size of data to store, Youtube keeps a list that is updated every 15 minutes of the most trending videos [2]. Some criterias for a trending video are the following:

- Must be appealing to a wide range of viewers,
- Cannot be misleading, clickbaity, or sensational,
- Must capture the breadth of what is currently happening in Youtube or in the world,
- Must showcase a diversity of content creators,
- Ideally, are surprising and novel.

In compiling a list of trending videos, Youtube considers several parameters: users interaction (i.e. number of views, number of likes, number of comments), how fast the videos generate views (i.e. "temperature"), how do viewers know about this video (i.e. inside or outside Youtube), the age of the video, how the videos perform compared to other recent uploads from the same channel

This project will specifically focus on the influence of user interaction on the top trending Youtube videos. The section below will explain the scope of the project and the dataset used. The purpose of this study is to better understand and quantify the relationships of some numerical variables of user interaction and the top-trending Youtube videos.

Data Description

The original dataset [3] consists of the top-trending Youtube videos from five different regions: USA, Great Britain, Germany, Canada, and France. However, for relevancy, we will be focusing on the USA video dataset. This dataset includes several months (and counting) of data on daily trending YouTube videos. There are multiple categorical, numerical, and boolean variables. Such categorical variables contain "title", "channel_title", "publish_time", "category_id", etc. "category_id" is a given number that corresponds to a specific category (i.e. video blogging or science & technology) that is listed in a given json file. Some numerical variables are "views", "likes", "dislikes", and "comment_count". Boolean variables consist of "comments_disabled", "ratings_disabled", etc. With these column variables, we can find correlations and explore various relationships that appear in the top 200 trending videos in the USA.

Research Questions

1. Which numerical category (i.e. dislikes, comments, likes) affects view counts the most? Is there a correlation between these numerical categories?
2. What time of the day is the best time to post a video?
3. Which video category is the most popular?
4. How can the video categories be grouped based on their user interaction?

Data Cleaning

We began our project by cleaning the data set. Since the original data set contained about 40,000 observations of videos spanning between November, 2017 to June, 2018, we decided to limit the scope to just the month of May, 2018 by finding the substring of the "trending_date" category and parsing it for dates in May, 2018. Then, we removed all the columns that would seem irrelevant to our study. These columns included the video title, tags, thumbnail_link, description, comments_disabled (logical), ratings_disabled (logical), and video_error_or_removed (logical). These columns wouldn't tell us much about data trends. We also decided to remove all the rows with NA's. Next, we changed all the category ID's into their respective strings. These strings represent the category name, and this information is found in the .json file on the Kaggle dataset page. We used the sapply() function to switch every ID to their respective category (ex: "1" = "Film and Animation") so plotting the dataset with respect to category ID would be more informative. Finally, we isolated the hour substring of the publishing time column. This column was originally presented in a very congregated format that included the year, month, date, hour, minute, and second during which the video was posted. Doing so generated a column with cleaner values for the publishing time.

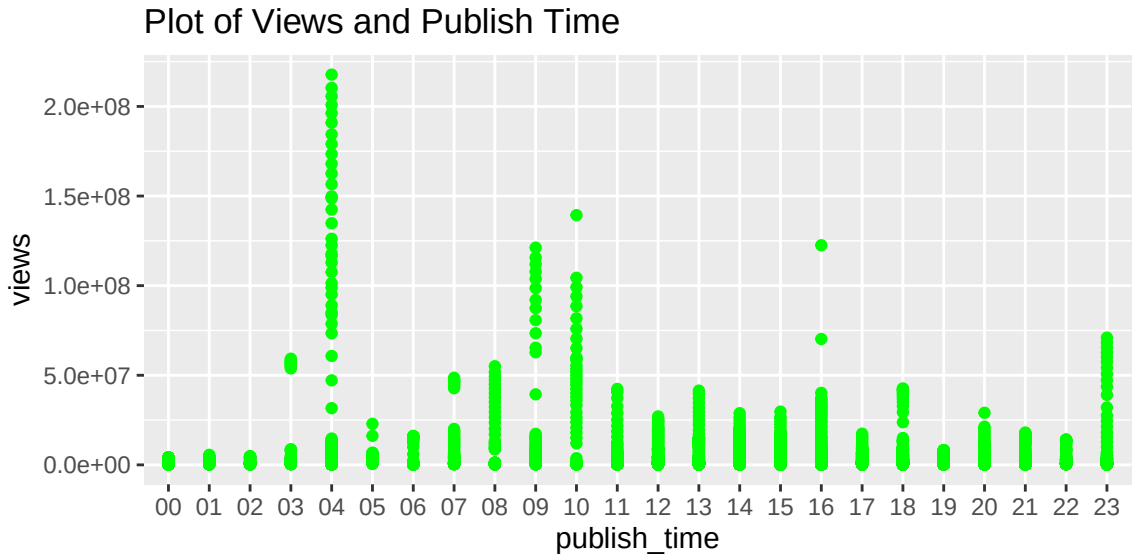
Elementary Statistics

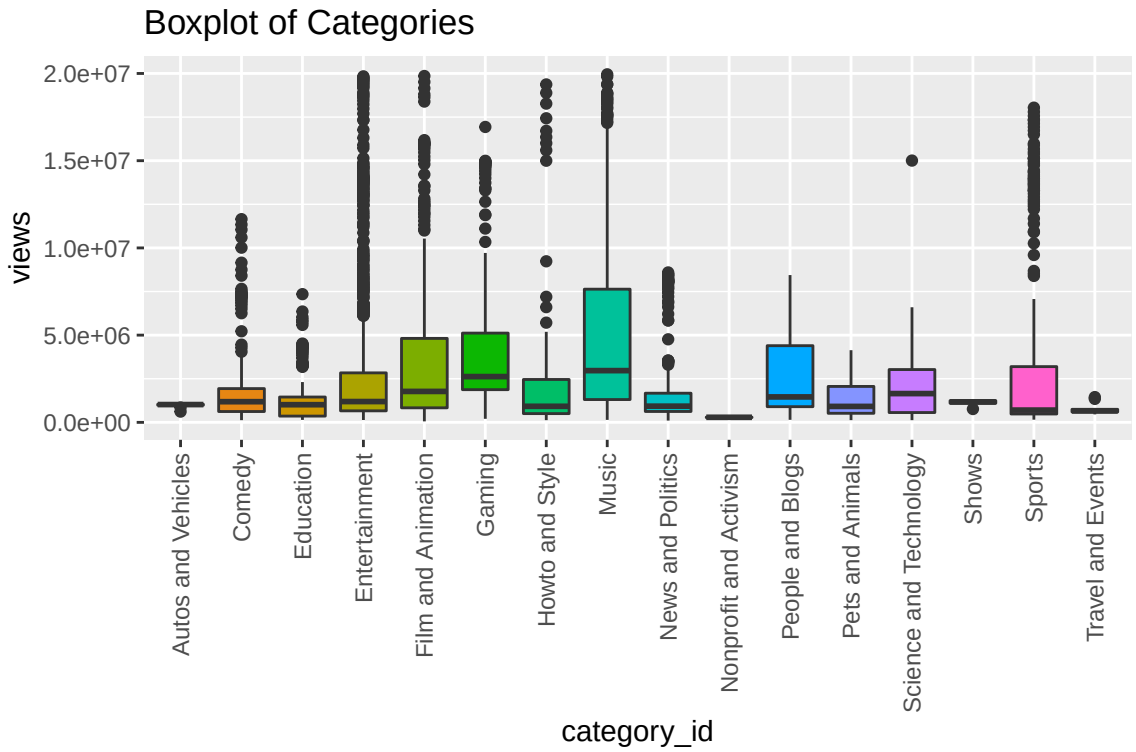
Categorical Data (channel title, category ID): We first found the number of unique channels that qualified for a trending video. There were 402 unique channels that posted a trending video during this month. We then found the most common channels that had videos during this time and concluded that 16 different channels had an equal number of videos with 32. These channels include popular TV shows like American Idol and Britain's Got Talent, as well as popular artists like Charlie Puth. In addition, we found out the top most common categories to be featured in this trending videos data set. The entertainment category was the most popular with 1701 videos, followed by Music with 1121.

```
##               Max.Values Min.Values Mean.Values
## views          217750076      56072 4707389.351
## likes            5595203        363  133593.569
## dislikes          335462         17    5949.444
## comment_count    1225326          0   14507.846
```

Numerical Data (views, likes, dislikes, comments): For numerical data analysis, we found the maximum, minimum, and mean of each numerical variable in the data set. The most viewed video had 21 million videos in May, while the most liked video had 5.6 million likes. On the other hand, the least viewed trending video had 56000 views and 363 likes. It is reasonable to conclude that "trending" videos have a wide range of popularity.

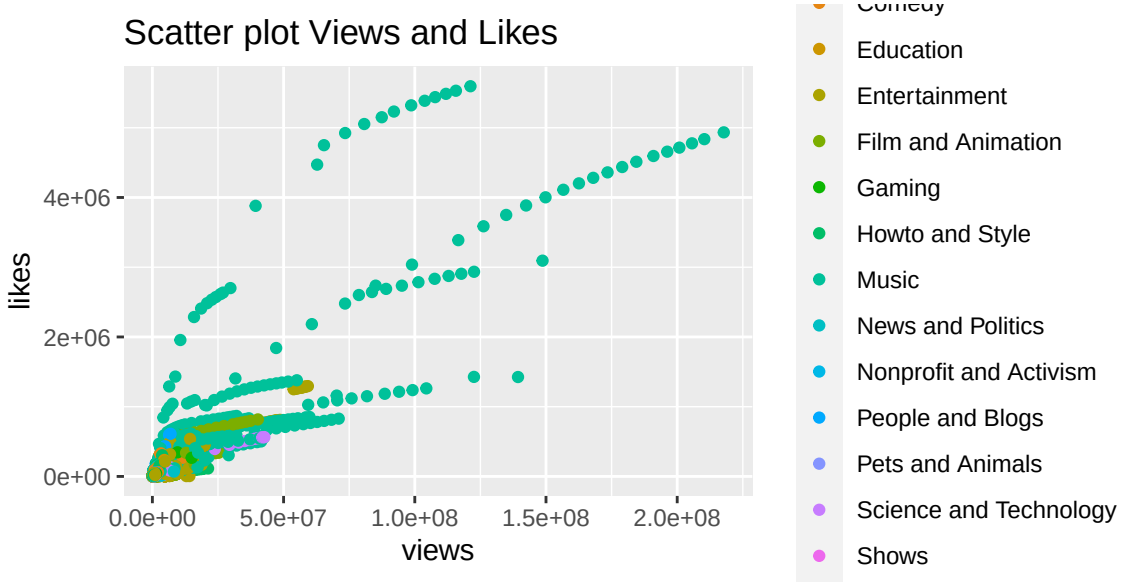
Data Visualization



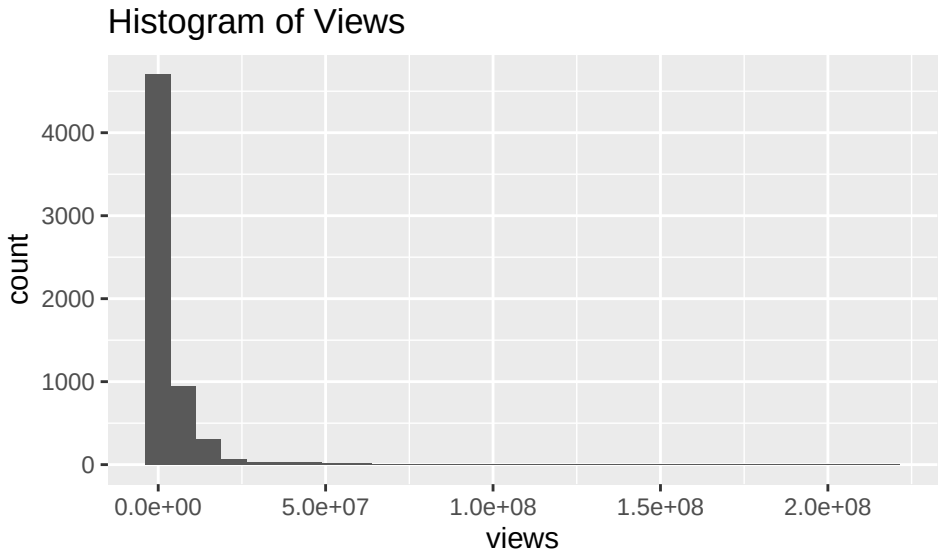


Publish Time x Views: By plotting the publish time against views, we can see which time of the day has the most and least views. It is clear that posting a video during the day tends to bring a great number of views to the video; however, an interesting insight is that posting at 4 AM brought the largest number of views to a video.

Category x Views: The boxplot for each category by views allows us to easily visualize the median for each category along with any outliers. We limited the graph to have a y-limit of views to be 20000000 as having a larger graph makes it more difficult to clearly see the boxplot itself due to the number of outliers. The medians appear to be similar amongst each category, but each 75th percentile differs. Certain categories have many outliers (i.e. entertainment and sports).



Views x Likes: With this scatter plot, the clustered data points appear once again, but it appears between the views and likes relationship. This may indicate that the skewness is present throughout most of the data points. This plot follows the same trend as likes and comment count as music remains to have a few videos that have much greater likes and views and are not within the bunched data points.



Histogram: The histogram of view count confirms that skewed feature of the dataset. Most of the dataset has views below 5*10^7. The right skewness indicates that most of the data points are in the lower cells and fewer cells on the right side.

Data Analysis

(i) Linear Regression (Attempt)

To answer the first question of whether the views of the video have a linear relationship with the user interaction variables, such as likes, dislikes, and comments, correlation analysis and linear regression methodologies

are used.

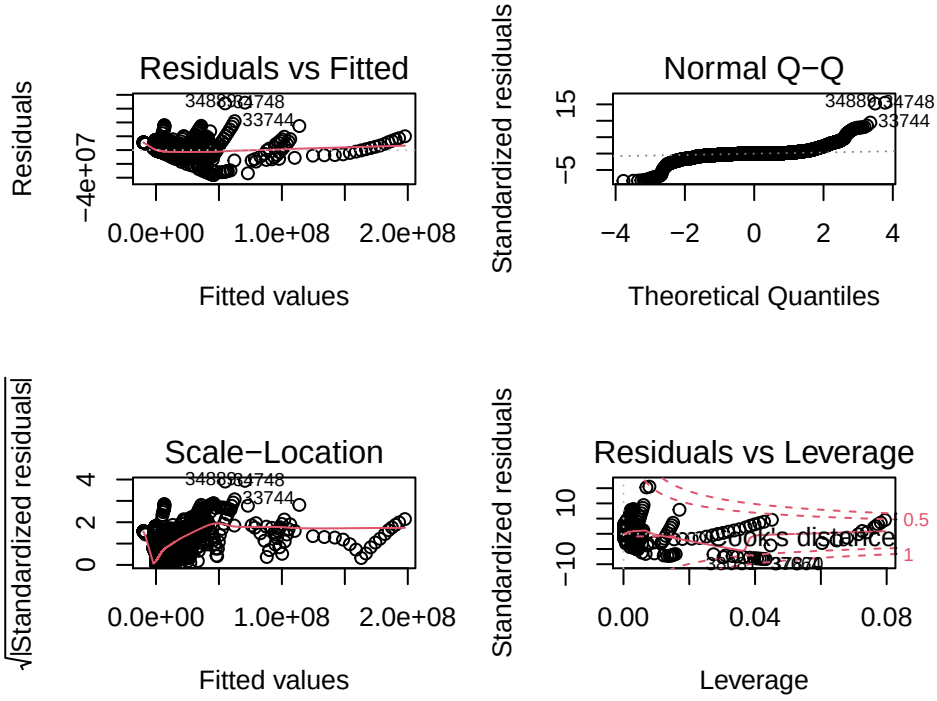
The correlation matrix was created between all the variables observed:

```
##                    views     likes  dislikes comment_count
## views          1.0000000 0.8794465 0.8546943     0.6766620
## likes          0.8794465 1.0000000 0.8183087     0.8978492
## dislikes       0.8546943 0.8183087 1.0000000     0.7221697
## comment_count  0.6766620 0.8978492 0.7221697     1.0000000
```

It can be seen that all the variables are highly correlated with each other. These high correlation coefficients, for instance, 0.898 between comment_count and dislikes and 0.879 between views and likes, might negatively affect the linear regression model because of a collinearity case.

A multiple linear regression model was developed with views as the response variable and likes, dislikes, and comments as the predictor variables.

All the beta coefficients for each predictor variable are statistically significant with very low p-values. The R-squared value indicates that 88.87% of the variation in view counts can be explained by these predictor variables. At a first glance, the model seems to be a good fit with the high R-squared value and significant coefficients. However, a risk of collinearity might exist in this model so some assumptions of the model were analyzed further.



The plot of residuals vs. fitted values shows how the residuals are not scattered randomly around 0, instead, there exists a pattern where the residuals are concentrated where the fitted values are low. This poses a possibility that the homoscedasticity (i.e. equal variance assumption) is violated. Additionally, the heavy-tailed QQ plot pinpoints how this model might not obey the normality assumption. The residual vs leverage plot echoes these trends where there is a lot of outliers present, measured by the Cook's distance. From this analysis, it can be concluded that the data is heavily skewed and a transformation might be needed.

A log transformation was performed to the response variable (i.e. the number of views) while keeping the predictor variables all the same. The transformed model shares a similarity with the original model in which

all the coefficients are statistically significant. However, transformation brings down the R-squared value of the model from 0.8887 to 0.3665, which means less variation in view counts can be explained with this new model.

Again, the residual plot shows how the homoscedasticity (i.e. equal variance) assumption is violated since the residuals form a pattern of getting more negative as the fitted values increase. The QQ plot also shows a case of heavy-tailed, even more extreme compared to the non-transformed model, which means that the normality assumption is also violated. There are also still large numbers of outliers and high-leverage points present. Therefore, these analyses render the transformation to not be helpful in this case.

After both attempts in constructing the multiple linear regression model, both without and with transformations, it is agreed that this methodology is not appropriate for the question of interest. The case of collinearity and heavy skew on the dataset create challenges in accurately predicting a response variable based on the dependent variables.

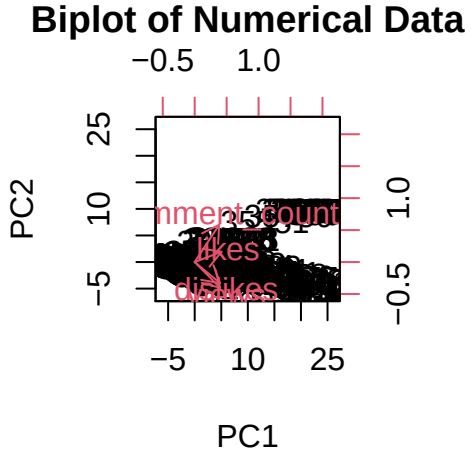
(ii) Principal Component Analysis

With a focus on the numerical data (views, likes, dislikes, comment count), PCA is utilized due to the possible correlation amongst the numerical values. The summary provided from the PCA allows us to summarize and explain the variability of the youtube dataset. As the PCA method is sensitive to scaling, the data is first scaled to have unit variance, transforming the data to have a mean of zero mean and a unit variance standard deviation. It is also shifted to zero centered.

```
##                    PC1        PC2        PC3        PC4
## views         0.4983585 -0.5056834 -0.5262759 -0.4679282
## likes         0.5250324  0.2007584 -0.3584418  0.7453567
## dislikes      0.4952390 -0.4125687  0.7542555  0.1249963
## comment_count 0.4803313  0.7305939  0.1601621 -0.4581076
```

From the correlation matrix, comment count and likes have the highest correlation while likes and views have the second highest. The found loading values of the principal components 1 have likes as the highest value, indicating that there is a large association with this variable. This also includes the other variables (views, dislikes, comment count) since the values are quite similar to one another. The second principal component has a high comment count value and a negative association to views.

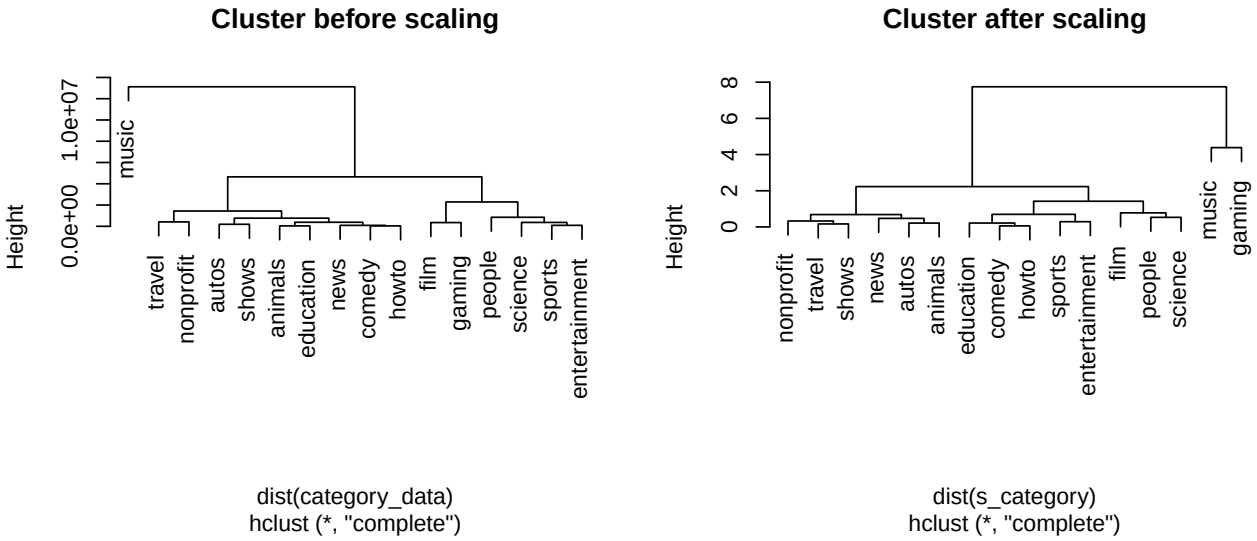
Each principal component has a proportion of variance, but principal component 1 has the highest proportion of 0.857. This indicates that 85.7% of the variability is explained by PC1. PC2 has a proportion of 0.09264. Due to its small value, it is not important to retain this value, along with PC3 and PC4.



The biplot indicates which variable has more effect on which principal component. Variables comment count, dislikes, and views have a stronger influence on PC1 while likes has a stronger influence on PC2. Comment count and dislikes appear to not have a significant correlation due to its 90 degree nature (same with comment count and views). However, since likes and views are close to one another, this signifies a possible correlation. Likes and dislikes may also have a positive relationship to one another. This correlation is confirmed through the correlation matrix previously found.

(iii) Hierarchical Clustering

While Principal Component Analysis centers around the numerical variables of the data set, we wanted to use a method that also focuses on the category ID of the data set. But since our current data set does not have each video category's specific statistics, we created a new data frame that highlights the mean views, likes, dislikes, and comment count of each category. We used a function called calculatemean that would simultaneously calculate the mean views, likes, dislikes, and comment counts, and then returns this data in a list. Then we created a data frame that concatenates all this data for every unique category in the original data set.



The method that we used to analyze this new data frame is called hierarchical clustering. This generates a cluster dendrogram which illustrates the various clustering assignments for each individual category in the data set. Each group in the dendrogram (horizontal line) leads to two leaves that have similar data points as the previous parental grouping. For instance, in the following dendrogram, comedy and howto are very similar in data to news, so they are clustered very closely together within the same parental pairing. Music, on the other hand, seems to have a leaf of its own, which suggests that it differs significantly from the other categories. The height difference of these two subtrees is significant and obvious.

When we cut this tree into its respective clusters, we see the exact distribution of each cluster. By using the cutree() function, we can conclude that the first cluster contains categories film, sports, gaming, people, entertainment, and science. The second cluster contains categories autos, animals, travel, comedy, news, howto, education, nonprofit, and shows. Finally the third cluster only contains the music category. This allows us to visualize how close to each other each category is; for instance, science videos are much closer to sports videos compared to education videos. This is surprising because many would think that science is a subcategory of education and they would go hand in hand. However, in terms of trending video data, this is not the case.

In the next section, we scaled the category data such that it has a standard deviation of 1. This would significantly change the clusterings of each category in the dataset because it will scale the individual piece

of data to have a normal distribution centered around the mean zero. This will change the height of the dendrogram to a smaller range of values. Since music and gaming are much higher in height than the other categories, they get placed in their own clusters. Scaling our data produced a surprising result because we never would have expected gaming to isolate itself in its own cluster because it seems very similar in height to the film category (cluster 1) in the previous dendrogram.

Function

```
searchtag <- function(x){
  df = data.frame("Title" = character(),"Views" = integer(),"Likes" = integer(),"Comments" = integer())
  for(i in 1:length(may_data)){
    if(grepl(x, may_data$tags[i], fixed = TRUE)){
      video <- c(may_data$title[i], may_data$views[i], may_data$likes[i], may_data$comment_count[i])
      df <- rbind(df, video)
    }
  }
  names(df) = c("Title", "Views", "Likes", "Comments")
  return(df)
}
```

We created a function called searchtag that models the search bar on YouTube.com. We brought back the original may_data data set that included the tags category. Although we did not use the tags category in our data analysis project, we found it useful when creating a function for searching relevant terms in the YouTube videos dataset. The function takes in one parameter "x", which is then used to match key words in the tags category. If there is a match found, the function will return the title of the videos that have a matching tag. For instance, if you run "searchtag("marvel")", the function will return "Tom Hiddleston's spot-on Korg impression" and "Where Were Ant-Man and the Wasp? | New Trailer Tomorrow" along with its views, likes, and comments from the may_data data set.

This is useful and appropriate for our study because it allows us to search up specific keywords that make a video trending. For instance, we can answer questions like "Would a video be more trendy with positive or negative connotation tags?" We can test this by typing in different kinds of words to see what tends to be more trendy on YouTube.

Data Interpretation

Which numerical category (i.e. dislikes, comments, likes) affects view counts the most? Is there a correlation between these numerical categories?

From the correlation matrix, the numerical categories have strong positive correlations with each other given the high correlation coefficient. The PCA also visually analyzes the correlation strength of the categories. Because of that, in answering the first part of the question (i.e. "Which numerical category affects view counts the most?") a linear regression model cannot be used, even after a data log-transformation. The case of collinearity and heavy skews on the data make it hard to determine the linear relationship between the numerical variables and view counts due to how closely dependent each variable is. A more appropriate method might be conducting a simple linear regression for each variable (e.g. dislikes and views, comments and views, likes and views) and comparing the three models with each other. Considering how highly correlated the variables are, it is also possible that all the numerical categories affect view counts equally; no variable is better than the other.

What time of the day is the best time to post a video?

Incorporating data visualization, it is found that 4 am publish time garners the most view counts to a video. However, a general trend is that a video posted before 11 am is viewed more than anytime after.

Which video category is the most popular?

According to the boxplot, the mean view counts of music videos is the highest among all. On the other hand, nonprofit and activism videos are the least popular.

How can the video categories be grouped based on user interaction?

A hierarchical clustering was applied to the dataset with a purpose to group the video categories. The mean of each numerical variable (i.e. views, comments, likes, and dislikes) was taken and used as a basis of clustering. Then a dendrogram with complete linkage and Euclidean distance was created to observe the closeness of each video category.

Conclusion

This study puts into perspective how several factors, such as user interaction, video category, and publish time, are correlated with the number of views of the top trending YouTube videos in the United States in May 2018. In conclusion, there are several factors that contribute to the popularity of a video. This study highlights some of the key ones, such as publish time, user interaction, and video categorization. However, there exists some limitations to this study such as the narrow timeframe (i.e. May 2018) and the number of videos analyzed. A more thorough analysis might be necessary to give a full picture of what makes a video trending on YouTube.

References

[1] "Top 100: The Most Visited Websites in the US [2021 Top Websites Edition]." Semrush Blog. https://www.semrush.com/blog/most-visited-websites/

[2] "Trending on YouTube." (n.d.). YouTube Help. https://support.google.com/youtube/answer/7239739?hl=en

[3] Link to dataset: "Trending YouTube Video Statistics." (2018). Kaggle. https://www.kaggle.com/datasnaek/youtube-new?select=CAvideos.csv

[4] Lathiya, K. (Nov 25, 2021). "grepl in R: How to Use R grepl () Function." R-lang. https://r-lang.com/grepl-in-r/

```
knitr::opts_chunk$set(
  error = F,
  message = F,
  warning = F,
  options(knitr.kable.NA = ''),
  tidy = F,
  fig.align = "center"
)
library(knitr)
# dataset
data <- read.csv("USvideos.csv")
# data cleaning
data_year<- subset(data, substr(trending_date, 1, 2) =="18")
may_data <- subset(data, substr(trending_date, 7, 8) =="05")
```

```r
maydata <- subset(may_data, select = -c(title, tags, thumbnail_link,description, comments_disabled, rati

head_data <- maydata
head_data <- na.omit(head_data)

# replacing category number with proper title
head_data$category_id <- sapply(as.character(head_data$category_id), switch,
                            "1" = "Film and Animation",
                            "2" = "Autos and Vehicles",
                            "10" = "Music",
                            "15" = "Pets and Animals",
                            "17" = "Sports",
                            "18" = "Short Movies",
                            "19" = "Travel and Events",
                            "20" = "Gaming",
                            "22" = "People and Blogs",
                            "23" = "Comedy",
                            "24" = "Entertainment",
                            "25" = "News and Politics",
                            "26" = "Howto and Style",
                            "27" = "Education",
                            "28" = "Science and Technology",
                            "29" = "Nonprofit and Activism",
                            "43" = "Shows")

# changing publishing time to publishing hour
head_data$publish_time <- substr(head_data$publish_time, 12, 13)
length(unique(head_data$channel_title))
head(sort(table(head_data$channel_title), decreasing = T), 10)
head(sort(table(head_data$category_id), decreasing = T), 10)

max_views <- sapply(head_data["views"], max, na.rm = T)
max_likes <- sapply(head_data["likes"], max, na.rm = T)
max_dislikes <- sapply(head_data["dislikes"], max, na.rm = T)
max_commentcount <- sapply(head_data["comment_count"], max, na.rm = T)
max_values <- c(max_views, max_likes, max_dislikes, max_commentcount)

min_views <- sapply(head_data["views"], min, na.rm = T)
min_likes <- sapply(head_data["likes"], min, na.rm = T)
min_dislikes <- sapply(head_data["dislikes"], min, na.rm = T)
min_commentcount <- sapply(head_data["comment_count"], min, na.rm = T)
min_values <- c(min_views, min_likes, min_dislikes, min_commentcount)
mean_views <- sapply(head_data["views"], mean, na.rm = T)
mean_likes <- sapply(head_data["likes"], mean, na.rm = T)
mean_dislikes <- sapply(head_data["dislikes"], mean, na.rm = T)
mean_commentcount <- sapply(head_data["comment_count"], mean, na.rm = T)
mean_values <- c(mean_views, mean_likes, mean_dislikes, mean_commentcount)
temp <- data.frame("Max Values" = max_values, "Min Values" = min_values, "Mean Values" = mean_values)
temp
require(ggplot2)
ggplot(data = head_data) +  geom_point(mapping = aes(x = publish_time, y = views), color = "green") + gg
ggplot(data = head_data, aes(x = category_id, y = views, fill = category_id)) +
  geom_boxplot() + ylim(min = 0, max = 20000000)+ theme(axis.text.x=element_text(angle=90,hjust=1,vjust=
```

```r
ggplot() + geom_point(data = head_data, aes(x = views, y = likes,color = category_id)) + ggtitle("Scatt
ggplot()+ geom_histogram(data = head_data, aes(x = views)) + ggtitle("Histogram of Views")
cormat <- cor(head_data[,6:9])
cormat
model <- lm(views~likes+dislikes+comment_count, head_data)
par(mfrow=c(2,2))
plot(model)
logmodel <- lm(log(views)~likes+dislikes+comment_count, head_data)
par(mfrow=c(2,2))
plot(logmodel)
pr.out <- prcomp(head_data[,6:9], center = TRUE, scale.=TRUE)
loadings <- pr.out$rotation
loadings
summary(pr.out)
biplot(pr.out,scale=0, main = "Biplot of Numerical Data")
film <- head_data[which(head_data$category_id == "Film and Animation"),]
autos <- head_data[which(head_data$category_id == "Autos and Vehicles"),]
music <- head_data[which(head_data$category_id == "Music"),]
animals <- head_data[which(head_data$category_id == "Pets and Animals"),]
sports <- head_data[which(head_data$category_id == "Sports"),]
shorts <- head_data[which(head_data$category_id == "Short Movies"),]
travel<- head_data[which(head_data$category_id == "Travel and Events"),]
gaming <- head_data[which(head_data$category_id == "Gaming"),]
people <- head_data[which(head_data$category_id == "People and Blogs"),]
comedy <- head_data[which(head_data$category_id == "Comedy"),]
entertainment <- head_data[which(head_data$category_id == "Entertainment"),]
news <- head_data[which(head_data$category_id == "News and Politics"),]
howto <- head_data[which(head_data$category_id == "Howto and Style"),]
education <- head_data[which(head_data$category_id == "Education"),]
science <- head_data[which(head_data$category_id == "Science and Technology"),]
nonprofit <- head_data[which(head_data$category_id == "Nonprofit and Activism"),]
shows <- head_data[which(head_data$category_id == "Shows"),]

calculatemean <- function(x){
  views <- mean(x[,6])
  likes <- mean(x[,7])
  dislikes <- mean(x[,8])
  comment_count <- mean(x[,9])
  return(c(views, likes, dislikes, comment_count))
}

filmmean <- calculatemean(film)
autosmean <- calculatemean(autos)
musicmean <- calculatemean(music)
animalsmean <- calculatemean(animals)
sportsmean <- calculatemean(sports)
shortsmean <- calculatemean(shorts)
shortsmean
travelmean <- calculatemean(travel)
gamingmean <- calculatemean(gaming)
peoplemean <- calculatemean(people)
comedymean <- calculatemean(comedy)
entertainmentmean <- calculatemean(entertainment)
```

```
newsmean <- calculatemean(news)
howtomean <- calculatemean(howto)
educationmean <- calculatemean(education)
sciencemean <- calculatemean(science)
nonprofitmean <- calculatemean(nonprofit)
showsmean <- calculatemean(shows)

category_data <- data.frame("views" = c(filmmean[1],autosmean[1], musicmean[1], animalsmean[1], sportsme

rownames(category_data) <- c("film", "autos", "music", "animals", "sports", "shorts", "travel", "gaming

category_data <- na.omit(category_data)


hc.complete <- hclust(dist(category_data), method="complete")
s_category <- scale(category_data)
s_hc.complete <- hclust(dist(s_category), method="complete")
par(mfrow=c(2,2))
plot(hc.complete, main = "Cluster before scaling")
plot(s_hc.complete, main = "Cluster after scaling")
cut <- cutree(hc.complete,3)
table(cut)
cut <- cutree(s_hc.complete,3)
table(cut)
searchtag <- function(x){
  df = data.frame("Title" = character(),"Views" = integer(),"Likes" = integer(),"Comments" = integer())
  for(i in 1:length(may_data)){
    if(grepl(x, may_data$tags[i], fixed = TRUE)){
      video <- c(may_data$title[i], may_data$views[i], may_data$likes[i], may_data$comment_count[i])
      df <- rbind(df, video)
    }
  }
  names(df) = c("Title", "Views", "Likes", "Comments")
  return(df)
}
#Appendix
```