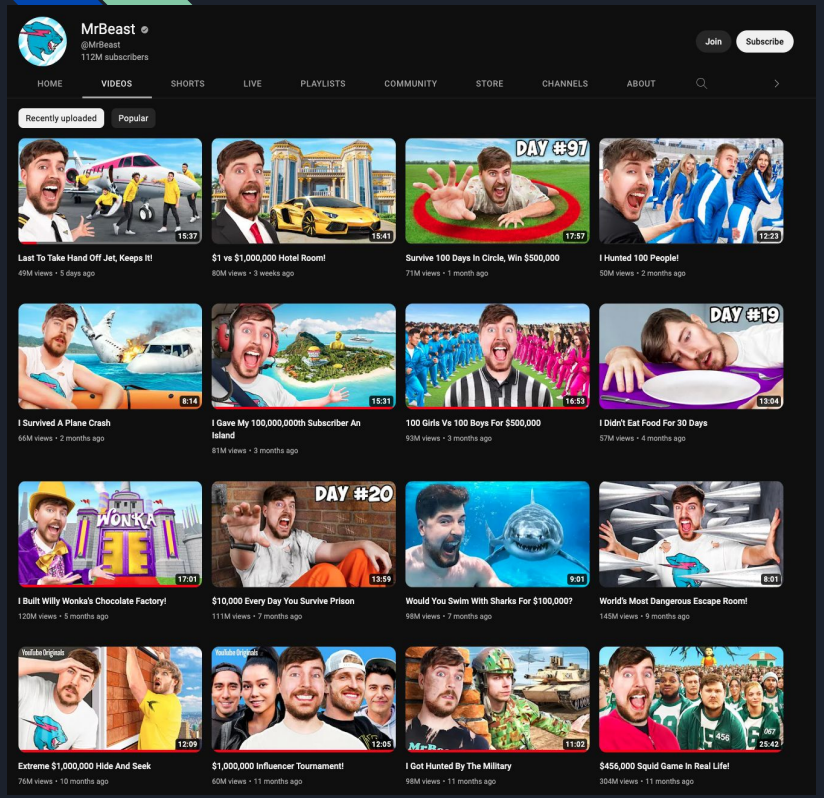
How Mr. Beast Grew on Youtube

Jessica Nguyen CDS 301

Who is Mr. Beast?

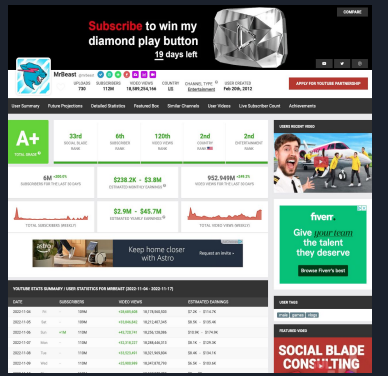


Being on the YouTube platforms for almost 10 years now, Mr Beast is considered one of the fastest-growing and most successful Youtube channels, hitting 112 million subscribers within his main and averaging 10 million subscribers on each of his four side channels. His content ranges from challenges, gaming, and philanthropy videos. His audiences' ages range between elementary to high school students.

I wanted to ask myself, *"how could I create a youtube channel as successful as Mr. Beast?"*

Surprisingly, there is already a website that could answer that question for me called *socialblade.com.* However, there are also limitations to this website.

- X No qualitative data analysis (#4 Tufte's Rule: Integration)
- X No story about interaction between creator vs. viewer being told with exception of "subscriber trend" graphs (#3 Tufte's Rule: Multivariate)
- X No way to find dependent variables influencing each other (#1 Tufte's Rule: Comparisons #2 Tufte's Rule: Casualty)
- X Too many ads



But there are also strengths.

- ✅ Trend line graphs for total videos and total subscribers (#6 Tufte's Rule: Context)
- ✅ Includes information about subscribers, video uploads, views, monthly earnings (#3 Tufte's Rule: Multivariate)

My Goal will be to **better optimize Mr. Beast's youtube channel API than socialblade.com to provide more insightful statistical analysis.**

Literature Review

Through my findings within Mr Beast's YouTube channel, I found that the correlation between Mr. Beast's most commonly used words within his YouTube titles is not a strong causative variable to explain the growth of his YouTube channel. This was supported by the lack of his most popular keywords in his YouTube titles within the graphs of both his best viewed YouTube videos and his best liked YouTube videos, **with only 2 videos having those keywords out of 20**. However, 14/20 of his best performing videos (best liked and best viewed combined) included numbers. A future graph I could create is a word cloud of the most used numbers within his YouTube channel and compare it with the numbers seen in his top performing videos. Within the word cloud and bar plots, I've utilized the all 5 of Tufte's Rules of Visualization (Comparisons, Causality, Multivariate, Integration, Documentation, Context).

The purpose of the correlation matrix used within this study is to re-emphasize Tufte's rule of Visualization #1: Comparisons through the multiplicity of dependent variables (YouTube comments, likes, views) and how they correlate together to eventually create the story of Mr. Beast's success on YouTube. I came to the conclusion that while Mr. Beast has positive correlations within his audience demographic over the years, his worst metric for success are YouTube comments. With the Youtube comments and timeline publish dates having the least positive correlation, this is where Mr. Beast should focus on improving on to further grow his channel. Future data analytic methods would be to create a word cloud of Mr. Beast's top performing video comments vs. a word cloud of Mr. Beast's worst performing video comments. This would allow for a better comparison on what Mr. Beast's audience finds engaging about the best performing video and what criticism they had for the worst performing video.

Dataset and Packages

library(dplyr)
library(readr)
library(devtools)
library(AnomalyDetection)
library(ggplot2)
library(tm)
library(stringi)
library(lattice)
library(udpipe)
library(wordcloud)
library(SnowballC)
library(highcharter)
library(tidyverse)

For the word cloud graph
For the scatter, line graphs
For reading external files

```
key <- ("AIzaSyC1ETOdFrV5QZHhBlTGhzflQpxXa8MMsuc") #api key
MrBeastChannelID<-"UCX6OQ3DkcsbYNE6H8uQQuVA" #channel id
base<- "https://www.googleapis.com/youtube/v3/" #baseurl

# Construct the API call: https://www.yuichiotsuka.com/youtube-data-extract-r/
api_params <-
  paste(paste0("key=", key),
        paste0("id=", MrBeastChannelID),
        "part=snippet,contentDetails,statistics",
        sep = "&")
api_call <- paste0(base, "channels", "?", api_params)
api_result <- GET(api_call)
json_result <- httr::content(api_result, "text", encoding="UTF-8")

#format channel information from API into dataframe
channel.json <- fromJSON(json_result, flatten = T)
channel.df <- as.data.frame(channel.json)

#take 'upload' playlist ID from the channel dataframe in order to get video information
playlist_id <- channel.df$items.contentDetails.relatedPlaylists.uploads

#temporary variables
nextPageToken <- ""
upload.df <- NULL
pageInfo <- NULL

#go through pages to get all the video information
while (!is.null(nextPageToken)) {
  # Construct the API call
  api_params <-
    paste(paste0("key=", key),
          paste0("playlistId=", playlist_id),
          "part=snippet,contentDetails",
          "maxResults=50",
          sep = "&")

  # Add the page token for page 2 onwards
  if (nextPageToken != "") {
    api_params <- paste0(api_params,
                         "&pageToken=",nextPageToken)
  }

  api_call <- paste0(base, "playlistItems", "?", api_params)
  api_result <- GET(api_call)
  json_result <- httr::content(api_result, "text", encoding="UTF-8")
  upload.json <- fromJSON(json_result, flatten = T)

  nextPageToken <- upload.json$nextPageToken
  pageInfo <- upload.json$pageInfo

  curr.df <- as.data.frame(upload.json$items)
  if (is.null(upload.df)) {2
    upload.df <- curr.df
  } else {
    upload.df <- bind_rows(upload.df, curr.df)
  }
}

#upload.df is the dataframe that contains all the video information (titles, description, when it was posted)
```

```
#Now, I need to get the video statistics such as the likes, view count, and comment count.

#create empty vectors to be the column names
items.id <- c()
items.statistics.viewCount <- c()
items.statistics.likeCount <- c()
items.statistics.commentCount <- c()

for (x in 1:730){ #loop through all 730 Mr Beast videos
  vid_api_params <-
    paste(paste0("key=", key),
          paste0("id=", upload.df$contentDetails.videoId[x]),
          "part=statistics",
          sep = "&")
  vid_api_call <- paste0(base, "videos", "?", vid_api_params)
  vid_api_result <- GET(vid_api_call)
  vid_json_result <- httr::content(vid_api_result, "text", encoding="UTF-8")
    vid.json <- fromJSON(vid_json_result, flatten = T)
    vid.df <- as.data.frame(vid.json)

    #If there are any variables within the row that are blank, say they are NA

    if("items.statistics.likeCount" %in% colnames(vid.df) == FALSE){
      items.statistics.likeCount[x] <- NA
    }
    else if("items.id" %in% colnames(vid.df) == FALSE)
      {
    items.id[x] <- NA
    }
    else if("items.statistics.viewCount" %in% colnames(vid.df) == FALSE){
    items.statistics.viewCount[x] <- NA
    }
    else if ('items.statistics.commentCount'%in% colnames(vid.df) == FALSE){
      items.statistics.commentCount[x] <- NA
    }
    else{
      items.id[x] <- vid.df$items.id[1] #if all the variables within the row is present, then add that variable into the vector
      items.statistics.commentCount[x] <- vid.df$items.statistics.commentCount[1]
      items.statistics.viewCount[x] <- vid.df$items.statistics.viewCount[1]
      items.statistics.likeCount[x] <- vid.df$items.statistics.likeCount[1]
    }

    }

vid.df <- cbind(items.id, items.statistics.viewCount,items.statistics.likeCount, items.statistics.commentCount) #bind the vector to get a matrix
vid.df[is.na(vid.df)] <- 0 #make all NA into 0

upload.df <- cbind(upload.df,vid.df) #combine the video information (titles, description, when it was posted) and video statistics (likes, view count, and comment count)

write.csv(upload.df, file = "upload.csv")
```

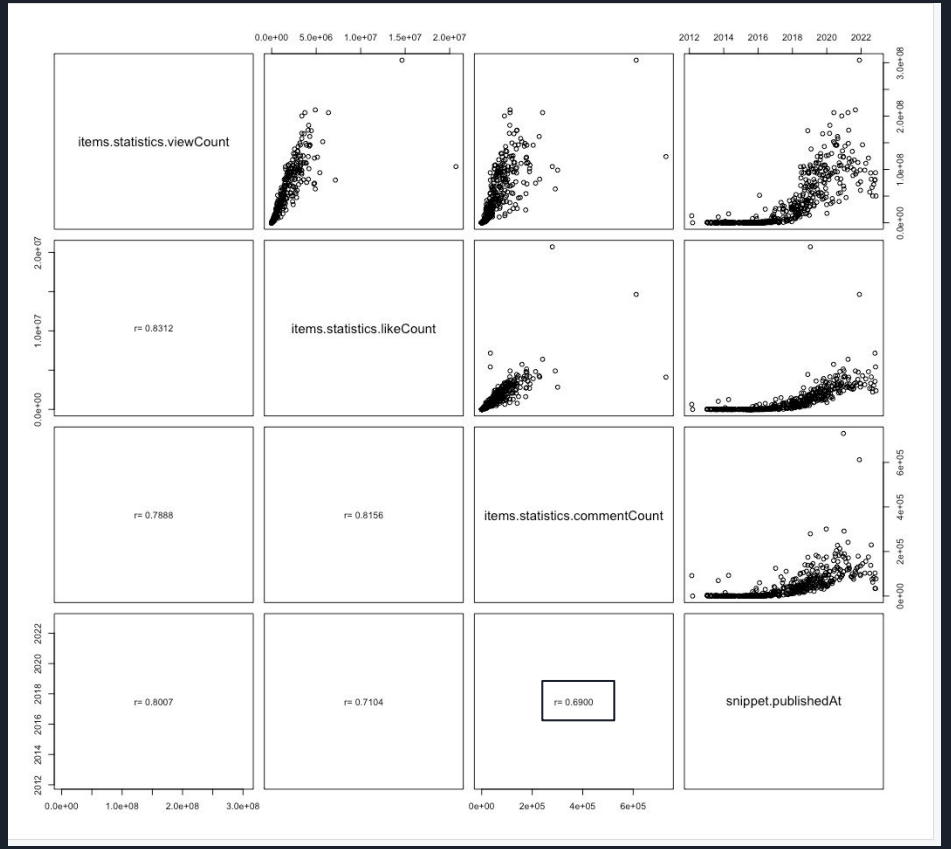
The Completed, Created Dataset

8 columns and 730 rows

	items.id	items.statistics.viewCount	items.statistics.likeCount	items.statistics.commentCount	snippet.publishedAt	snippet.description	snippet.title	snippet.playlistId
1	kX3nB4PpJko	50139135	2370866	77669	2022-11-12T21:00:00Z	Close this and watch to the end to see who wins! New Merch - https://shopmrbeast.com/ At the end I said we would offset all carbon emissions but what we actually did was offset it by 10x!! SUBSCRIBE OR I TAKE YOUR DOG  --- follow all of these or i will kick you • Facebook - https://www.facebook.com/MrBeast6000/ • Twitter - https://twitter.com/MrBeast • Instagram - https://www.instagram.com/mrbeast • Im Hiring! - https://www.mrbeastjobs.com/ ---	Last To Take Hand Off Jet, Keeps It!	UUX6OQ3DkcsbYNE6H8uQQuVA
2	YlvcFJOE-OE	93869725	5409195	33870	2022-11-02T21:00:01Z	New Merch - https://shopmrbeast.com/ SUBSCRIBE OR I TAKE YOUR DOG  --- follow all of these or i will kick you • Facebook - https://www.facebook.com/MrBeast6000/ • Twitter - https://twitter.com/MrBeast • Instagram - https://www.instagram.com/mrbeast • Im Hiring! - https://www.mrbeastjobs.com/ ---	Giving iPhones Instead Of Candy on Halloween	UUX6OQ3DkcsbYNE6H8uQQuVA
3	iogcY_4xGjo	80845611	3566546	103190	2022-10-22T21:00:00Z	The hotel at the end is worth the wait! Download the Experian App: https://smart.link/n3op1gefxlzjn or Visit Experian.com/Boost. *Results will vary. Not all payments are boost-eligible or considered by lenders. Credit Card offers are not availab Check out Conrad Maldives Rangali Island! Facebook @conradmaldivesrangaliisland Youtube https://www.youtube.com/c/ConradMaldivesRangaliIsland New Merch - https://shopmrbeast.com/ SUBSCRIBE OR I TAKE YOUR DOG  --- follow all of these or i will kick you • Facebook - https://www.facebook.com/MrBeast6000/ • Twitter - https://twitter.com/MrBeast • Instagram - https://www.instagram.com/mrbeast • Im Hiring! - https://www.mrbeastjobs.com/ ---	$1 vs $1,000,000 Hotel Room!	UUX6OQ3DkcsbYNE6H8uQQuVA
4	S_CUEOBZ0P4	79919105	7158110	34282	2022-10-21T20:00:06Z	Winning totally Rocks!		

Correlation Matrix

```r
#plot1 correlation coefficient chart
panel.cor <- function(x, y, digits = 4, cex.cor, ...)
{
  usr <- par("usr"); on.exit(par(usr))
  par(usr = c(0, 1, 0, 1))
  # correlation coefficient
  r <- cor(x, y)
  txt <- format(c(r, 0.123456789), digits = digits)[1]
  txt <- paste("r= ", txt, sep = "")
  text(0.5, 0.5, txt)
}
pairs(upload[,3:6], lower.panel = panel.cor)
```



Word Cloud

```
upload <- read_csv("/Users/jessicanguyen/mrbeast/MrBeastYTAnalytics/upload.csv")

#creating word cloud for Mr Beast's Video Titles
titles <- read_file('/Users/jessicanguyen/mrbeast/MrBeastYTAnalytics/mrbeast_titles.txt')

tdocs<-Corpus(VectorSource(titles))

## pre-process the texts
tdocs <- tm_map(tdocs, function(x) stri_replace_all_regex(x, "<.+?>", " "))
tdocs <- tm_map(tdocs, function(x) stri_replace_all_fixed(x, "\n", " "))

# consider all raw text
tdocs <- tm_map(tdocs, PlainTextDocument)
# remove punctuation
tdocs <- tm_map(tdocs, removePunctuation)
# remove numbers and symbols
tdocs <- tm_map(tdocs, removeNumbers)
# set capital letters to lower letters
tdocs <- tm_map(tdocs, content_transformer(tolower))
# remove common words ("the", "and")
tdocs <- tm_map(tdocs, removeWords, stopwords("english"))

tdtm<-DocumentTermMatrix(tdocs)
ttdm<-TermDocumentMatrix(tdocs)

tmywords<-tdtm$dimnames$Terms

tdtm$dimnames$Docs<-as.character(c(1:length(text)))
rowTotals<- apply(tdtm , 1, sum)
tdtm.new <- tdtm[rowTotals> 0,]

tdictionary<-tdtm$dimnames$Terms

tfreq <- colSums(as.matrix(tdtm))
tord <- order(tfreq,decreasing=TRUE)
t_dict <- as.data.frame(tfreq[tord])

tm <- as.matrix(tdtm)
tv <- sort(colSums(tm), decreasing=TRUE)
t_myNames <- names(tv)
t_dtmnew <- data.frame(word=t_myNames, freq=tv)
wordcloud(t_dtmnew$word, family = "serif", font = 6, colors=colorRampPalette(brewer.pal(9,"Greens"))(32)[seq(8,32,6)], t_dtmnew$freq, min.freq = 10)
```

Commonly Used Words in Mr. Beast's Titles

Comparison of Graphs: Best Viewed Videos vs. Best Liked Videos

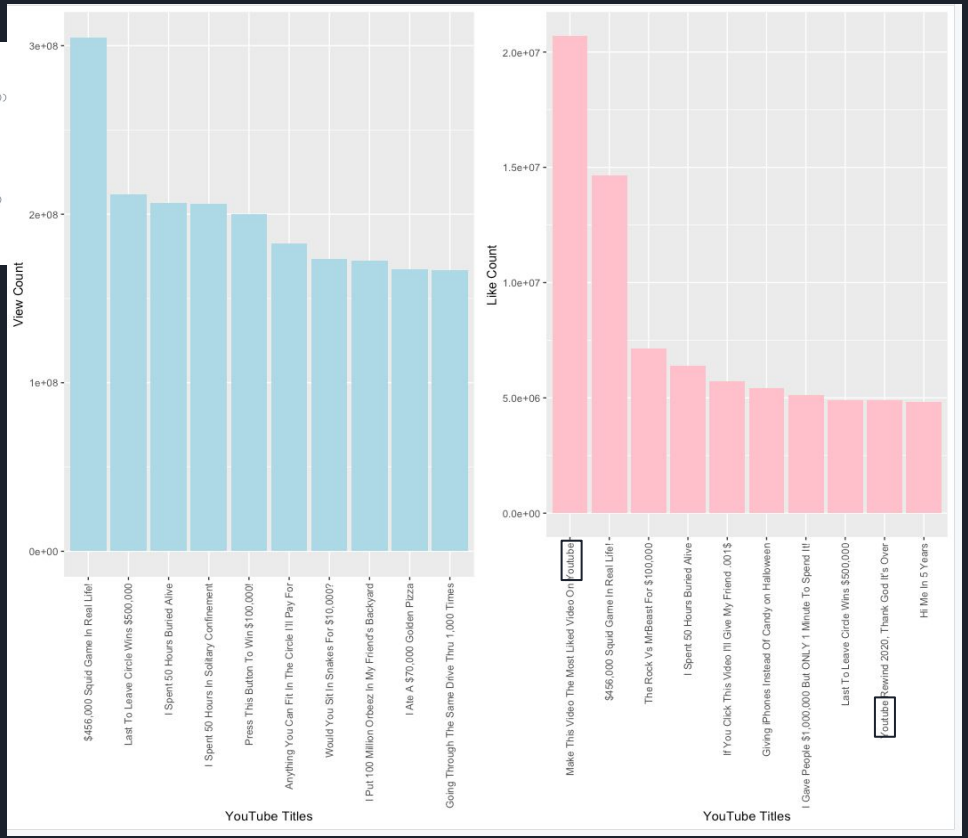
```r
#best videos via views
upload <- upload[ order(upload$items.statistics.viewCount, decreasing = TRUE ),] #making sure it is in highest to lowest order
best_viddf_views<-data.frame(upload$items.snippet.title[1:10],upload$items.statistics.viewCount[1:10]) ##capturing only top ten videos

plot2 <- ggplot(data = best_viddf_views, mapping = aes(x = reorder(as.factor(upload.snippet.title.1.10.), -upload.items.statistics.viewCount.1.10.), y = upload.items.statistics.viewCount.1.10.))
  geom_bar(stat = 'identity', fill = 'lightblue') +
  labs(x = "YouTube Titles", y = 'View Count') +
theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))

#best videos via likes
upload <- upload[ order(upload$items.statistics.likeCount, decreasing = TRUE ),] #making sure it is in highest to lowest order
best_viddf_likes <- data.frame(upload$snippet.title[1:10], upload$items.statistics.likeCount[1:10]) ##capturing only top ten videos

plot3 <- ggplot(data = best_viddf_likes, mapping = aes(x = reorder(as.factor(upload.snippet.title.1.10.),-upload.items.statistics.likeCount.1.10.), y = upload.items.statistics.likeCount.1.10.))
  geom_bar(stat = 'identity', fill = 'pink') +
  labs(x = "YouTube Titles", y = 'Like Count') +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))

plot_grid(plot2, plot3, ncol = 2, nrow = 1)
```



Citations

Code for extracting YouTube API into R-studio: https://www.yuichiotsuka.com/youtube-data-extract-r/

Inspiration for analyzing YouTube analytics:
https://www.youtube.com/watch?v=D56_Cx36oGY&t=500s