YouTube Video Marketing: Top 4 Ways to Drive Traffic to YouTube Video

This recommendation memo is based on below analysis methodologies using R upon the YouTube trending video dataset. First, we utilize linear regression models to learn the existence and strength of the correlations between number of views (dependent variable) and number of likes, number of dislikes, number of comments (independent variables). We also utilized linear regression analysis to verify the video improving tips from so called video promotion expert.

Then, in order to explore the potential information offered by the non-numeric variables in the video dataset, we applied text-mining methods, explored the general sentiments of YouTube trending list videos, extracted the emotions described in the title, tag and descriptions, and also leveraged word clouds to extract the keywords of trending video titles.

We have already used the linear regression models in visualizing the correlations between the amount of views and the number of likes, dislikes, and comments, respectively. Now we would try to launch the logistic regression analysis to see the multivariable influence on the number of views. Logistic regression model would allow us to combine the different independent variables together for understanding the overall effect. Moreover, we could avoid the negative effects from the extraneous in the previous tests. However, we found that the relationship between the views and the independent variables are all not binomial. In this case, the logistic regression may not be suitable for our analysis, and we should find alternative ways to understand the overall effects, which would lead to further explorations. Instead of the logistic regression model, we also tried the analytical method of decision tree, to figure out a step by step process for creating a popular trending video. Unfortunately, since there is correlation between every independent variables, we cannot conclude a step by step guidance for marketing strategy.

After understanding the key elements affecting the popularity of trending videos in micro view, we decide to analysis the data in macro view. Whether our provided recommendation is appropriate to all individuals and industries or not? We would like to use the segmentation analysis to have a overall view of the spreading of the current data set. The cluster analysis would help us to visualize the group distribution of the exist data, so that we might understand the potential meaning of the data set more clearly.

*Recommendation 1: Choose the right keywords into video metadata**

channel title

Through text-mining analysis, YouTube videos in the trending list have popular keywords in their channel names, such as **News, CBS, Shows, and Sports**. These keywords show the most popular topics that audience is watching on YouTube. Selecting the relevant words and including these keywords in channel names could help our videos to get on the trending list of YouTube and drive more traffic.

video titles

Keywords in the trending video titles are found as **Official, Live, Trailer, New, Music**. Among the top 5 keywords, the word "official" is much more frequently used than the other four here. This finding echoes the first two keywords in the channel title: News and CBS, indicating that the users in YouTube are very interested in the news type of videos. We could consider this video title (or theme/ format) in future videos. In addition, since YouTube is the second largest search engine in the world and title name has the biggest weight on video placement, all popular keywords should be added to our ideal keywords list for future use.

tags

Using the similar text-mining method to analyze the trending videos' tags, keywords of highest frequency are found as **Bowl, Super, Music, New, iPhone, Life**. Consider to properly tag these popular keywords to the videos could bring more views.

One side note here is about the amount of tags. During our research, some "experts" claimed that one should add as many tags as you can think of to get more views (Neil Petals, Mindvalley Insights). However, through a linear regression analysis between the video views and tag length, there is no correlation in between (*p value* = 0.2214). Thus, adding the most popular and most relevant tags would be good enough.

descriptions

Video descriptions help YouTube audience understand the context of a video, and also help YouTube and Google to search the video. Here we found the top 5 keywords in the video descriptions as **New, Music, Now, World, News**. We could also consider including the top keywords into the descriptions of our videos to attract more traffic and search results.

Another side note similar to the note of video tags is that super long descriptions don't help. As Brian Dean points out in "5 Advanced YouTube SEO Tactics to Drive More Traffic to Your Videos & Website" that long video descriptions can contain more keywords so that the video can get more views, we also did a linear regression to verify the correlation between the views of trending videos and their description length and found nothing (*p value* = 0.0534).

An overall recommendation would be that producing videos related to the **current hot News or events** would generate more traffic. Since the sample dataset that we analyzed contains the YouTube trending video data collected during 2017 November to 2018 January, we found the keywords like super, bowl, world, show, and Christmas. Thus, we would positively assume that the hot news relevant videos are more likely to attract more views.

*(*Video metadata is about all the textual information that accompanies a video and helps search engines better understand the content as well as helps users find a video on the internet.*
To check the top 30 keywords of video titles, channel titles, tags, and descriptions, please refer to Appendix A.)

Recommendation 2: Choose the right timing to release videos

Release day

Through analyzing the summary of views for the trending videos in different weekdays, we found the top 3 days that generated most views: **Thursday, Sunday, Friday**. (The least amount of views was generated on Tuesday.)

In addition, since we also would have videos to release at a given time slot, we analyzed the counts of views at specific hours upon different weekdays. As a result, we could utilize below chart to select a right weekday to publish the videos at a specific hour. Selecting a right weekday would increase the probability for our videos to attract more views.

9:00 AM - Monday
11:00 AM - Thursday
13:00 PM - Wednesday
15:00 PM - Thursday/ Friday
17:00 PM - Friday
19:00 PM - Saturday
21:00 PM - Sunday
23:00 PM - Friday/ Saturday/ Sunday

Release time

By applying similar analysis method, we would suggest publishing our videos at below specific time slots: morning – **11 AM/ 9 AM**; afternoon – **13 PM**; evening – **19 PM-21 PM**. Furthermore, if we need to publish videos on a specific weekday. We could leverage below chart to pick up a perfect timing.

Monday - 9:00 AM/ 7:00 AM

Tuesday - 21:00 PM/ 20:00 PM
Wednesday - 13:00 PM/ 9:00 AM
Thursday - 11:00 AM
Friday - 7:00 AM/ 9:00 AM
Saturday - 19:00 PM/ 5:00 AM
Sunday - 20:00 PM/ 8:00 AM

If we combine the right release weekday and time, we found that the best practice for publishing a video is: **Thursday morning at 11AM**. (This was exactly the timing when I released the marketing content through WeChat platform in my previous work experience at Lubrizol. The media vendor just orally recommended this timing for publishing based on their practical experience. And it is verified now through data analysis.)

Recommendation 3: Understand the power of your enterprise

By segmenting the amounts of likes in our USvideos data set into four main clustering groups, we received the resulting groups with size members of 120, 1585, 23, 5, respectively.
80 percent of the sample group received likes amounts less than 500,000, whereas the remaining 20 percent of the sample group received much higher amounts of likes, which are between 500,000 to 2,000,000.

Similarly, by segmenting the amounts of views in our data set into four main clustering groups, we received the resulting groups with size members of 190, 1503, 37, 3, respectively.
80 percent of the sample group is viewed by less than 3,000,000 audience, whereas the remaining 20 percent of the sample group has been viewed by much higher amounts of audience, which are between 3,000,000 to 150,000,000.

Moreover, based on the amounts of views and likes and their interconnection, we build up four main clustering groups by segmentation analysis as well. The resulting groups have sizes of 190, 1503, 37, 3, respectively. 80 percent of the sample group is viewed by less than 3,000,000 audience and received likes less than 500,000, whereas the remaining 20 percent of the sample group has been viewed by much more audiences and received much higher amounts of likes.

The result of this clustering analysis reflects the Pareto Principle, which is well known by 80/20 Rule. The 80/20 Rule means that in any set of things (workers, customers, etc.) a few (20 percent) are vital and many (80 percent) are considered trivial. In Pareto's case, he found that roughly 20 percent of the people in his country dominated with 80 percent of the wealth. In our

analysis, 80 percent of the receive trivial popularity compared to popularity received by the remaining 20 percent of trending videos.

This huge gap is because of the initial power of influence of the producing company or individuals. For example, the production of the big tycoons, such as an advertisement video released by Chanel perfume, would receive high attention from the audience and customers immediately without any intentional advertising. Nonetheless, a video from non-famous individuals or companies, even though its content in innovative and attractive, it would not have the power to receive such concentrated attention as the big tycoons.

In this case, enterprises with large influential power could manipulate their platform for more attractions while releasing videos, whereas the small companies and individuals, could seek to get connection between the well-known enterprises and famous individuals, such as idols and influential politicians, to promote their original videos.

Recommendation 4: Understand your target market and potential market layer

Based on our current date set, we could segment the target market of Youtube into three main groups, young teenagers and students, middle aged workers, and elders. We could also segment the potential market into women market and men market by separating the gender feature. Moreover, based on content, the market might also be segmented into mainly sports market, news market, and entertainment market. Understanding the specific features, such as days, time, and content, of each market would help a video become popular among the audience in its target market. For instance, students and middle-aged workers may study and work during the day and have more time in watching videos at night, whereas elders might sleep earlier and spend more time in watching videos during the day.

Limitations and mitigations

Variables: Since we don't have the variables like share numbers, subscribe numbers, and video length in this dataset, recommendations would be limited. Future updates to the dataset could be used to analyze and to give complementary recommendations.

Video comment content: In the dataset, we only have the count of comments but not the detailed content of the comments which we could do text-mining analysis to figure out what are the emotions of those comments and why audience like or dislike this video. Future updated version of this dataset is expected for further explorations.

Geographic limitation: the whole dataset contains the trending video data across five different countries. Since we have chosen the data of US to apply the analysis, recommendations might be limited from cross-country consideration. Further analysis could focus on the other four countries to provide recommendations that are regional wise or based on different countries.

Appendix A - R output

```
---
output:
  word_document: default
  html_document: default
  pdf_document: default
---
# Project Deliverable #2
# APANPS5200 Prof.Vishal Lala
# Group 10
# Xin Xin (xx2292)
# Jiaying Wu (jw3644)

# Topic: How to get more traffic for your YouTube video?

# Previous work from Deliverable #1

# load and review data
```{r}
usvideos <- read.csv("~/Desktop/5200 AAFM/Group Project/USvideos.csv")
View(usvideos)
names(usvideos)
str(usvideos)
summary(usvideos)
```
```

```
 [1] "video_id"              "trending_date"         "title"
 [4] "channel_title"         "category_id"           "publish_time"
 [7] "tags"                  "views"                 "likes"
[10] "dislikes"              "comment_count"         "thumbnail_link"
[13] "comments_disabled"     "ratings_disabled"      "video_error_or_removed"
[16] "description"
'data.frame':   16778 obs. of  16 variables:
 $ video_id              : Factor w/ 3719 levels "__-22AJoFxY",..: 262 224 447 2579 1069 1464 304 2268 1843 3059
...
 $ trending_date         : Factor w/ 84 levels "17.01.12","17.02.12",..: 14 14 14 14 14 14 14 14 14 14 ...
 $ title                 : Factor w/ 3767 levels "'Call Me by Your Name' star Timothee Chalamet on the time he
embarrassed himself with Saoirse Ronan",..: 3520 3219 2591 2366 1514 103 2704 162 3111 3672 ...
 $ channel_title         : Factor w/ 1709 levels "12 News","1theK (원더케이)",..: 238 843 1267 577 1093 679 1287 346
3 1641 ...
 $ category_id           : int  22 24 23 24 24 28 24 28 1 25 ...
 $ publish_time          : Factor w/ 3664 levels "2006-07-23T08:24:11.000Z",..: 318 287 271 291 269 323 256 274
297 295 ...
 $ tags                  : Factor w/ 3527 levels ""“Developer Update |Happy New Year | Overwatch”|“Happy New
Year”|2017|2018|Jeff Kaplan|Developer Update|Overwatc"| __truncated__,..: 2676 1722 2452 2560 2600 1388 2747 35
3205 3361 ...
 $ views                 : int  748374 2418783 3191434 343168 2095731 119180 2103417 817732 826059 256426 ...
 $ likes                 : int  57527 97185 146033 10172 132235 9763 15993 23663 3543 12654 ...
 $ dislikes              : int  2966 6146 5339 666 1989 511 2445 778 119 1363 ...
 $ comment_count         : int  15954 12703 8181 2146 17518 1434 1970 3432 340 2368 ...
 $ thumbnail_link        : Factor w/ 3719 levels "https://i.ytimg.com/vi/__-22AJoFxY/default.jpg",..: 262 224 447
2579 1069 1464 304 2268 1843 3059 ...
```

```
 $ comments_disabled     : Factor w/ 2 levels "False","True": 1 1 1 1 1 1 1 1 1 1 ...
 $ ratings_disabled      : Factor w/ 2 levels "False","True": 1 1 1 1 1 1 1 1 1 1 ...
 $ video_error_or_removed: Factor w/ 2 levels "False","True": 1 1 1 1 1 1 1 1 1 1 ...
 $ description           : Factor w/ 3864 levels "","  ","- Charities -\\nUnited Way:
http://vcunitedway.org/\\nVCCF: https://vccf.org/donate/make-a-donation/\\nHumane S"| __truncated__,..: 2731 2394
3589 3458 1411 3534 874 1487 1625 1008 ...
       video_id      trending_date
 sXP6vliZIHI:   14   17.01.12:  200
 8eo-L_3OWAQ:   13   17.02.12:  200
 b2EOxGUuBWA:   13   17.03.12:  200
 cMD63TwzB1o:   13   17.04.12:  200
 E60w-0q8Y0Y:   13   17.05.12:  200
 fEMxFSjmo5Q:   13   17.06.12:  200
 (Other)    :16699   (Other) :15578
                                                                        title
 Selena Gomez, Marshmello - Wolves                                         :   18
 Cardi B - Bartier Cardi (feat. 21 Savage) [Official Audio]                :   14
 2016 vs 2017                                                              :   13
 Cut for Time: Hallmark Channel Christmas Promo (James Franco) - SNL       :   13
 G-Eazy - No Limit REMIX ft. A$AP Rocky, Cardi B, French Montana, Juicy J, Belly:   13
 GoPro: Gorilla Tickling at the GRACE Center                               :   13
 (Other)                                                                   :16694
    channel_title    category_id                  publish_time
 NFL       :   84   Min.   : 1   2017-11-17T05:00:01.000Z:   20
 Refinery29:   84   1st Qu.:17   2018-01-12T05:00:01.000Z:   18
 Vox       :   83   Median :24   2017-11-17T05:00:00.000Z:   16
 ESPN      :   81   Mean   :20   2017-11-22T18:00:02.000Z:   15
 Netflix   :   80   3rd Qu.:25   2018-01-17T15:00:02.000Z:   15
 NBA       :   79   Max.   :43   2017-12-08T05:00:01.000Z:   14
 (Other)   :16287                (Other)                 :16680

tags
 [none]
:  770
 The Late Show|Stephen Colbert|Colbert|Late Show|celebrities|late night|talk show|skits|bit|monologue|The Late
Late Show|Late Late Show|letterman|david letterman|comedian|impressions|CBS|joke|jokes|funny|funny video|funny
videos|humor|celebrity|celeb|hollywood|famous|James Corden|Corden|Comedy:   63
 James Corden|The Late Late Show|Colbert|late night|late night show|Stephen
Colbert|Comedy|monologue|comedian|impressions|celebrities|carpool|karaoke|CBS|Late Late
Show|Corden|joke|jokes|funny|funny video|funny videos|humor|celebrity|celeb|hollywood|famous
:   58
 Viral|Video|Epic
:   38
 cupcakes|how to make vanilla cupcakes|over the top recipes|easy cupcake recipes|vanilla cupcakes|chocolate
cupcakes|french macarons|how to make macarons|the scran line|the scranline|nick makrides|pastry design|how to pipe
cupcakes                                                                  :   31
 nba|basketball|starters
:   19
 (Other)
:15799
```

```
     views              likes            dislikes       comment_count
 Min.   :      549   Min.   :      0   Min.   :      0   Min.   :      0
 1st Qu.:    95419   1st Qu.:   1906   1st Qu.:     87   1st Qu.:    279
 Median :   322766   Median :   8844   Median :    323   Median :   1017
 Mean   :  1228605   Mean   :  44662   Mean   :   3204   Mean   :   5880
 3rd Qu.:   995030   3rd Qu.:  29064   3rd Qu.:   1125   3rd Qu.:   3377
 Max.   :149376127   Max.   :3093544   Max.   :1674420   Max.   :1361580

                                       thumbnail_link  comments_disabled
 https://i.ytimg.com/vi/sXP6vliZIHI/default.jpg:   14   False:16488
 https://i.ytimg.com/vi/8eo-L_3OWAQ/default.jpg:   13   True :  290
 https://i.ytimg.com/vi/b2EOxGUuBWA/default.jpg:   13
 https://i.ytimg.com/vi/cMD63TwzB1o/default.jpg:   13
 https://i.ytimg.com/vi/E60w-0q8Y0Y/default.jpg:   13
 https://i.ytimg.com/vi/fEMxFSjmo5Q/default.jpg:   13
 (Other)                                       :16699
 ratings_disabled video_error_or_removed
 False:16679      False:16774
 True :   99      True :    4
```

```
description

:  309
 ▶ Listen LIVE: http://power1051fm.com/\\n▶ Facebook: https://www.facebook.com/Power1051NY/\\n▶ Twitter:
https://twitter.com/power1051/\\n▶ Instagram: https://www.instagram.com/power1051/
:   38
 Jukin Media Verified (Original) * For licensing / permission to use: Contact -
licensing(at)jukinmediadotcom\\nSubmit your videos here: http://bit.ly/2iFnUya
:   38
 To get this complete recipe with instructions and measurements, check out my website:
http://www.LaurainteKitchen.com\\n\\nInstagram: http://www.instagram.com/mrsvitale\\n\\nOfficial Facebook Page:
http://www.facebook.com/LaurainteKitchen\\n\\nContact: Business@LaurainteKitchen.com\\n\\nTwitter:
@Lauraskitchen
:   26
 Get Cut swag here: http://cut.com/shop\\n\\nDon't forget to subscribe and follow us!\\nYouTube:
http://cut.com/youtube \\nFacebook: http://cut.com/facebook \\nInstagram: http://cut.com/instagram \\nSnapchat:
@watchcut\\n\\nProduced, directed, and edited by https://cut.com \\n\\nWant to work with us? http://cut.com/hiring
\\nWant to be in a video? http://cut.com/casting \\nLove Cut? Fill out this form for exclusive updates:
http://cut.com/fanform \\n\\nWant to sponsor a video? http://cut.com/sponsorships  \\nFor licensing inquiries:
http://cut.com/licensing:   15
 Cardi B - Bartier Cardi (feat. 21 Savage) available now!\\nStream/Download:
https://lnk.to/BartierCardiID\\n\\nExclusive Bardi Gang merchandise available here:
http://smarturl.it/BardiGangMerchYT\\n\\nFollow Cardi
B\\nhttp://Facebook.com/IAmCardiB\\nhttp://Instagram.com/IAmCardiB\\nhttp://Twitter.com/IAmCardiB\\nhttp://Soundclc

:   14
 (Other)
:16338
```

```
# Deal with video repetitions of the data
# Command learnt from https://stackoverflow.com
```{r}
sum(duplicated(usvideos$title))
sum(duplicated(usvideos$channel_title))
usvideos <- usvideos[!duplicated(usvideos$title), ]
usvideos <- usvideos[!duplicated(usvideos$channel_title), ]
sum(duplicated(usvideos$title))
sum(duplicated(usvideos$channel_title))
```
```

```
[1] 13011
[1] 15069
[1] 0
[1] 0
```

```
# Seperate the date and time
# Command learnt from http://garrettgman.github.io/tidying/
```{r}
library(tidyr)
usvideos <- separate(usvideos, publish_time, into = c("release_date", "release_time"), sep = "T")
names(usvideos)
usvideos[1:5,6:7]
```
```

R Console

data.frame
5 x 2

	release_date <chr>	release_time <chr>
1	2017-11-13	17:13:01.000Z
2	2017-11-13	07:30:00.000Z
3	2017-11-12	19:05:24.000Z
4	2017-11-13	11:00:04.000Z
5	2017-11-12	18:01:41.000Z

5 rows

```
# Check missing Values, and identify location of missing values by listing the rows containing missing values
# By this command, we can see that there is no missing value in this data set
```{r}
usvideos[is.na(usvideos),]
```
```

0 rows | 1-9 of 17 columns

```
# The tag column contains many different types of video categories,
# which is difficult to sort, and this issue would not affect our analytics.
# So, we ignored this issue.

# Dataset is ready for analytics
```{r}
View(usvideos)
save.image("~/Desktop/Analytical data usvideos.RData")
```
```

```
# Work on Deliverable #2

# After review deliverable #1 output and deliverable #2 questions, we need to further clean the data.

# Step 1: remove the columns of "category_id" and "video_error_or_removed" as we will not use this data in our
analysis
```

```{r}
usvideos$category_id <- NULL
usvideos$video_error_or_removed <- NULL
names(usvideos)
```

```
 [1] "video_id"          "trending_date"     "title"
 [4] "channel_title"     "release_date"      "release_time"
 [7] "tags"              "views"             "likes"
[10] "dislikes"          "comment_count"     "thumbnail_link"
[13] "comments_disabled" "ratings_disabled"  "description"
```

```
# Step 2: only remain the hours of "release_time" by removing minutes, seconds and the following ".000Z" because we
don't need to see the specific time when analyzing
```

```{r}
library(stringr)
usvideos$release_time <- str_sub(usvideos$release_time, 1, str_length(usvideos$release_time)-11)
usvideos$release_time[1:5]
```

```
[1] "17" "07" "19" "11" "18"
```

```
# Step 3: add "release_weekday" column as we want to analyze if the number of views is related to the day of week as
well
```

```{r}
usvideos$release_weekday <- weekdays(as.Date(usvideos$release_date))
usvideos$release_weekday[1:5]
```

```
[1] "Monday" "Monday" "Sunday" "Monday" "Sunday"
```

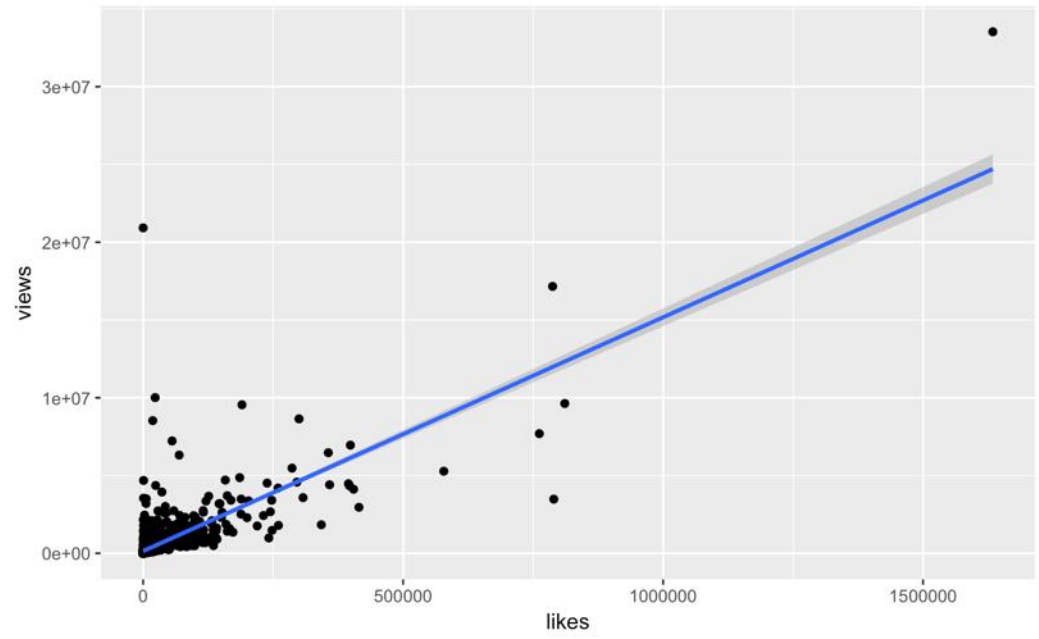
```
# Now the dataset is ready for analytics.

# First, we would utilize liner regression models to learn the existence and strength of the correlations between:
dependent variable: number of views, and independent variables: number of likes, number of dislikes, number of
comments.
```

```
# Regression Model1: views VS. likes
```{r}
regModel1 = lm(views~likes,usvideos)
summary(regModel1)
```
```

```
Call:
lm(formula = views ~ likes, data = usvideos)

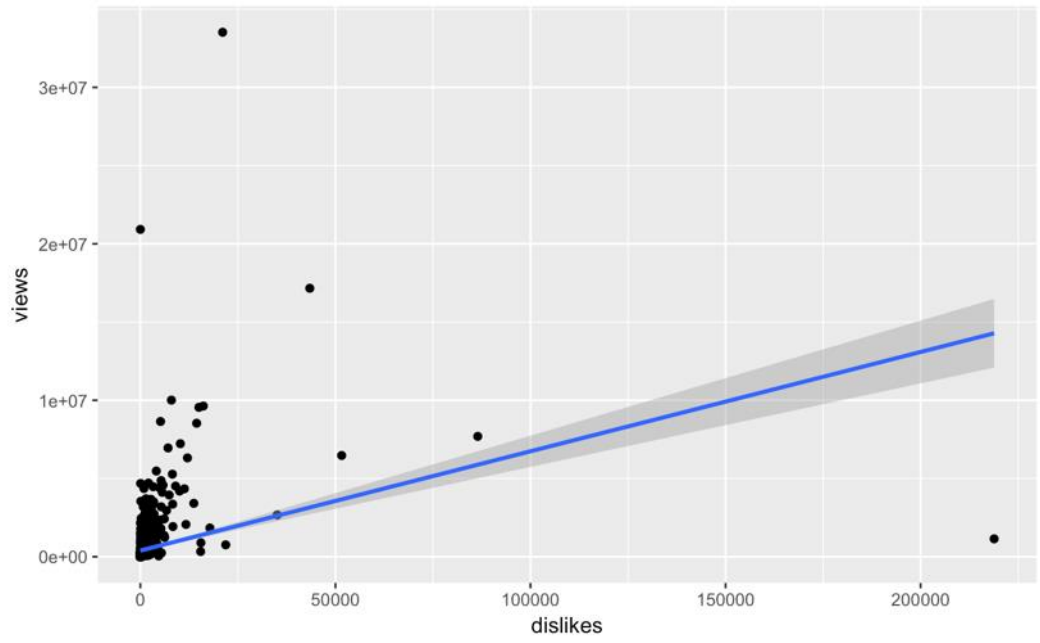
Residuals:
     Min       1Q   Median       3Q      Max
-8530092  -142234  -121483   -28452 20777220

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.446e+05  2.179e+04   6.634 4.36e-11 ***
likes       1.503e+01  2.980e-01  50.432  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 865500 on 1701 degrees of freedom
Multiple R-squared:  0.5992,    Adjusted R-squared:  0.599
F-statistic:  2543 on 1 and 1701 DF,  p-value: < 2.2e-16
```

```
# Plot regression line1: views VS. likes
```{r}
library(ggplot2)
ggplot(data=usvideos,aes(x=likes,y=views))+
  geom_point()+
  geom_smooth(method='lm')
```
```



```
# Regression Model2: views VS. dislikes
```{r}
regModel2 = lm(views~dislikes,usvideos)
summary(regModel2)
```
```

```
Call:
lm(formula = views ~ dislikes, data = usvideos)

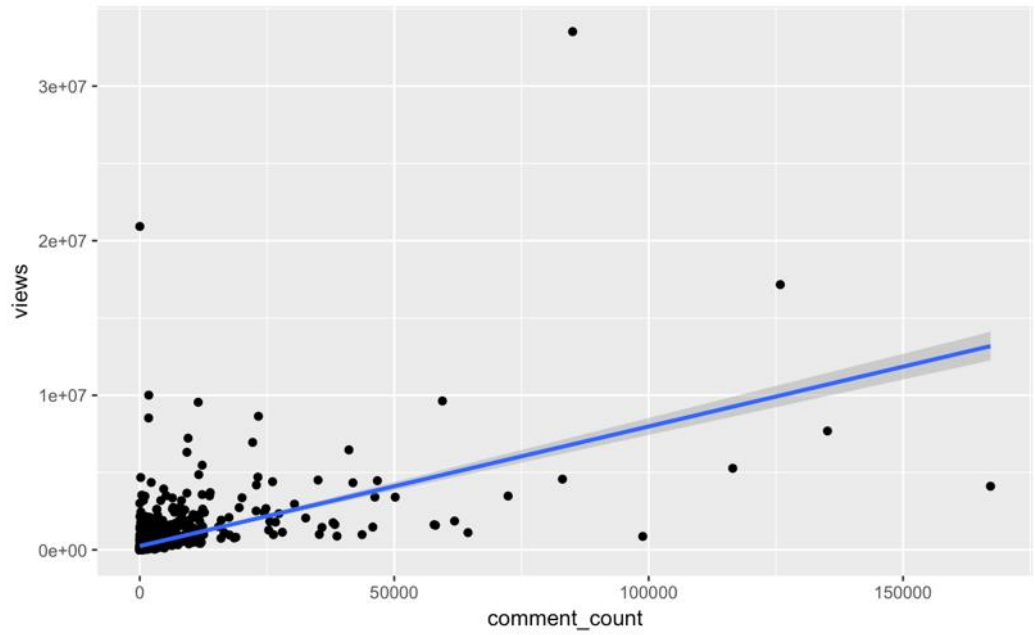
Residuals:
      Min        1Q    Median        3Q       Max
-13140057   -369752   -301467    -60514  31793028

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 392489.43   31993.67   12.27   <2e-16 ***
dislikes        63.47       5.13   12.37   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1309000 on 1701 degrees of freedom
Multiple R-squared:  0.08256,   Adjusted R-squared:  0.08202
F-statistic: 153.1 on 1 and 1701 DF,  p-value: < 2.2e-16
```

```
# Plot regression line2: views VS. dislikes
```{r}
ggplot(data=usvideos,aes(x=dislikes,y=views))+
  geom_point()+
  geom_smooth(method='lm')
```
```



```
# Regression Model3: views VS. comment_count
```{r}
regModel3 = lm(views~comment_count,usvideos)
summary(regModel3)
```
```

```
Call:
lm(formula = views ~ comment_count, data = usvideos)

Residuals:
     Min       1Q   Median       3Q      Max
-9062836  -238611  -197392   -45679 26696403

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.475e+05  2.866e+04   8.635   <2e-16 ***
comment_count 7.735e+01  2.869e+00  26.964   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1144000 on 1701 degrees of freedom
Multiple R-squared:  0.2994,    Adjusted R-squared:  0.299
F-statistic:   727 on 1 and 1701 DF,  p-value: < 2.2e-16
```

```
# Plot regression line3: views VS. comment_count
```{r}
ggplot(data=usvideos,aes(x=comment_count,y=views))+
  geom_point()+
  geom_smooth(method='lm')
```
```



```
# Model with many independent variables to verify the correlations and strengths
```{r}
model6 = lm(views~likes+dislikes+comment_count+release_time+release_weekday,data=usvideos)
summary(model6)
```
```

```
Call:
lm(formula = views ~ likes + dislikes + comment_count + release_time +
    release_weekday, data = usvideos)

Residuals:
     Min       1Q   Median       3Q      Max
-8473249  -183475   -93191    36608 20451351

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              2.010e+05  1.054e+05   1.907   0.0567 .
likes                    1.665e+01  4.665e-01  35.686  < 2e-16 ***
dislikes                 1.674e+01  3.617e+00   4.628 3.98e-06 ***
comment_count           -1.962e+01  3.455e+00  -5.680 1.59e-08 ***
release_time01          -4.588e+04  1.460e+05  -0.314   0.7535
release_time02          -1.436e+04  1.544e+05  -0.093   0.9259
release_time03          -6.253e+04  1.498e+05  -0.418   0.6763
release_time04           2.853e+04  1.684e+05   0.169   0.8655
release_time05          -2.377e+05  1.386e+05  -1.715   0.0865 .
release_time06          -1.079e+05  1.927e+05  -0.560   0.5759
release_time07           1.867e+05  1.851e+05   1.008   0.3134
release_time08           3.562e+04  1.578e+05   0.226   0.8215
release_time09          -2.084e+04  1.903e+05  -0.110   0.9128
release_time10          -7.163e+04  2.056e+05  -0.348   0.7276
release_time11           5.415e+04  1.643e+05   0.330   0.7418
release_time12          -1.108e+05  1.638e+05  -0.676   0.4991
release_time13           1.588e+05  1.382e+05   1.150   0.2504
release_time14          -8.900e+04  1.311e+05  -0.679   0.4975
release_time15          -3.019e+04  1.203e+05  -0.251   0.8019
release_time16          -1.988e+05  1.215e+05  -1.636   0.1020
release_time17          -4.971e+04  1.188e+05  -0.418   0.6758
release_time18          -5.125e+04  1.215e+05  -0.422   0.6733
release_time19          -3.231e+04  1.308e+05  -0.247   0.8050
release_time20           1.404e+05  1.280e+05   1.097   0.2728
release_time21          -2.091e+04  1.343e+05  -0.156   0.8763
release_time22          -6.921e+04  1.314e+05  -0.527   0.5985
release_time23          -1.106e+05  1.357e+05  -0.816   0.4149
release_weekdayMonday    9.338e+02  7.343e+04   0.013   0.9899
release_weekdaySaturday -8.474e+04  8.401e+04  -1.009   0.3132
release_weekdaySunday    1.291e+05  8.647e+04   1.493   0.1356
release_weekdayThursday  1.131e+04  7.275e+04   0.155   0.8765
release_weekdayTuesday  -6.666e+04  7.208e+04  -0.925   0.3552
release_weekdayWednesday -4.036e+04  7.276e+04  -0.555   0.5792
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 855200 on 1670 degrees of freedom
Multiple R-squared:  0.6158,    Adjusted R-squared:  0.6085
F-statistic: 83.66 on 32 and 1670 DF,  p-value: < 2.2e-16
```

```
# Additional analysis:
# During our research, some "experts" claimed that one should add as many tags as you can think of to get more views. Also long video descriptions can help get more views. Thus we did a linear regression analysis between the video views and tag/description length and found there are no correlations in between.

# views VS. tag length
```{r}
taglength <- nchar(as.character(usvideos$tags))
taglength[1:5]

regModel7 = lm(views~taglength,usvideos)
summary(regModel7)
```
```

```
[1]  15  96 272 425  94

Call:
lm(formula = views ~ taglength, data = usvideos)

Residuals:
     Min       1Q   Median       3Q      Max
 -516849  -399173  -337642   -72697 33108079

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 391688.3    53473.9   7.325 3.68e-13 ***
taglength      280.6      229.4   1.223    0.221
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

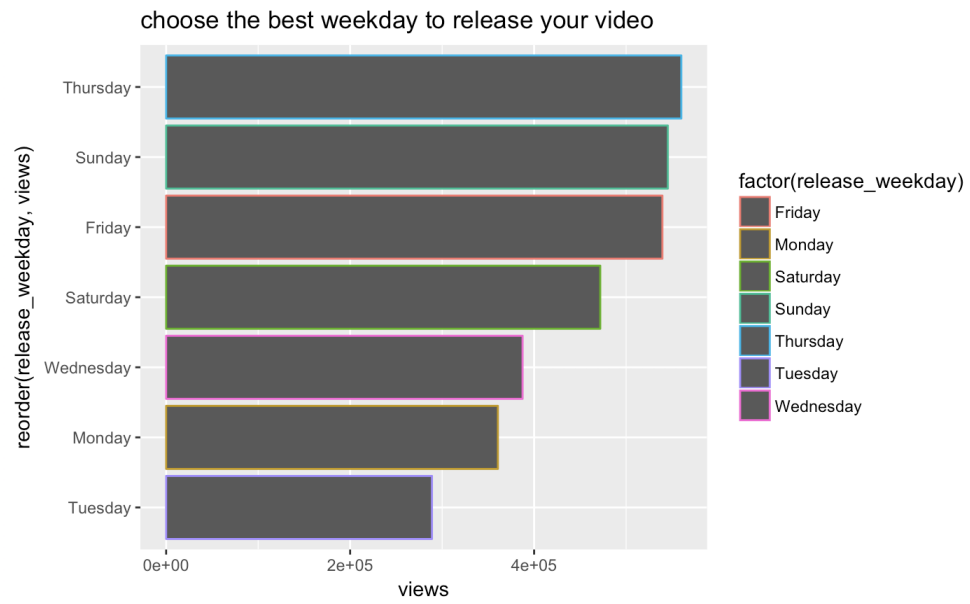
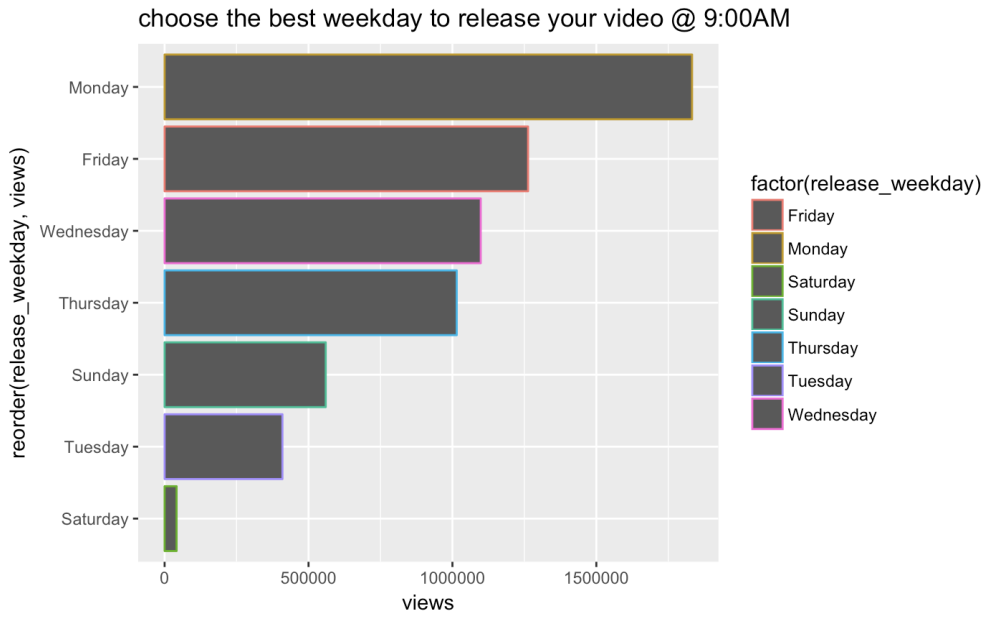
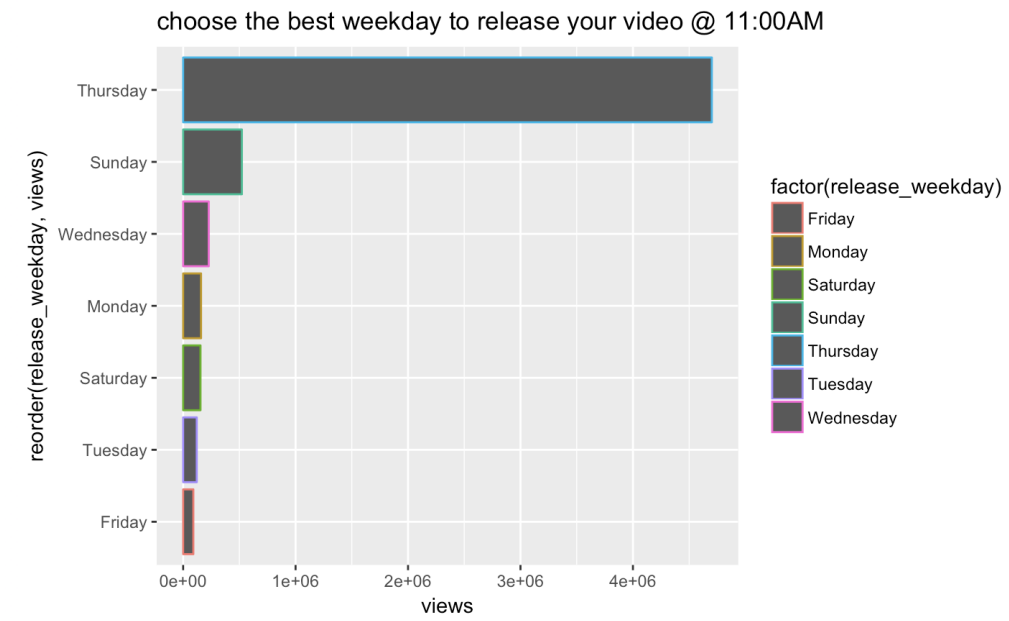
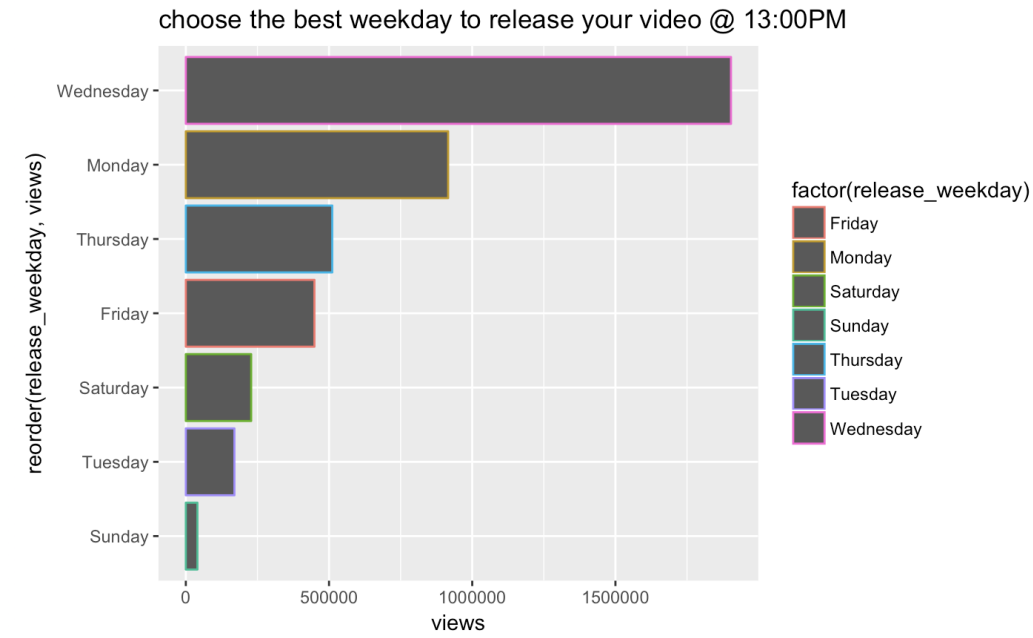
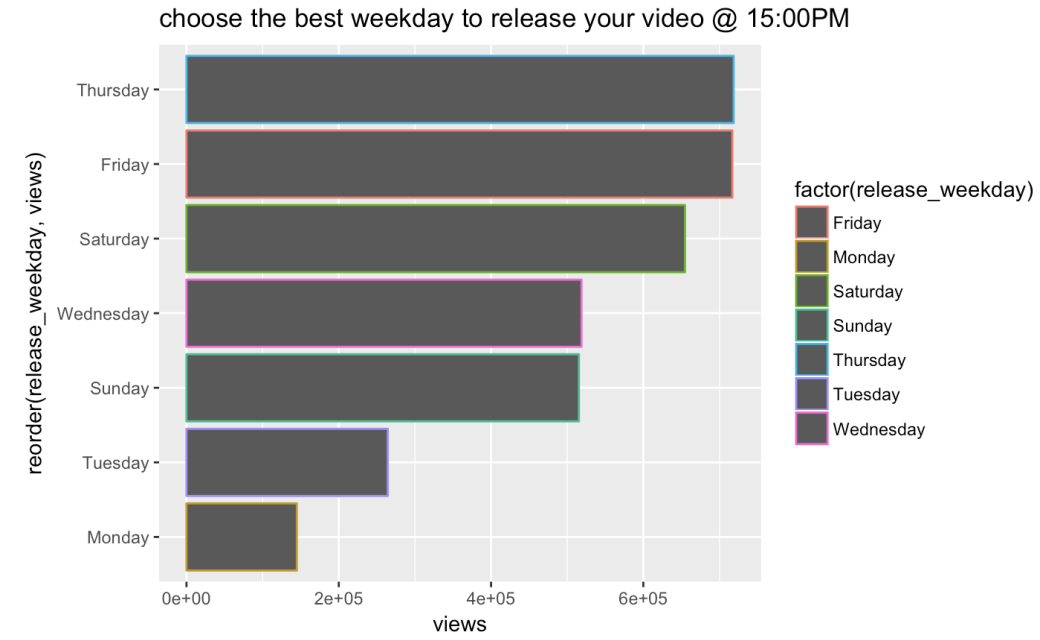
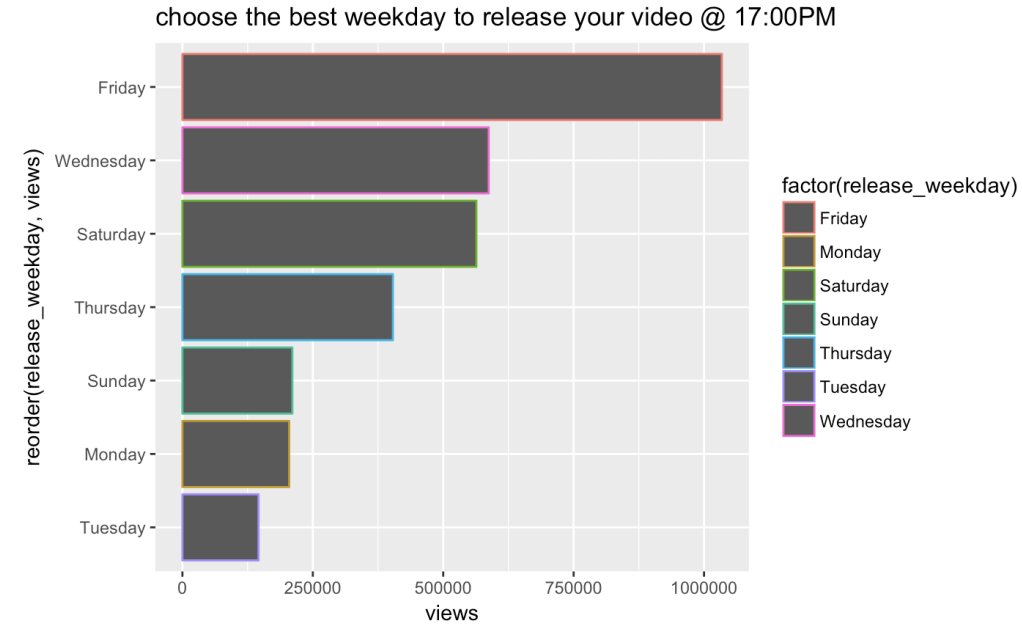
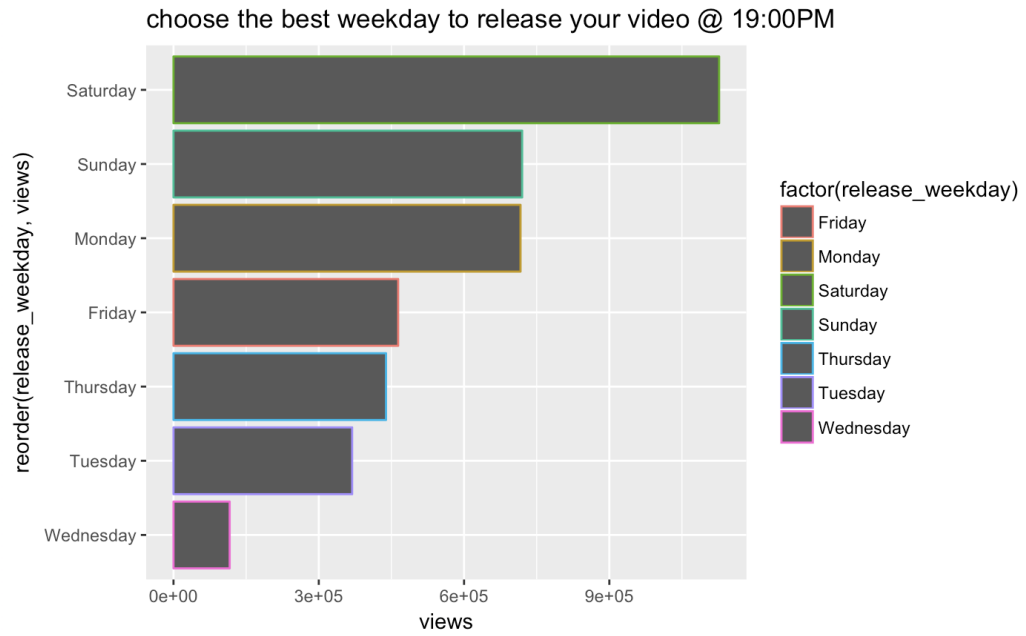
Residual standard error: 1367000 on 1701 degrees of freedom
Multiple R-squared:  0.0008789, Adjusted R-squared:  0.0002915
F-statistic: 1.496 on 1 and 1701 DF,  p-value: 0.2214
```

```
# views VS. description length
```{r}
deslength <- nchar(as.character(usvideos$description))
deslength[1:5]

regModel8 = lm(views~deslength,usvideos)
summary(regModel8)
```
```

```
[1] 1410  630 1177 1403  636

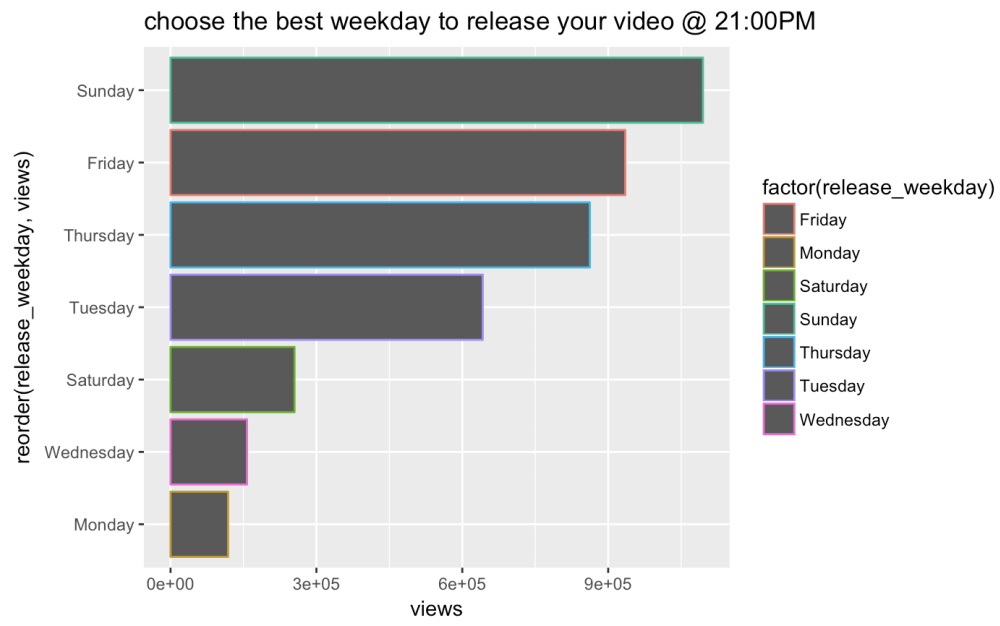
Call:
lm(formula = views ~ deslength, data = usvideos)

Residuals:
     Min       1Q   Median       3Q      Max
 -702367  -390763  -332005   -70886 33045845

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 379638.11   46595.85   8.147 7.12e-16 ***
deslength       78.51      40.62   1.933   0.0534 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

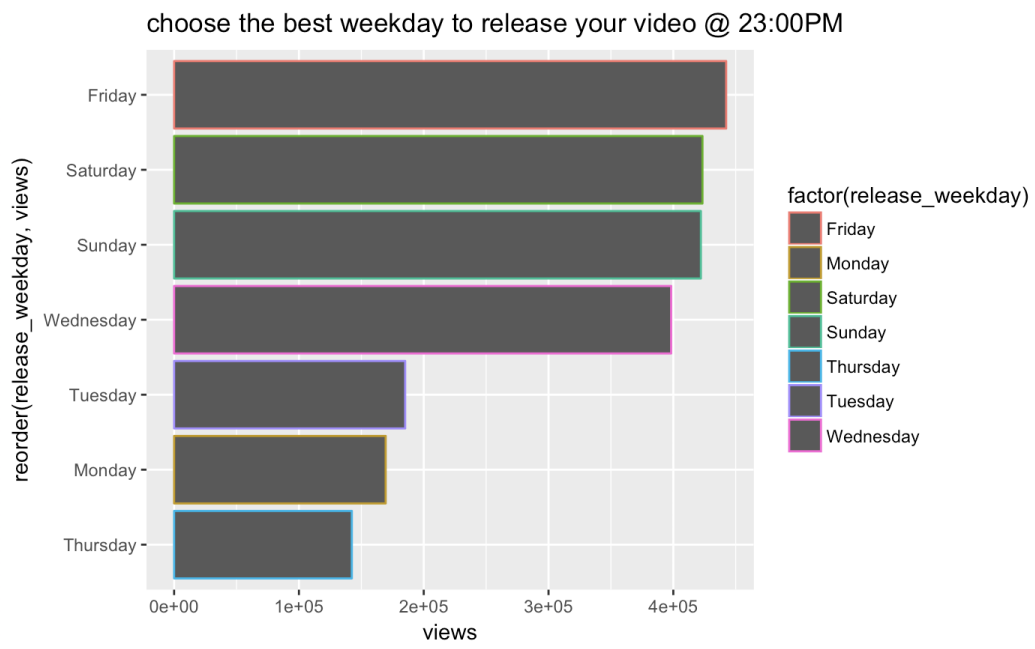
Residual standard error: 1366000 on 1701 degrees of freedom
Multiple R-squared:  0.002192,  Adjusted R-squared:  0.001605
F-statistic: 3.737 on 1 and 1701 DF,  p-value: 0.0534
```

```
# Explore relationship between views VS. release day and time

# Barchart with release_weekday
```{r}
ggplot(aes(x=reorder(release_weekday,views), y=views, color=factor(release_weekday)),data=usvideos)+
  geom_bar(stat='summary',fun.y='mean')+ggtitle('choose the best weekday to release your video')+coord_flip()
```
```




```
# Barchart with release_weekday based on different time slot
# 9:00 AM
```{r}
ggplot(aes(x=reorder(release_weekday,views), y=views,
color=factor(release_weekday)),data=usvideos[usvideos$release_time == "09",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best weekday to release your video @ 9:00AM')+coord_flip()
```
```



```
# 11:00 AM
```{r}
ggplot(aes(x=reorder(release_weekday,views), y=views,
color=factor(release_weekday)),data=usvideos[usvideos$release_time == "11",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best weekday to release your video @ 11:00AM')+coord_flip()
```
```




```
# 13:00 PM
```{r}
ggplot(aes(x=reorder(release_weekday,views), y=views,
color=factor(release_weekday)),data=usvideos[usvideos$release_time == "13",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best weekday to release your video @ 13:00PM')+coord_flip()
```
```



```
# 15:00 PM
```{r}
ggplot(aes(x=reorder(release_weekday,views), y=views,
color=factor(release_weekday)),data=usvideos[usvideos$release_time == "15",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best weekday to release your video @ 15:00PM')+coord_flip()
```
```




```
# 17:00 PM
```{r}
ggplot(aes(x=reorder(release_weekday,views), y=views,
color=factor(release_weekday)),data=usvideos[usvideos$release_time == "17",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best weekday to release your video @ 17:00PM')+coord_flip()
```
```



```
# 19:00 PM
```{r}
ggplot(aes(x=reorder(release_weekday,views), y=views,
color=factor(release_weekday)),data=usvideos[usvideos$release_time == "19",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best weekday to release your video @ 19:00PM')+coord_flip()
```
```




```
# 21:00 PM
```{r}
ggplot(aes(x=reorder(release_weekday,views), y=views,
color=factor(release_weekday)),data=usvideos[usvideos$release_time == "21",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best weekday to release your video @ 21:00PM')+coord_flip()
```
```



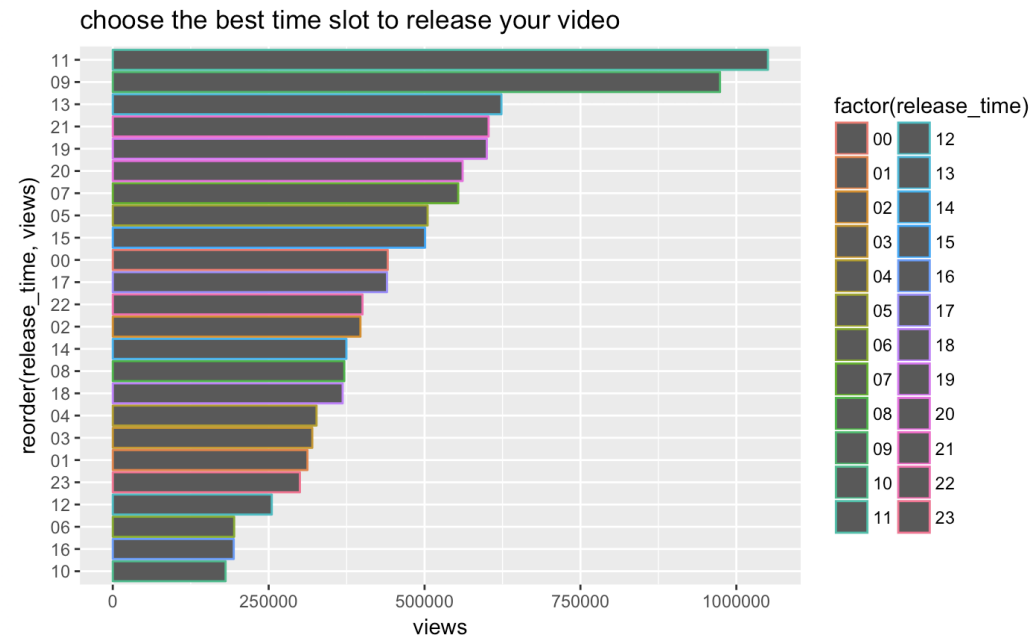
23:00 PM

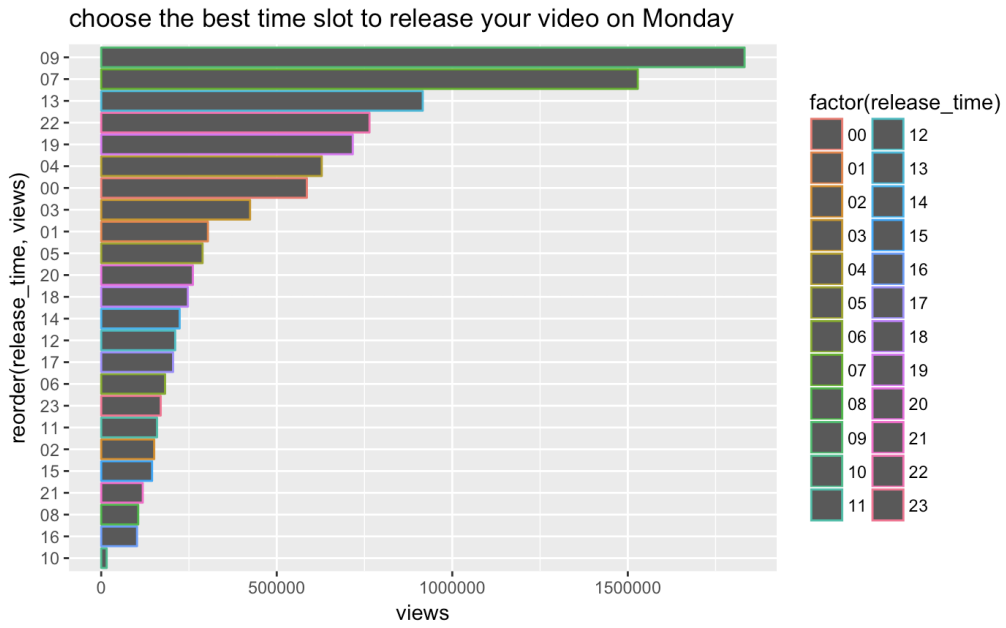
```{r}
ggplot(aes(x=reorder(release_weekday,views), y=views,
color=factor(release_weekday)),data=usvideos[usvideos$release_time == "23",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best weekday to release your video @ 23:00PM')+coord_flip()
```

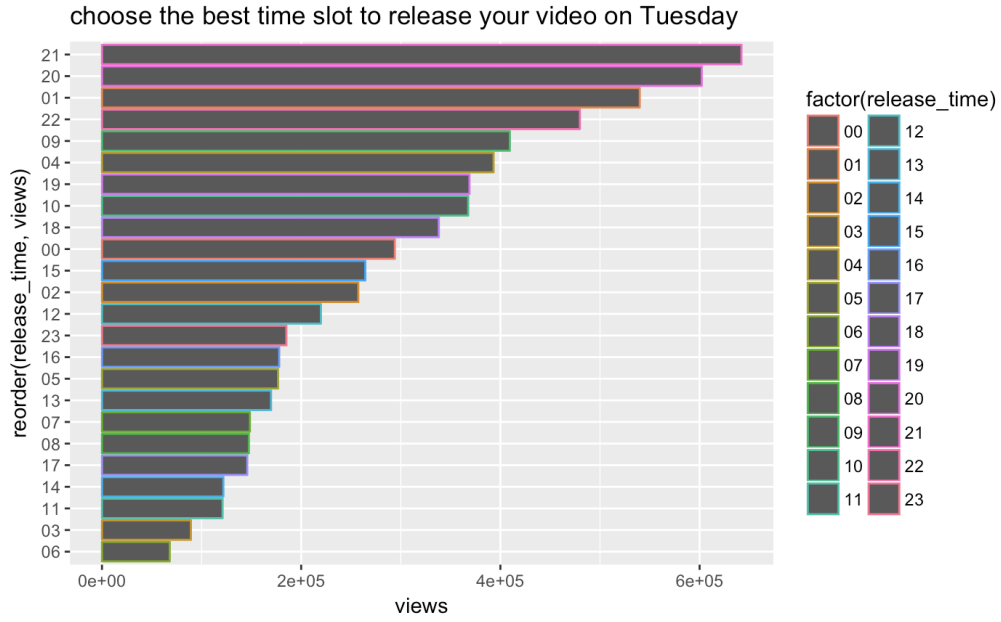


Barchart with release_time

```{r}
ggplot(aes(x=reorder(release_time, views), y=views, color=factor(release_time)),data=usvideos)+
  geom_bar(stat='summary',fun.y='mean')+ggtitle('choose the best time slot to release your video')+coord_flip()
```

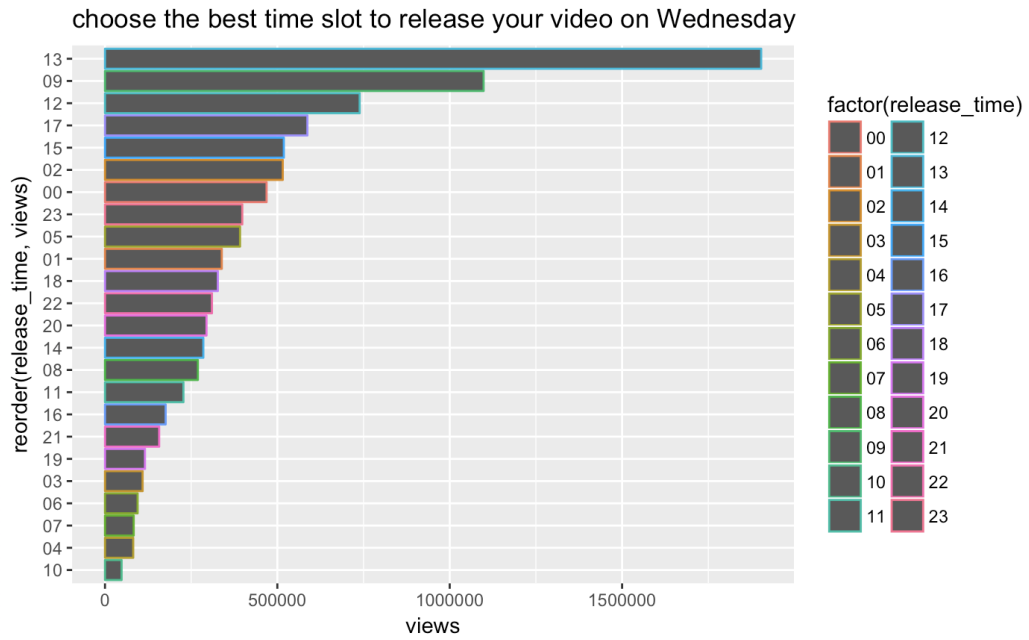


```
# Barchart with release_time based on different release_weekday
# Monday
```{r}
ggplot(aes(x=reorder(release_time,views), y=views,
color=factor(release_time)),data=usvideos[usvideos$release_weekday == "Monday",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best time slot to release your video on Monday')+coord_flip()
```
```




```
# Tuesday
```{r}
ggplot(aes(x=reorder(release_time,views), y=views,
color=factor(release_time)),data=usvideos[usvideos$release_weekday == "Tuesday",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best time slot to release your video on Tuesday')+coord_flip()
```
```



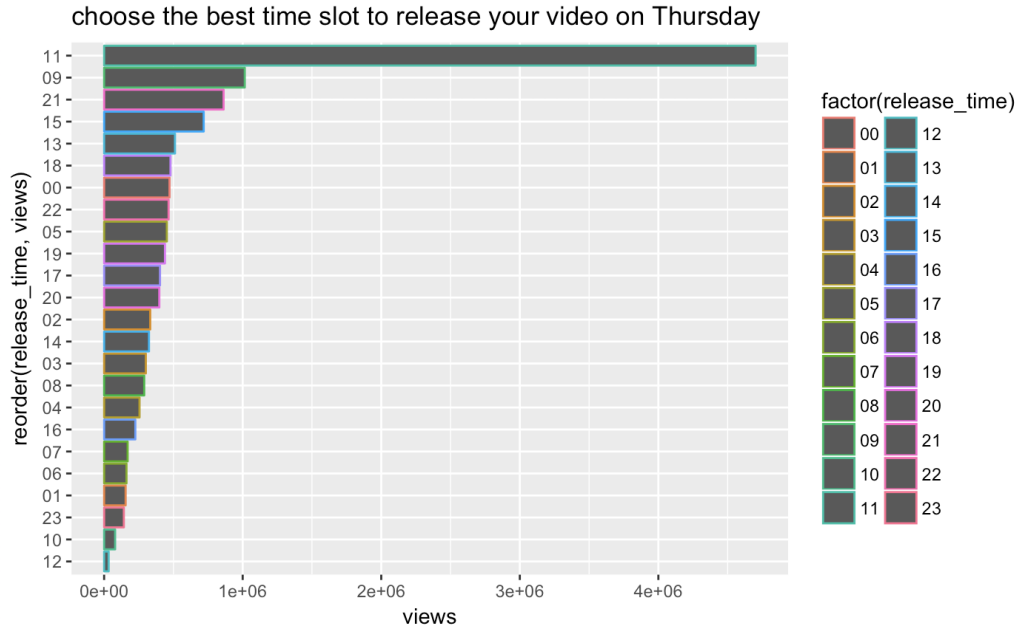
Wednesday

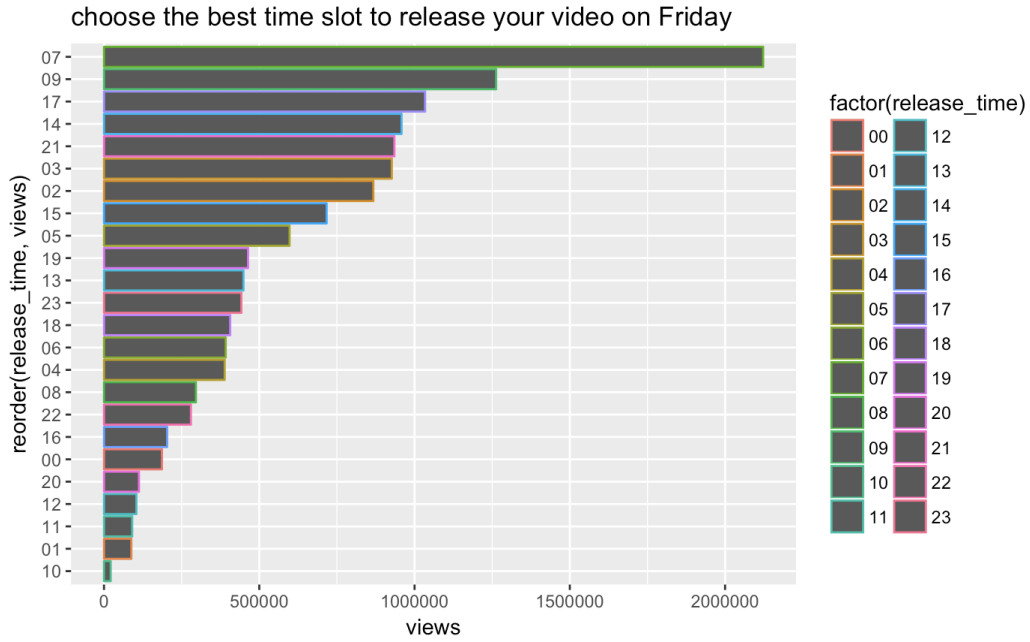
```{r}
ggplot(aes(x=reorder(release_time,views), y=views,
color=factor(release_time)),data=usvideos[usvideos$release_weekday == "Wednesday",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best time slot to release your video on Wednesday')+coord_flip()
```

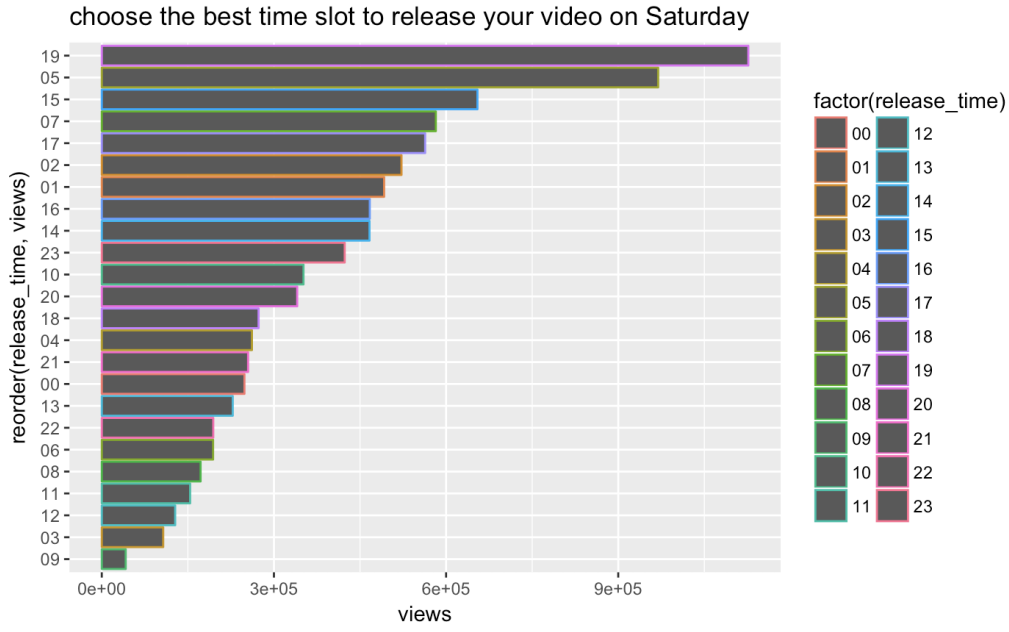


Thursday

```{r}
ggplot(aes(x=reorder(release_time,views), y=views,
color=factor(release_time)),data=usvideos[usvideos$release_weekday == "Thursday",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best time slot to release your video on Thursday')+coord_flip()
```

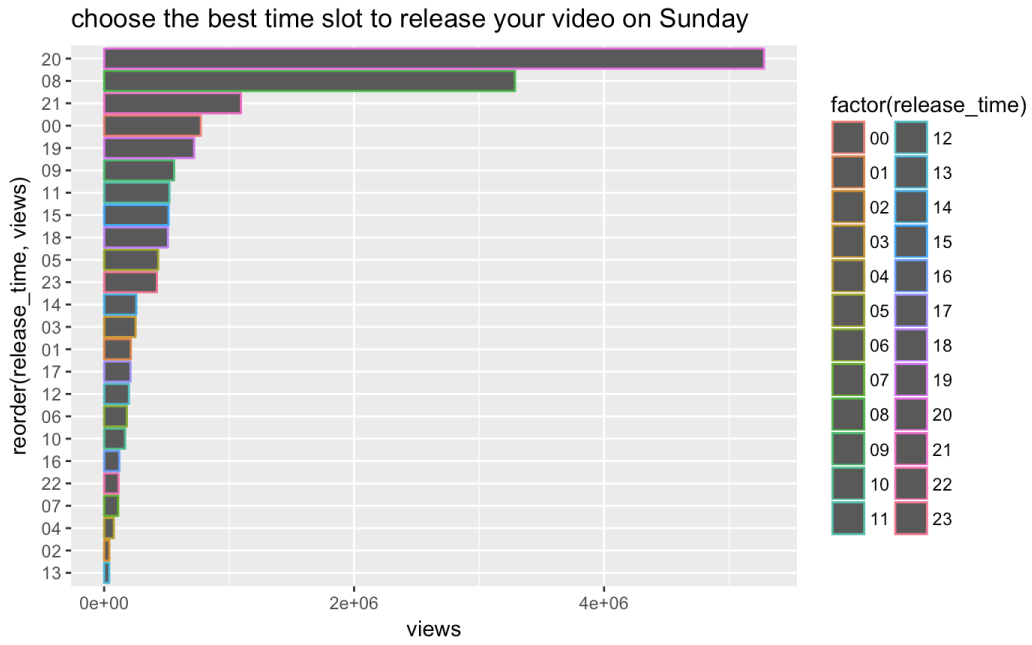


```
# Friday
```{r}
ggplot(aes(x=reorder(release_time,views), y=views,
color=factor(release_time)),data=usvideos[usvideos$release_weekday == "Friday",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best time slot to release your video on Friday')+coord_flip()
```
```




```
# Saturday
```{r}
ggplot(aes(x=reorder(release_time,views), y=views,
color=factor(release_time)),data=usvideos[usvideos$release_weekday == "Saturday",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best time slot to release your video on Saturday')+coord_flip()
```
```



Sunday

```{r}
ggplot(aes(x=reorder(release_time,views), y=views,
color=factor(release_time)),data=usvideos[usvideos$release_weekday == "Sunday",])+
  geom_bar(stat='summary',fun.y='mean')+
  ggtitle('choose the best time slot to release your video on Sunday')+coord_flip()
```



```
# Second, in order to explore the potential information offered by the non-numeric variables in the data, we would apply the text-mining methods.

# Explore the general sentiments of YouTube trending list videos
```{r}
library(tidytext)
library(dplyr)
library(twitteR)
library(ROAuth)
library(rlang)

get_sentiments('bing')%>%
  group_by(sentiment)%>%
  count()
```
```

grouped_df
2 x 2

sentiment <chr>	n <int>
negative	4782
positive	2006

2 rows

```
# Extract the emotions described in the title, tag and descriptions of YouTube trending list videos
```{r}
get_sentiments('nrc')%>%
  group_by(sentiment)%>%
  count()
```
```

sentiment <chr>	n <int>
anger	1247
anticipation	839
disgust	1058
fear	1476
joy	689
negative	3324
positive	2312
sadness	1191
surprise	534
trust	1231

1–10 of 10 rows

```
# Use wordcloud to extract the keywords of trending video titles
# Install and load the packages
```{r}
install.packages("tm")
install.packages("SnowballC")
install.packages("wordcloud")
install.packages("RColorBrewer")
library("tm")
library("SnowballC")
library("wordcloud")
library("RColorBrewer")
```
```

```
trying URL 'https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.4/tm_0.7-3.tgz'
Content type 'application/x-gzip' length 969407 bytes (946 KB)
==================================================
downloaded 946 KB


The downloaded binary packages are in
        /var/folders/5_/t3qgxdt139b3xysq88l8s0gc0000gp/T//RtmpvrK6Bn/downloaded_packages
Error in install.packages : Updating loaded packages
trying URL 'https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.4/wordcloud_2.5.tgz'
Content type 'application/x-gzip' length 143945 bytes (140 KB)
==================================================
downloaded 140 KB

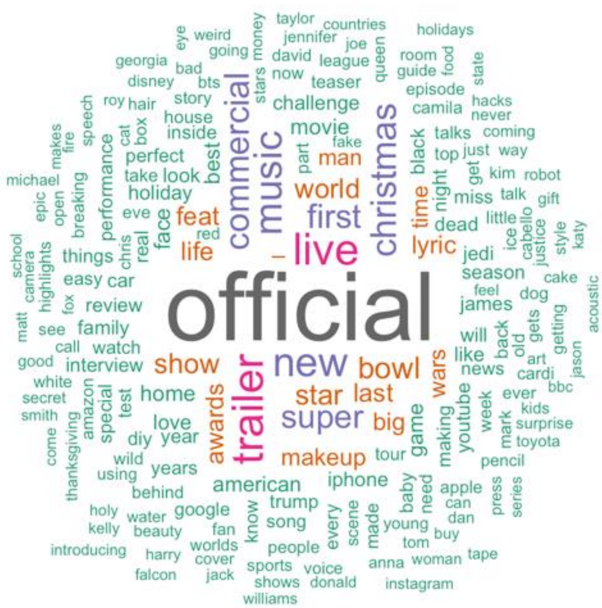
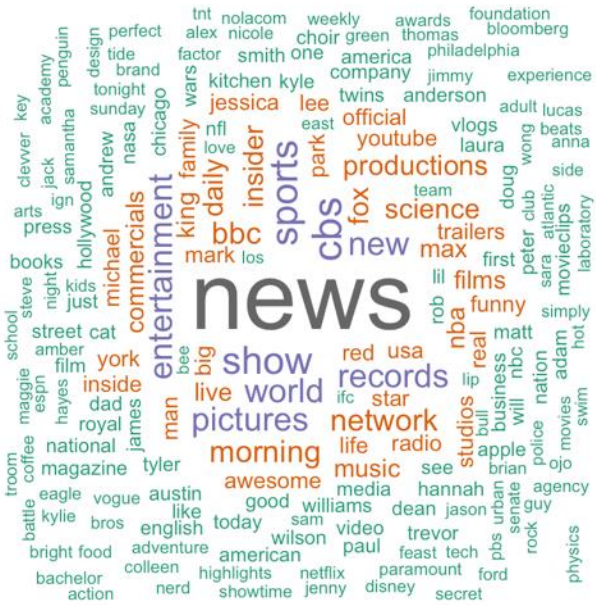

The downloaded binary packages are in
        /var/folders/5_/t3qgxdt139b3xysq88l8s0gc0000gp/T//RtmpvrK6Bn/downloaded_packages
trying URL 'https://cran.rstudio.com/bin/macosx/el-capitan/contrib/3.4/RColorBrewer_1.1-2.tgz'
Content type 'application/x-gzip' length 24322 bytes (23 KB)
==================================================
downloaded 23 KB


The downloaded binary packages are in
        /var/folders/5_/t3qgxdt139b3xysq88l8s0gc0000gp/T//RtmpvrK6Bn/downloaded_packages
Loading required package: NLP

Attaching package: 'NLP'

The following object is masked from 'package:ggplot2':

    annotate

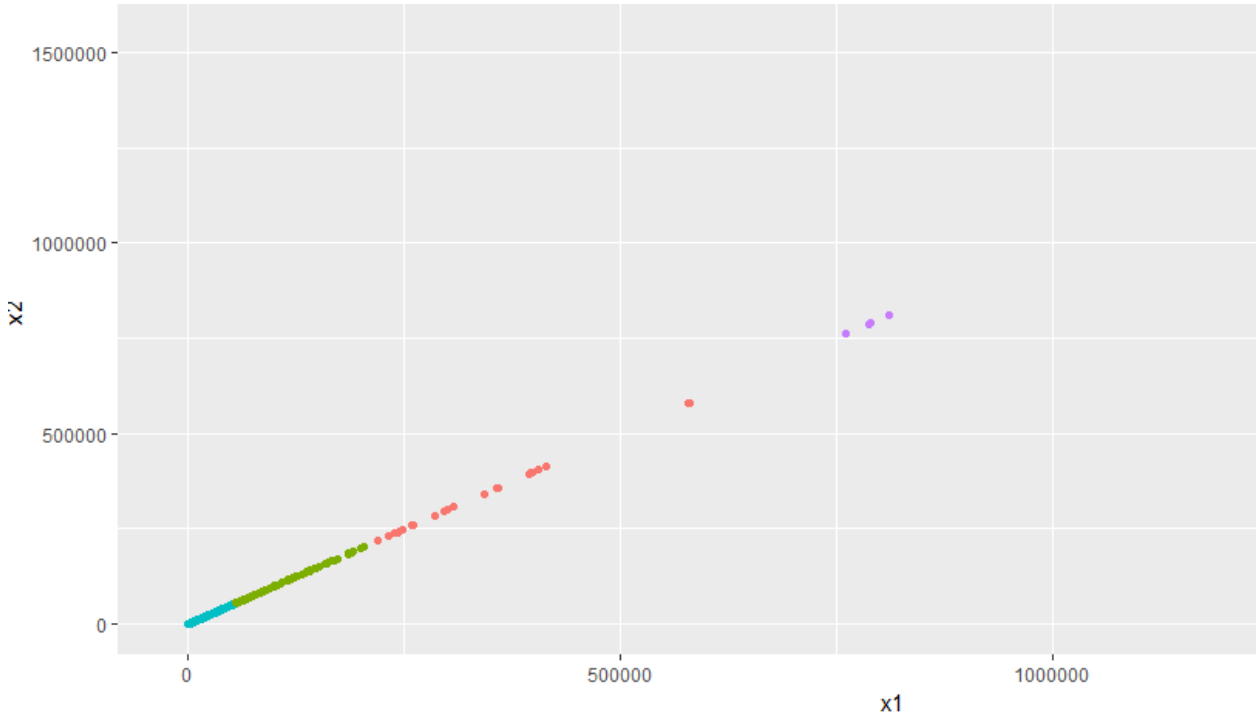
Loading required package: RColorBrewer
```

```
# Import the title text, and load the title text as a corpus
```{r}
titletext <- usvideos$title
titledocs <- Corpus(VectorSource(titletext))
```
```

```
# Further clean the title content:
# Convert the text to lower case
# Remove numbers
# Remove common english stopwords
# Remove all the punctuations
```{r}
titledocs <- tm_map(titledocs, content_transformer(tolower))
titledocs <- tm_map(titledocs, removeNumbers)
titledocs <- tm_map(titledocs, removeWords, stopwords("en"))
titledocs <- tm_map(titledocs, removeWords, c("video","day","full","one","-","make","john","audio"))
titledocs <- tm_map(titledocs, removePunctuation)
```
```

```
# Build a term-document matrix
```{r}
dtm <- TermDocumentMatrix(titledocs)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 30)
```
```

	word <fctr>	freq <dbl>
official	official	139
live	live	54
trailer	trailer	53
new	new	49
music	music	46
christmas	christmas	40
commercial	commercial	38
first	first	36
super	super	35
bowl	bowl	31

1–10 of 30 rows Previous 1 2 3 Next

	word <fctr>	freq <dbl>
star	star	28
show	show	27
world	world	26
last	last	23
feat	feat	23
life	life	23
lyric	lyric	22
awards	awards	22
big	big	21
makeup	makeup	20

11–20 of 30 rows Previous 1 2 3 Next

	word <fctr>	freq <dbl>
–	–	20
wars	wars	19
time	time	18
man	man	18
home	home	17
movie	movie	17
face	face	17
best	best	17
game	game	16
american	american	16

21-30 of 30 rows Previous 1 2 3 Next

```
# Generate the Word cloud for key words of trending video title
```{r}
wordcloud(words = d$word, freq = d$freq, min.freq = 1,
          max.words=200, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```
```



```
# Use similar processes to generate a wordcloud to extract the keywords of trending video channels
```{r}
channeltext <- usvideos$channel_title
channeldocs <- Corpus(VectorSource(channeltext))

channeldocs <- tm_map(channeldocs, content_transformer(tolower))
channeldocs <- tm_map(channeldocs, removeNumbers)
channeldocs <- tm_map(channeldocs, removeWords, stopwords("en"))
channeldocs <- tm_map(channeldocs, removeWords,
c("channel","chris","david","john","mike","tom","late","abc","ryan","johnson","ben","kevin","taylor"))
channeldocs <- tm_map(channeldocs, removePunctuation)

dtm <- TermDocumentMatrix(channeldocs)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 30)
```
```

	word <fctr>	freq <dbl>
news	news	31
cbs	cbs	11
show	show	10
sports	sports	10
world	world	8
records	records	8
entertainment	entertainment	8
pictures	pictures	8
new	new	8
morning	morning	7

1–10 of 30 rows Previous 1 2 3 Next

	word <fctr>	freq <dbl>
bbc	bbc	7
network	network	7
fox	fox	6
insider	insider	6
productions	productions	6
daily	daily	6
science	science	6
live	live	5
king	king	5
nba	nba	5

11–20 of 30 rows Previous 1 2 3 Next

	word <fctr>	freq <dbl>
music	music	5
commercials	commercials	5
films	films	5
max	max	5
life	life	4
awesome	awesome	4
youtube	youtube	4
park	park	4
official	official	4
red	red	4

21–30 of 30 rows

Previous 1 2 3 Next

```
# Generate the Word cloud for key words of trending video channel
```{r}
wordcloud(words = d$word, freq = d$freq, min.freq = 1,
          max.words=200, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```
```



```
# Use similar processes to generate a wordcloud to extract the keywords of trending video tags
```{r}
tagstext <- usvideos$tags
tagsdocs <- Corpus(VectorSource(tagstext))

tagsdocs <- tm_map(tagsdocs, content_transformer(tolower))
tagsdocs <- tm_map(tagsdocs, removeNumbers)
tagsdocs <- tm_map(tagsdocs, removeWords, stopwords("en"))
tagsdocs <- tm_map(tagsdocs, removeWords, c("none","make","video","like","things","box","one","get"))
tagsdocs <- tm_map(tagsdocs, removePunctuation)

dtm <- TermDocumentMatrix(tagsdocs)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 30)
```
```

	word <fctr>	freq <dbl>
bowl	bowl	82
super	super	71
music	music	68
new	new	59
iphone	iphone	49
life	life	46
best	best	37
wars	wars	37
last	last	35
makeup	makeup	33

1–10 of 30 rows Previous 1 2 3 Next

	word <fctr>	freq <dbl>
official	official	32
game	game	31
show	show	30
world	world	30
christmas	christmas	28
league	league	27
live	live	26
friday	friday	26
love	love	25
top	top	25

11–20 of 30 rows Previous 1 2 3 Next

	word <fctr>	freq <dbl>
star	star	25
perfect	perfect	24
red	red	22
school	school	21
perry	perry	21
ice	ice	20
first	first	20
home	home	20
commercial	commercial	20
black	black	18

21–30 of 30 rows Previous 1 2 3 Next

```
# Generate the Word cloud for key words of trending video tags
```{r}
wordcloud(words = d$word, freq = d$freq, min.freq = 1,
          max.words=200, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```
```

```
# Use similar processes to generate a wordcloud to extract the keywords of trending video descriptions
```{r}
destext <- usvideos$description
desdocs <- Corpus(VectorSource(destext))

desdocs <- tm_map(desdocs, content_transformer(tolower))
desdocs <- tm_map(desdocs, removeNumbers)
desdocs <- tm_map(desdocs, removeWords, stopwords("en"))
desdocs <- tm_map(desdocs, removeWords,
c("video","one","get","channels","videos","like","will","can","make","just","see","use","know","watch","channel","su
bscribe","time","▶","every","day","way","available","made","'s","also","want","people","full","follow","check","thi
gs","please","-","links","find","got","year","used","thanks","years","twitter","facebook","instagram","youtube","▶"
"'s","back","-","website","camera","big","little","series"))
desdocs <- tm_map(desdocs, removePunctuation)

dtm <- TermDocumentMatrix(desdocs)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 30)
```
```

	word <fctr>	freq <dbl>
new	new	450
music	music	365
now	now	254
world	world	230
news	news	221
live	live	190
first	first	186
love	love	183
show	show	180
best	best	155

1–10 of 30 rows Previous 1 2 3 Next

	word <fctr>	freq <dbl>
▶	▶	150
life	life	150
official	official	144
's	's	134
album	album	130
free	free	109
today	today	108
-	-	106
latest	latest	103
social	social	102

11–20 of 30 rows Previous 1 2 3 Next

	word <fctr>	freq <dbl>
code	code	98
christmas	christmas	96
song	song	94
film	film	94
play	play	94
last	last	92
story	story	92
think	think	90
media	media	86
right	right	86

21–30 of 30 rows Previous 1 2 3 Next

```
# Generate the Word cloud for key words of trending video descriptions
```{r}
wordcloud(words = d$word, freq = d$freq, min.freq = 1,
          max.words=200, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```
```

```
#Segmentation(Cluster Analytsis)
library(ggplot2)

#Hierarchical Cluster Analysis and K-means Clustering of the likes amounts
usvideos1 = data.frame(x1=usvideos$likes,x2=usvideos$likes)
distances = dist(usvideos1,method = "euclidean")
clust = hclust(distances,method = "ward.D2")
clusters = cutree(clust,k = 2)
usvideos2 = cbind(usvideos1,clusters)

set.seed(100)
km = kmeans(usvideos2, centers = 4,iter.max = 1000,nstart = 100)
km
km$centers
km$cluster

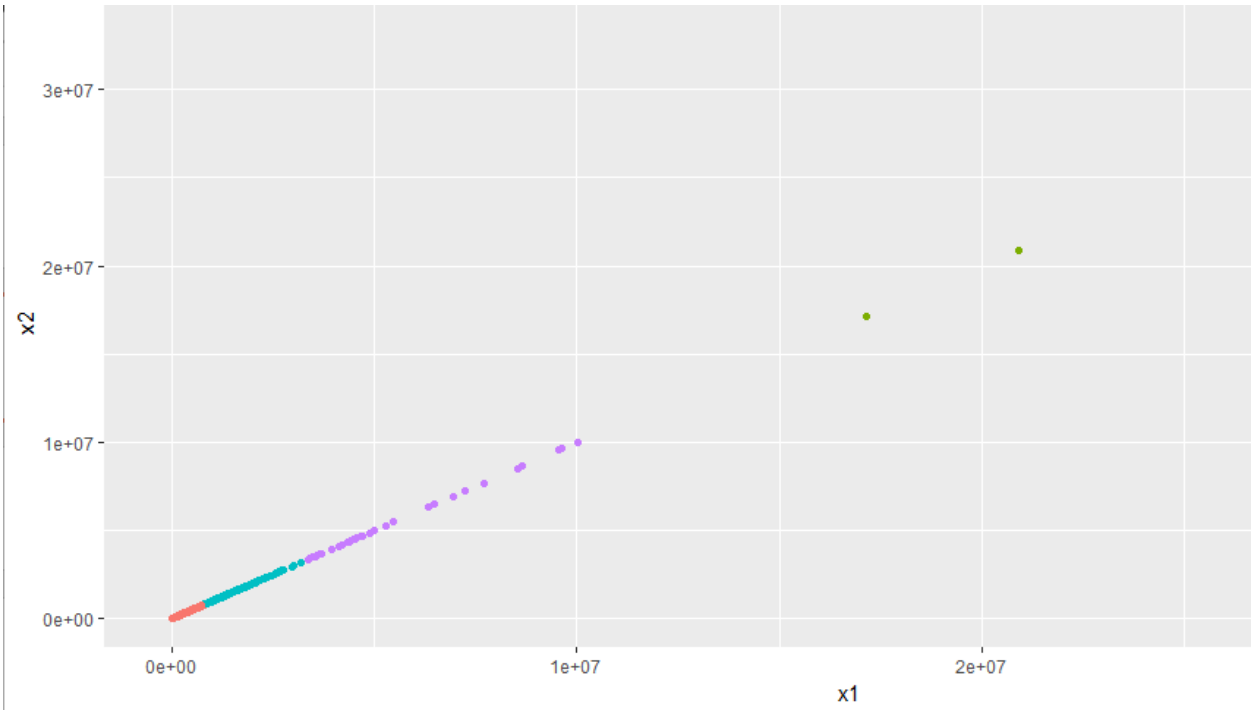
usvideos3 = cbind(usvideos2,kmc=km$cluster)
ggplot(data=usvideos3,aes(x=x1,y=x2,color=factor(kmc)))+
  geom_point()
```



```
#Hierarchical Cluster Analysis and K-means Clustering of the views amounts
usvideos4 = data.frame(x1=usvideos$views,x2=usvideos$views)
distances = dist(usvideos4,method = "euclidean")
clust = hclust(distances,method = "ward.D2")
clusters = cutree(clust,k = 2)
usvideos5 = cbind(usvideos4,clusters)

set.seed(100)
km = kmeans(usvideos5, centers = 4,iter.max = 1000,nstart = 100)
km
km$centers
km$cluster

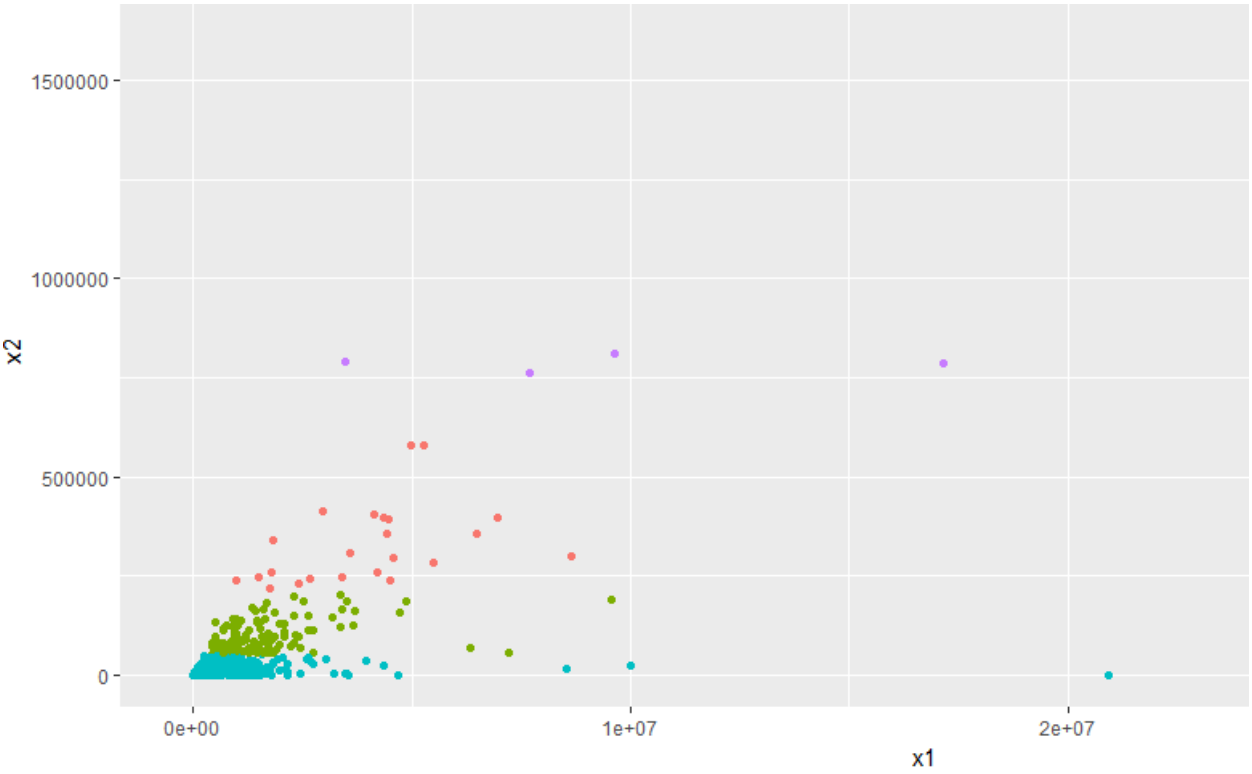
usvideos6 = cbind(usvideos5,kmc=km$cluster)
ggplot(data=usvideos6,aes(x=x1,y=x2,color=factor(kmc)))+
  geom_point()
```



```
#Hierarchical Cluster Analysis and K-means Clustering of the likes&views amounts
usvideos7 = data.frame(x1=usvideos$views,x2=usvideos$likes)
distances = dist(usvideos7,method = "euclidean")
clust = hclust(distances,method = "ward.D2")
clusters = cutree(clust,k = 2)
usvideos8 = cbind(usvideos7,clusters)

set.seed(100)
km = kmeans(usvideos2, centers = 4,iter.max = 1000,nstart = 100)
km
km$centers
km$cluster

usvideos8 = cbind(usvideos7,kmc=km$cluster)
ggplot(data=usvideos8,aes(x=x1,y=x2,color=factor(kmc)))+
  geom_point()
```



Appendix B - Reference

1. Brian D. YouTube SEO: How to Rank YouTube Videos in 2018. Backlinko. https://backlinko.com/how-to-rank-youtube-videos. Published February 20, 2018. Retrieved April 16, 2018.

2. Chris A. Understanding the life of video metadata from production to publishing & why it's so important. http://tubularinsights.com/video-metadata-production-publishing/. Published June 12, 2012. Retrieved April 17,2018.

3. Mushroom Networks. YouTube - The Second Largest Search Engine. https://www.mushroomnetworks.com/infographics/youtube---the-2nd-largest-search-engine-infographic/. Retrieved April 17,2018.
4. Jim Bartlett. Accelerating Results for Growing Business. http://www.pinnicle.com/Articles/Pareto_Principle/pareto_principle.html/. Retrieved April 26,2018.