Youtube Trending Video

Carol

4/4/2020

1. Project Introdcution

YouTube is the most popular video platform with the largest user base. One of the most important reasons for its popularity is that YouTube allows everyone to be a blogger. It enables content creators to share their content with a large audience. As a result, there's a wide range of content to choose from. No matter what you want, such as makeup tutorials, cooking guides, product reviews or travel vlogs, you can always find them on YouTube.

However, with millions of contents published every day, it's super hard for a video to stand out. Our project aims at figuring out what factors make a YouTube video a hit. Here, we use Python and Hadoop to analyze the Trending YouTube Video dataset, which includes the data on daily trending YouTube videos from 10 regions, with up to 200 listed trending videos per day.

The main goal of this project is to get insights into trending videos, to find out what's the common features. Moreover, YouTube has a wide range of users who are from all over the world. People from different areas have different tastes on videos, so we analyzed user preference based on regions. Knowing that which category is their favorite, we could deliver exactly what they want. Hopefully, these insights will be helpful to the content creators who want to increase the popularity of their videos.

2. Description of the data

What is YouTube and trending video? YouTube has a sophisticated algorithm to select 200 "trending video" of everyday. "trending video" is not simply the top 200 most-viewed videos, but those videos selected by a combination of factors including measuring users interactions, such as number of views, shares, comments and likes/dislikes.

About data source

This dataset is an opensource dataset from Kaggle, link: https://www.kaggle.com/datasnaek/youtube-new (https://www.kaggle.com/datasnaek/youtube-new) The data is collected by Kaggle user Mitchell J using the YouTube API in June 2019.

3. Dataset Description

Loading libraries

```
set.seed(1)
# Data manipulation
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.6.2
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(DT)
library(stringi)

# Time manipulation,ymd function
library(lubridate)
```

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

```
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday,
##     week, yday, year
```

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

```r
# Visualization
library(knitr)
library(ggplot2)
library(plotrix)
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.6.3
```

```
## corrplot 0.84 loaded
```

```
library(ggrepel)
library(gbm)
```

```
## Loaded gbm 2.1.5
```

```
# Extract data from json
library(rjson)
library(jsonlite)
```

```
##
## Attaching package: 'jsonlite'
```

```
## The following objects are masked from 'package:rjson':
##
##     fromJSON, toJSON
```

Reading and preparing data

```
setwd("C:/Users/xiaol/OneDrive/Documents/GitHub/brisk-guacamole")

#Load GB data
gb_r = read.csv("D:/2020 Coding Projects/youtube_trending_videos/youtube data/GBvideos.csv")
gb_category = fromJSON("D:/2020 Coding Projects/youtube_trending_videos/youtube data/GB_category
_id.json")

category_id = gb_category[["items"]][["id"]]
category = gb_category[["items"]][["snippet"]][["title"]]
gb_category = data.frame(category_id , category)
gb = merge(gb_r,gb_category)
gb$location = "GB"

dim(gb)
```

```
## [1] 38826    18
```

```
# Load CA data
ca_r = read.csv("D:/2020 Coding Projects/youtube_trending_videos/youtube data/CAvideos.csv")
ca_category = fromJSON("D:/2020 Coding Projects/youtube_trending_videos/youtube data/CA_category
_id.json")

category_id = ca_category[["items"]][["id"]]
category = ca_category[["items"]][["snippet"]][["title"]]
ca_category = data.frame(category_id , category)
ca = merge(ca_r,ca_category)
ca$location = "CA"

dim(ca)
```

```
## [1] 40807    18
```

```
#Load US data
us_r = read.csv("D:/2020 Coding Projects/youtube_trending_videos/youtube data/USvideos.csv")
us_category = fromJSON("D:/2020 Coding Projects/youtube_trending_videos/youtube data/US_category
_id.json")

category_id = us_category[["items"]][["id"]]
category = us_category[["items"]][["snippet"]][["title"]]
us_category = data.frame(category_id , category)
us = merge(us_r,us_category)
us$location = "US"

dim(us)
```

```
## [1] 40949    18
```

```
#Load FR data
fr_r = read.csv("D:/2020 Coding Projects/youtube_trending_videos/youtube data/FRvideos.csv")
fr_category = fromJSON("D:/2020 Coding Projects/youtube_trending_videos/youtube data/FR_category
_id.json")

category_id = fr_category[["items"]][["id"]]
category = fr_category[["items"]][["snippet"]][["title"]]
fr_category = data.frame(category_id , category)
fr = merge(fr_r,fr_category)
fr$location = "FR"

dim(fr)
```

```
## [1] 40610    18
```

```
#Load DE data
de_r = read.csv("D:/2020 Coding Projects/youtube_trending_videos/youtube data/DEvideos.csv")
de_category = fromJSON("D:/2020 Coding Projects/youtube_trending_videos/youtube data/DE_category
_id.json")

category_id = de_category[["items"]][["id"]]
category = de_category[["items"]][["snippet"]][["title"]]
de_category = data.frame(category_id , category)
de = merge(de_r,de_category)
de$location = "DE"

dim(de)
```

```
## [1] 40584    18
```

```
#Merge data from 5 countries into videos dataset
videos <- as.data.table(rbind(gb,ca,us,fr,de))

dim(videos)
```

```
## [1] 201776     18
```

```
#convert trending_date to datetime
videos$trending_date <- ydm(videos$trending_date)

#extract publish hour
videos$publish_time_hour = hour(hms(substr(videos$publish_time,start =12, stop=19)))

#extract video publish time: day of week 1=Sunday, 2=Monday
videos$publish_time_dayofweek = wday(videos$publish_time)
head(videos$publish_time_dayofweek)
```

```
## [1] 6 7 5 7 1 4
```

```
#extract publish date
videos$publish_time <- ymd(substr(videos$publish_time,start = 1,stop = 10))

#Calculate the time difference between publish time and trending date
videos$dif_days <- videos$trending_date-videos$publish_time

#Let's see the new dataset
dim(videos)
```

```
## [1] 201776     21
```

```
names(videos)
```

```
##  [1] "category_id"            "video_id"              
##  [3] "trending_date"          "title"                 
##  [5] "channel_title"          "publish_time"          
##  [7] "tags"                   "views"                 
##  [9] "likes"                  "dislikes"              
## [11] "comment_count"          "thumbnail_link"        
## [13] "comments_disabled"      "ratings_disabled"      
## [15] "video_error_or_removed" "description"           
## [17] "category"               "location"              
## [19] "publish_time_hour"      "publish_time_dayofweek"
## [21] "dif_days"
```

Each dataset collected trending videos over 205 days, from 2017/11/14 to 2018/06/14. Average views: 2055209, average likes: 56699, average dislikes: 3011 and average comments: 6112.

Part 1- Global trending videos Overview

First let’s see some number of global trending videos.

Top 10 videos in absolute value

Viewed videos Liked videos Disliked videos Commented videos

```
mviews = videos[,.("Total_Views"=round(max(views,na.rm = T),digits = 2)),by=.(title,thumbnail_li
nk)][order(-Total_Views)]
mviews %>%
  mutate(image = paste0('<img width="80%" height="80%" src="', thumbnail_link , '"></img>')) %>%
  arrange(desc(Total_Views)) %>%
  head(10)%>%
  select(image,title,Total_Views) %>%
  datatable(class = "nowrap hover row-border", escape = FALSE, options = list(dom = 't',scrollX
 = TRUE, autoWidth = TRUE))
```

```
## Warning in instance$preRenderHook(instance): It seems your data is too
## big for client-side DataTables. You may consider server-side processing:
## https://rstudio.github.io/DT/server.html
```

	image	title
1		Nicky Jam x J. Balvin - X (EQUIS) | Video Oficial | Prod. Afro Bros & Jeon
2		Te Bote Remix - Casper, Nio Garc??a, Darell, Nicky Jam, Bad Bunny, Ozuna | Video Oficial
3		Bad Bunny - Amorfoda | Video Oficial
4		Ozuna x Romeo Santos - El Farsante Remix
5		Childish Gambino - This Is America (Official Video)
6		Drake - God?€?s Plan
7		Ariana Grande - No Tears Left To Cry
8		Becky G, Natti Natasha - Sin Pijama (Official Video)

9		YouTube Rewind: The Shape of 2017 | #YouTubeRewind
10		Dura - Daddy Yankee (Video Oficial)

Top 10 videos in percentage

The absolute number of likes, dislikes and comments didn't show the whole picture of video influence. Here I use percentage to further discover the trending data. To guarantee the validity of result, I filter the videos with at least 75626 views (1st quartile).

Top 10 videos in percentage

% Liked videos % Disliked videos % Commented videos

```
mlikes_percent <- videos[,.("Views" = views, "Percentage_Likes"=round(100*max(likes,na.rm = T)/m
ax(views,na.rm = T),digits = 2)),by=.(title,thumbnail_link)][order(-Percentage_Likes)]
mlikes_percent %>%
  mutate(image = paste0('<img width="80%" height="80%" src="', thumbnail_link , '"></img>')) %>%
  filter(Views>=75626)%>%
  arrange(-Percentage_Likes) %>%
  head(10)%>%
  select(image, title, Percentage_Likes)%>%
  datatable(class = "nowrap hover row-border", escape = FALSE, options = list(dom = 't',scrollX
 = TRUE, autoWidth = TRUE))
```

```
## Warning in instance$preRenderHook(instance): It seems your data is too
## big for client-side DataTables. You may consider server-side processing:
## https://rstudio.github.io/DT/server.html
```

	image	title	Percenta
1		PLANETE FOOT | LA COUPE DE KURZAWA... !	
2		PLANETE FOOT | LA COUPE DE KURZAWA... !	
3		Gott w??rfelt nicht nur einmal! - [Full Movie 4k]	
4		The Reaction of The Streets (I Wait-Day6 Edition)	

5		?????€?? ???????? - ?? ?????????? (DK REMAKE) ?????€???????? ??????
6		Harry & Louis ?€? CRACK!VID #9
7		Harry & Louis ?€? CRACK!VID #9
8		[MIXTAPE] I.M - Fly With Me (Teaser)
9		[MIXTAPE] I.M - Fly With Me (Teaser)
10		Imagine Dragons ?€? THUNDER (PARODIE) - LiDiRo

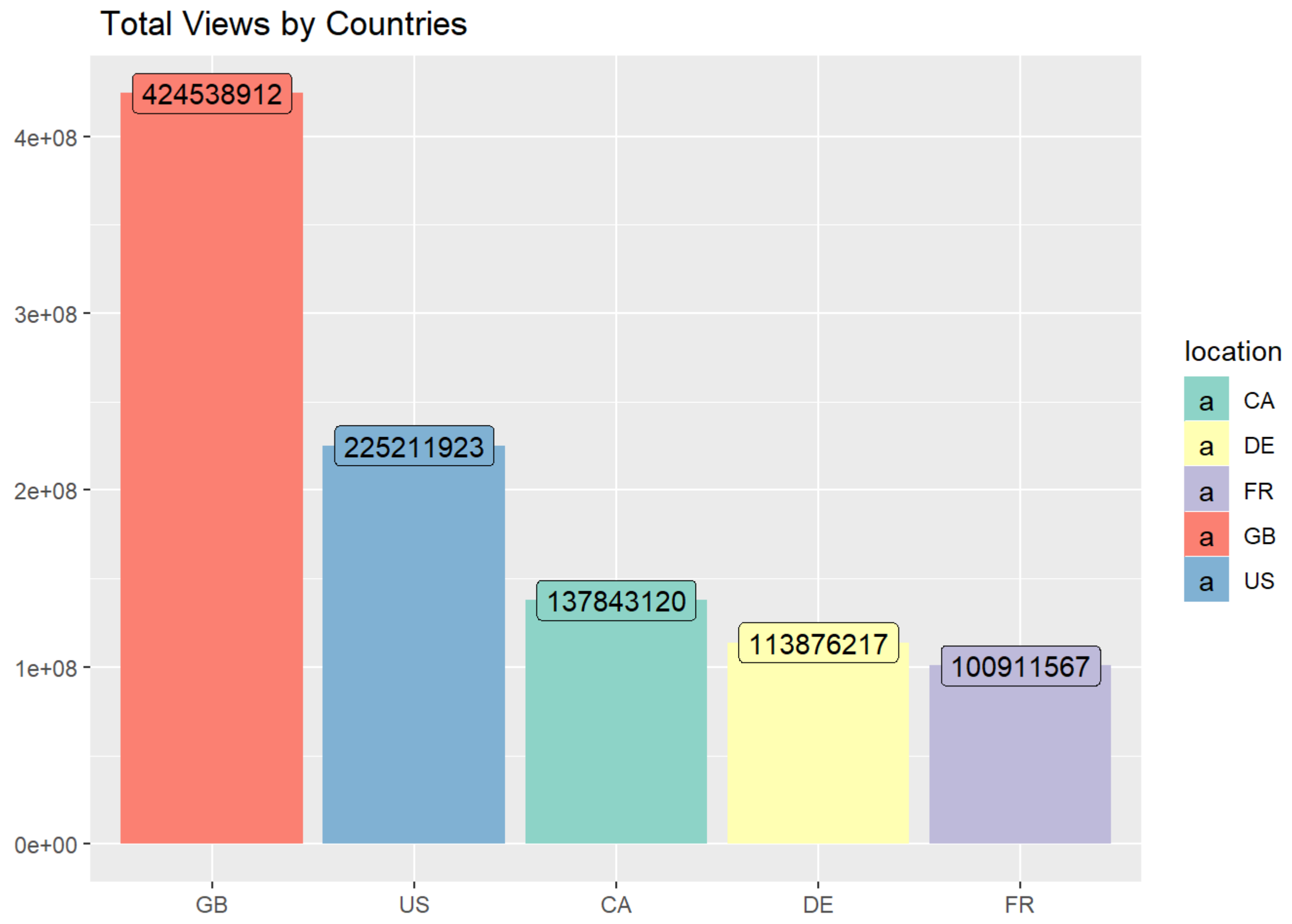
Part 2- Geographical Analysis

This part I go down to country level to discover which country has the most active Youtube users.

Geographical Analysis

Total Views In total number of Interactions Percentage of Interactions

```
country_view = videos[,.("Total_Views"=max(views)),by=location]

ggplot(country_view,aes(reorder(location,-Total_Views),Total_Views,fill=location))+geom_bar(stat="identity")+
geom_label(aes(label=Total_Views))+  labs(title=" Total Views by Countries")+labs(x=NULL, y= NULL)+
scale_fill_brewer(palette = "Set3")
```



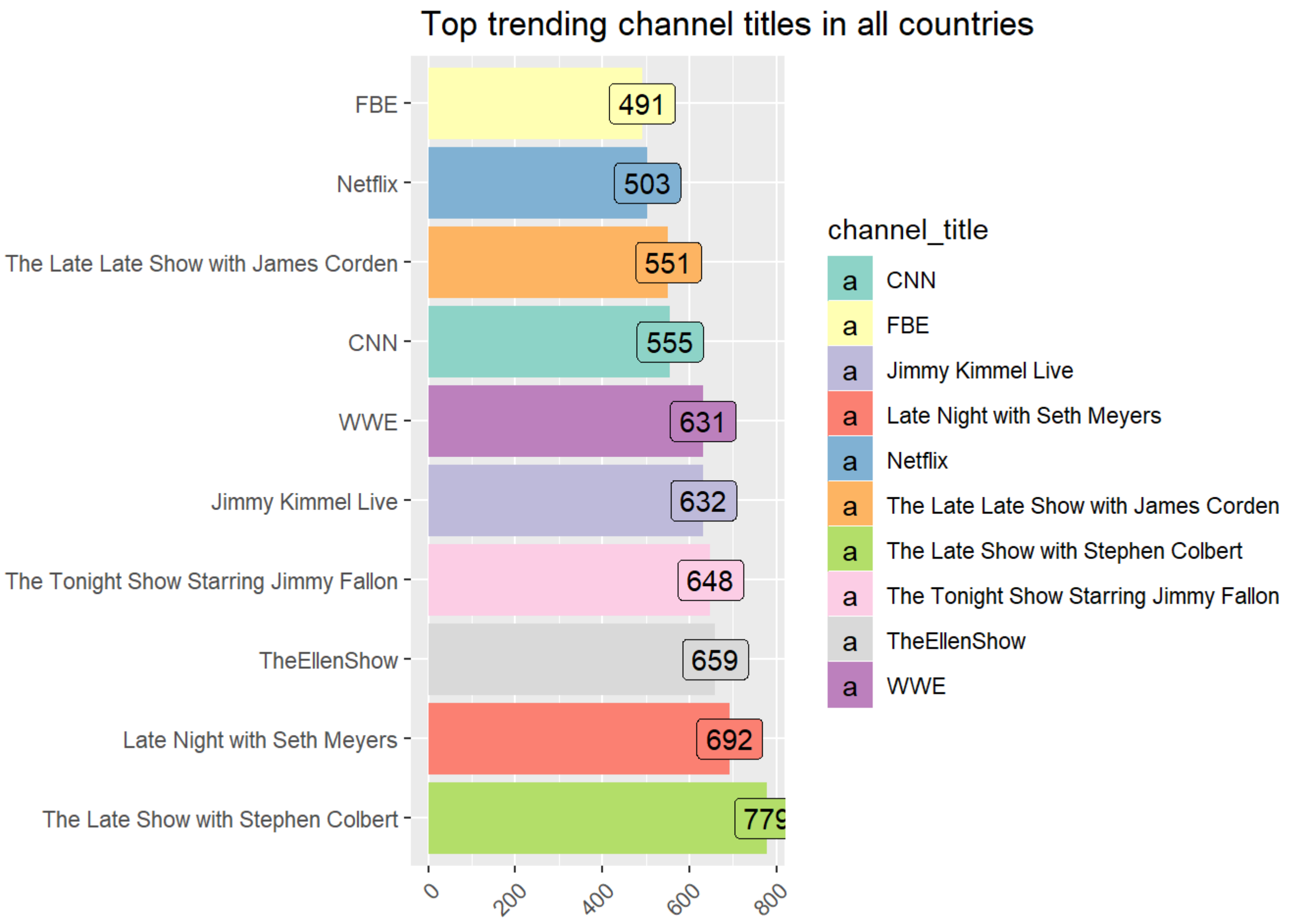
British users have most views on Youtube trending videos, far beyond the second region, United States.

Part-3: Top trending Channels in all countries

Top Channels in each country

Overall US CA GB

```
top_channel = videos[,.N,by=channel_title][order(-N)][1:10]

ggplot(top_channel,aes(reorder(channel_title,-N),N,fill=channel_title))+geom_bar(stat="identity"
)+
geom_label(aes(label=N))+theme(axis.text.x = element_text(angle = 45,hjust = 1))+
labs(title=" Top trending channel titles in all countries")+labs(x = NULL, y= NULL)+coord_flip()
+
scale_fill_brewer(palette = "Set3")
```



Most popular trending channels are US talk shows, like the late show with stephen colbert,and Late Night with Seth Meyers. The top trending channels are almost from United States. I took a wild guess that the results may be dominated by US users strong love over talk show.

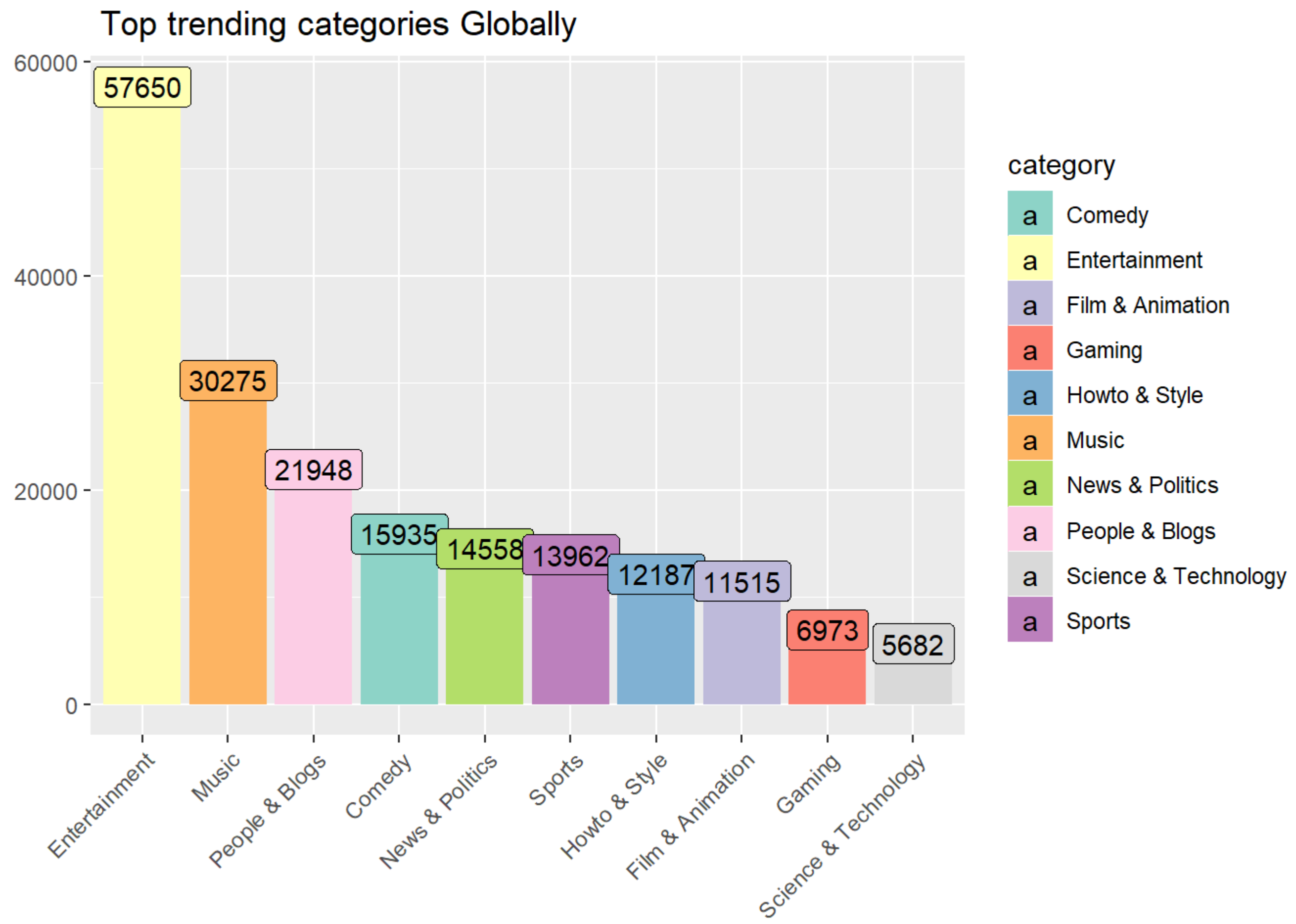
To justify my guess, I look deep into the trending video channels in three English country: US, GB, CA.

Part-4: Top trending Categories in all countries

Top Categories in each country

Global US CA GB FR DE

```
top_category = videos[,.N,by=category][order(-N)][1:10]

gl_ca = ggplot(top_category,aes(reorder(category,-N),N,fill=category))+geom_bar(stat="identity")
+
geom_label(aes(label=N))+theme(axis.text.x = element_text(angle = 45,hjust = 1))+
labs(title=" Top trending categories Globally")+labs(x = NULL, y= NULL)+scale_fill_brewer(palett
e = "Set3")
gl_ca
```

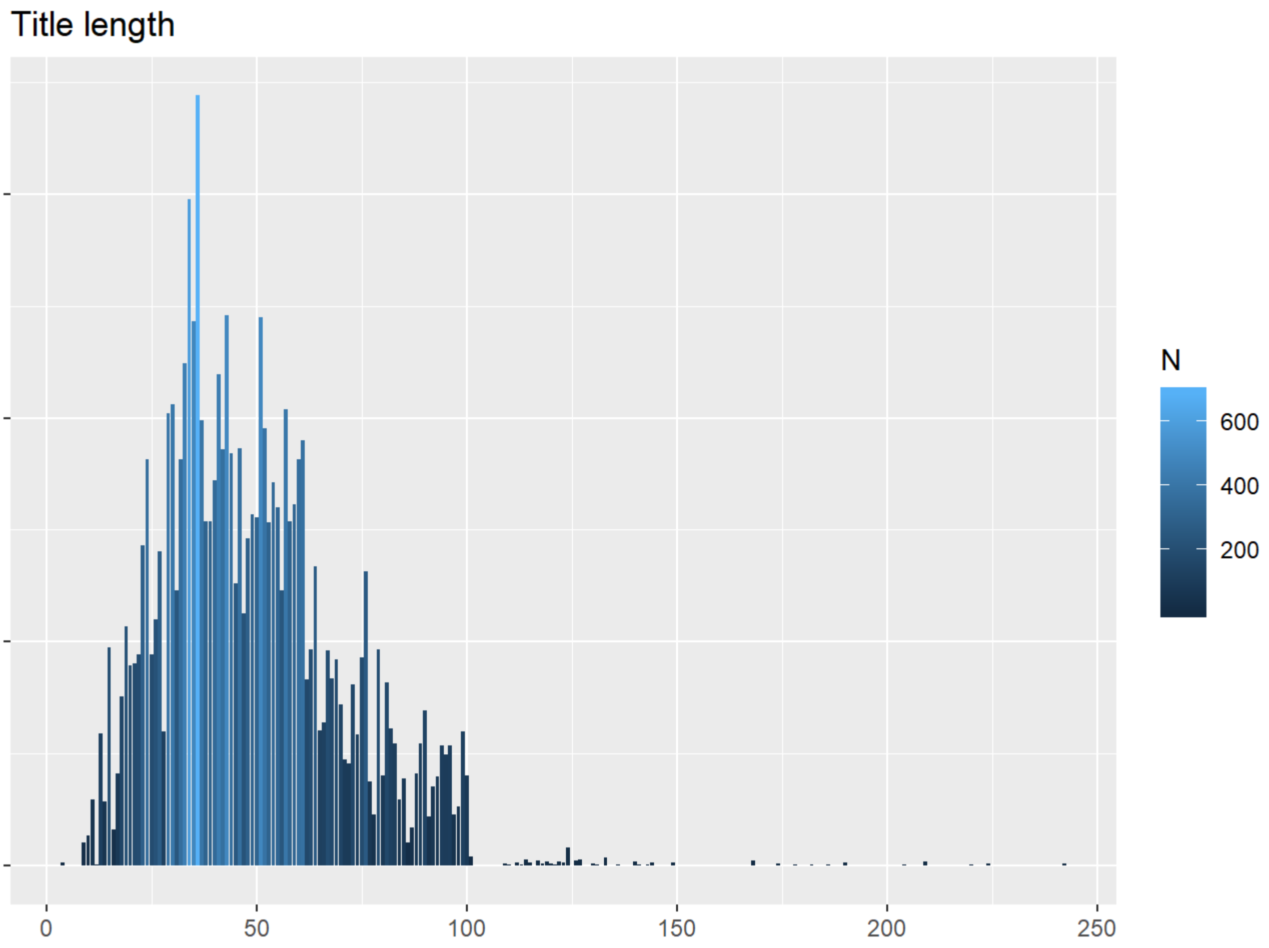


Part-5: Title length

Does video title length matters?

To answer this question, I did the hypothesis testing among the top 20000 viewed videos and the bottom 20000 viewed videos.

```
# Most likes video string length
v_best_title = videos[order(-views)][1:20000][,.("Title_Len"= stri_length(title))][,.N,by=Title_
Len][order(-N)]
ggplot(v_best_title,aes(Title_Len,N,fill=N))+geom_bar(stat = "identity")+
labs(title="Title length")+labs(x = NULL, y= NULL)
```



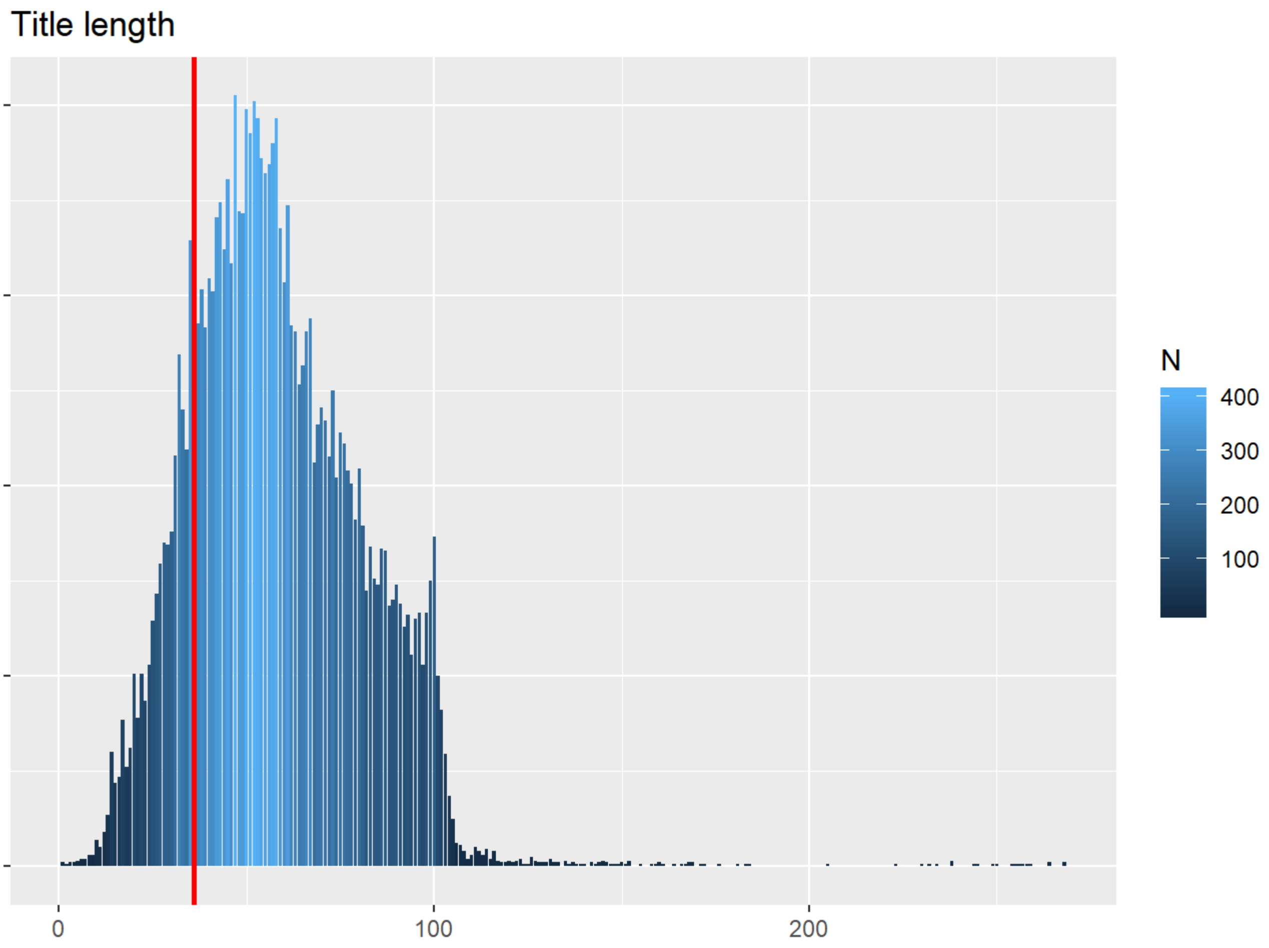
```
v_best_title$Title_Len[1]
```

```
## [1] 36
```

```
v_best_title
```

```
##      Title_Len   N
##   1:        36 688
##   2:        34 595
##   3:        43 492
##   4:        51 490
##   5:        35 486
##  ---
## 126:       143   1
## 127:       113   1
## 128:        12   1
## 129:       136   1
## 130:       110   1
```

```
v_worst_title = videos[order(views)][1:20000][,.("Title_Len"= stri_length(title))][,.N,by=Title_
Len][order(-N)]
ggplot(v_worst_title,aes(Title_Len,N,fill=N))+geom_bar(stat = "identity")+
labs(title="Title length")+labs(x = NULL, y= NULL)+ geom_vline(xintercept = 36, lwd = 1, col='re
d')
```



```
t.test(v_best_title, v_worst_title)
```

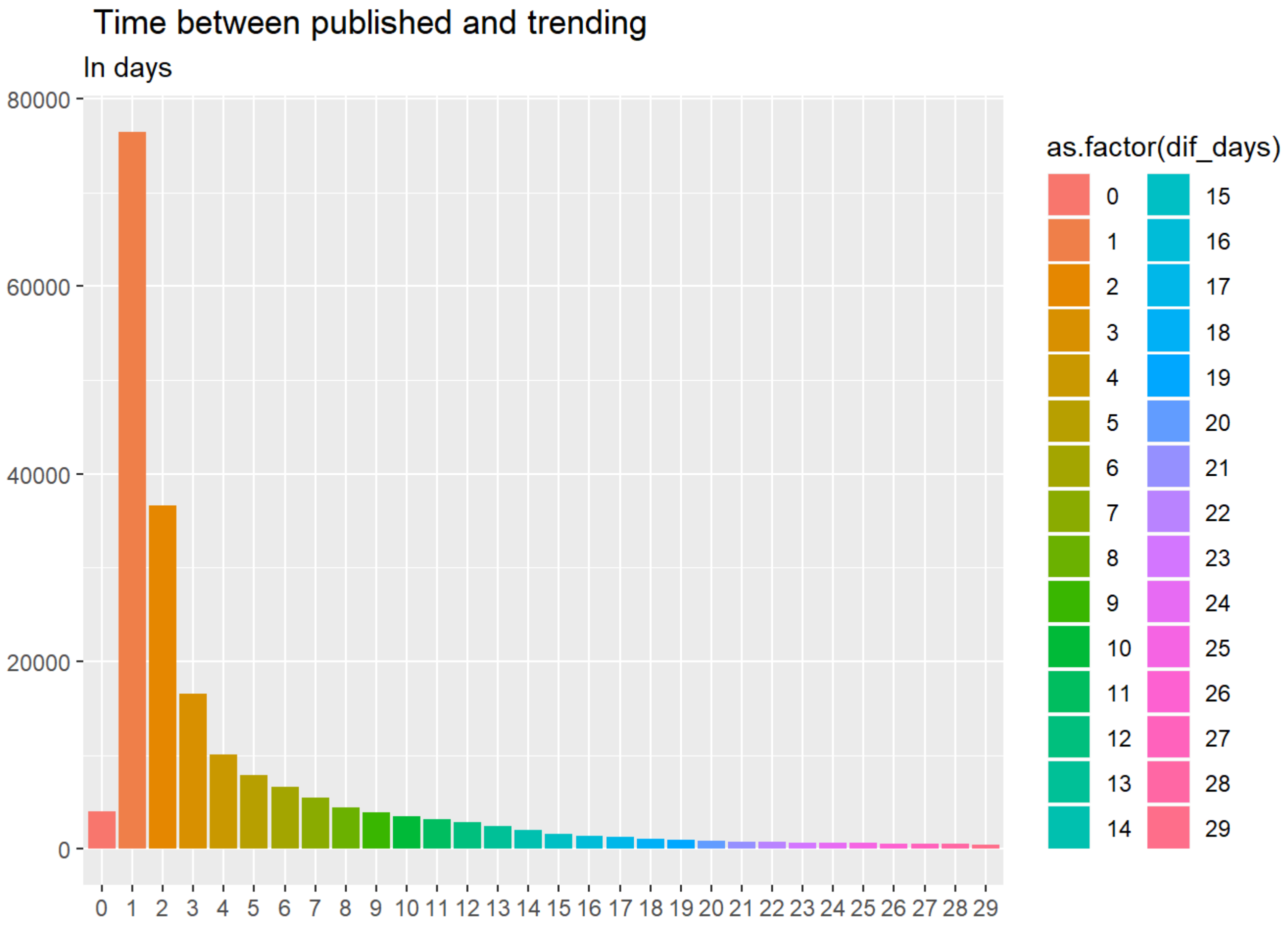
```
## 
##  Welch Two Sample t-test
## 
## data:  v_best_title and v_worst_title
## t = 1.3521, df = 497.84, p-value = 0.177
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -5.671856 30.706639
## sample estimates:
## mean of x mean of y 
##  117.0500  104.5326
```

P value is 0.177(>0.05), which means statistically title length is not significantly different between two groups. However, by observing these two graphs, we could tell that the most popular videos with highest views tend to have a more compact title, with a title length of 33 to 43.

Part-6

How much time passes between published and trending?

```
ggplot(videos[dif_days<30],aes(as.factor(dif_days),fill=as.factor(dif_days)))+geom_bar()+
labs(title=" Time between published and trending",subtitle="In days")+labs(x = NULL, y= NULL)
```

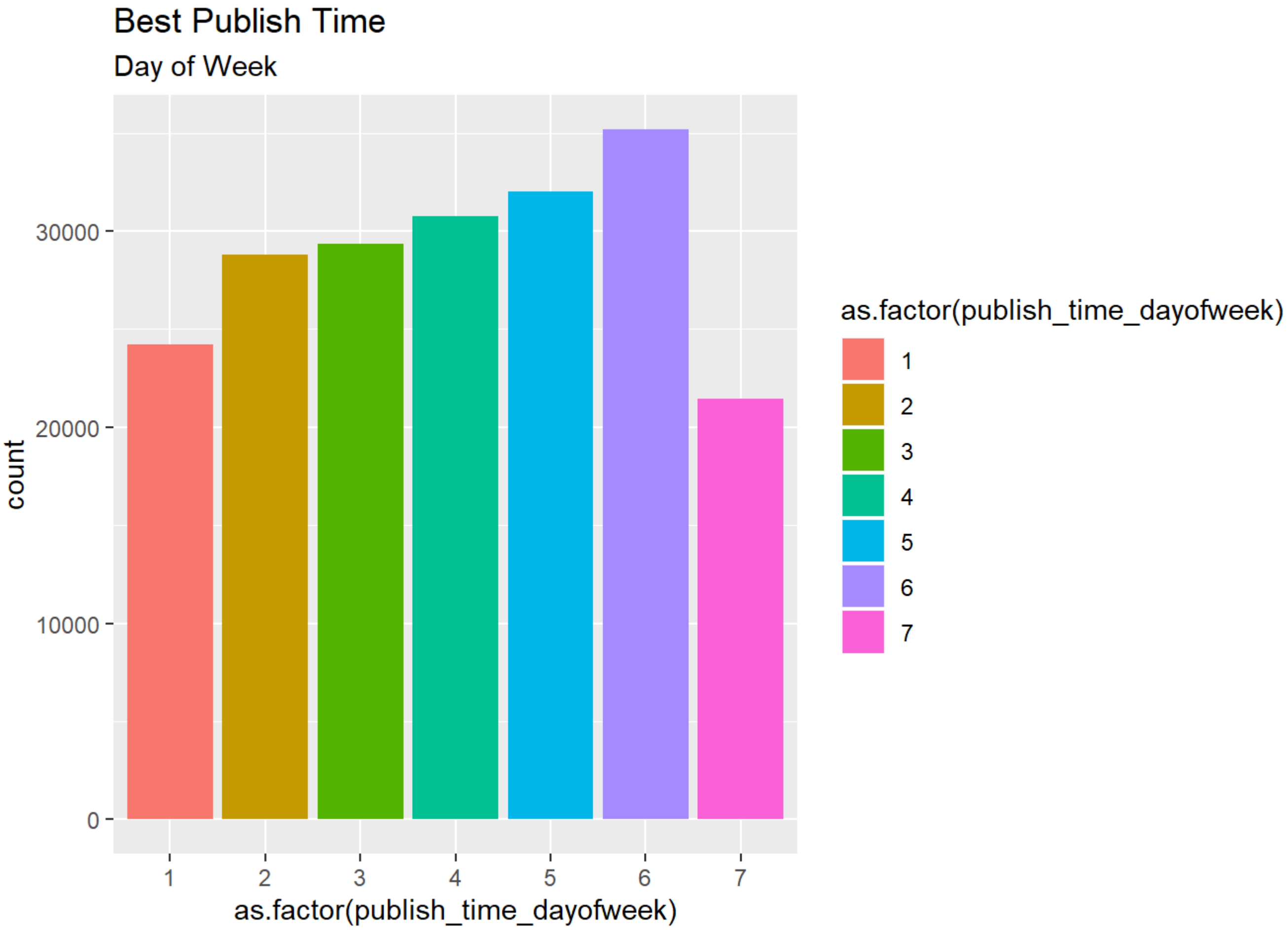


It usually takes 1-3 days between video published and become trending.

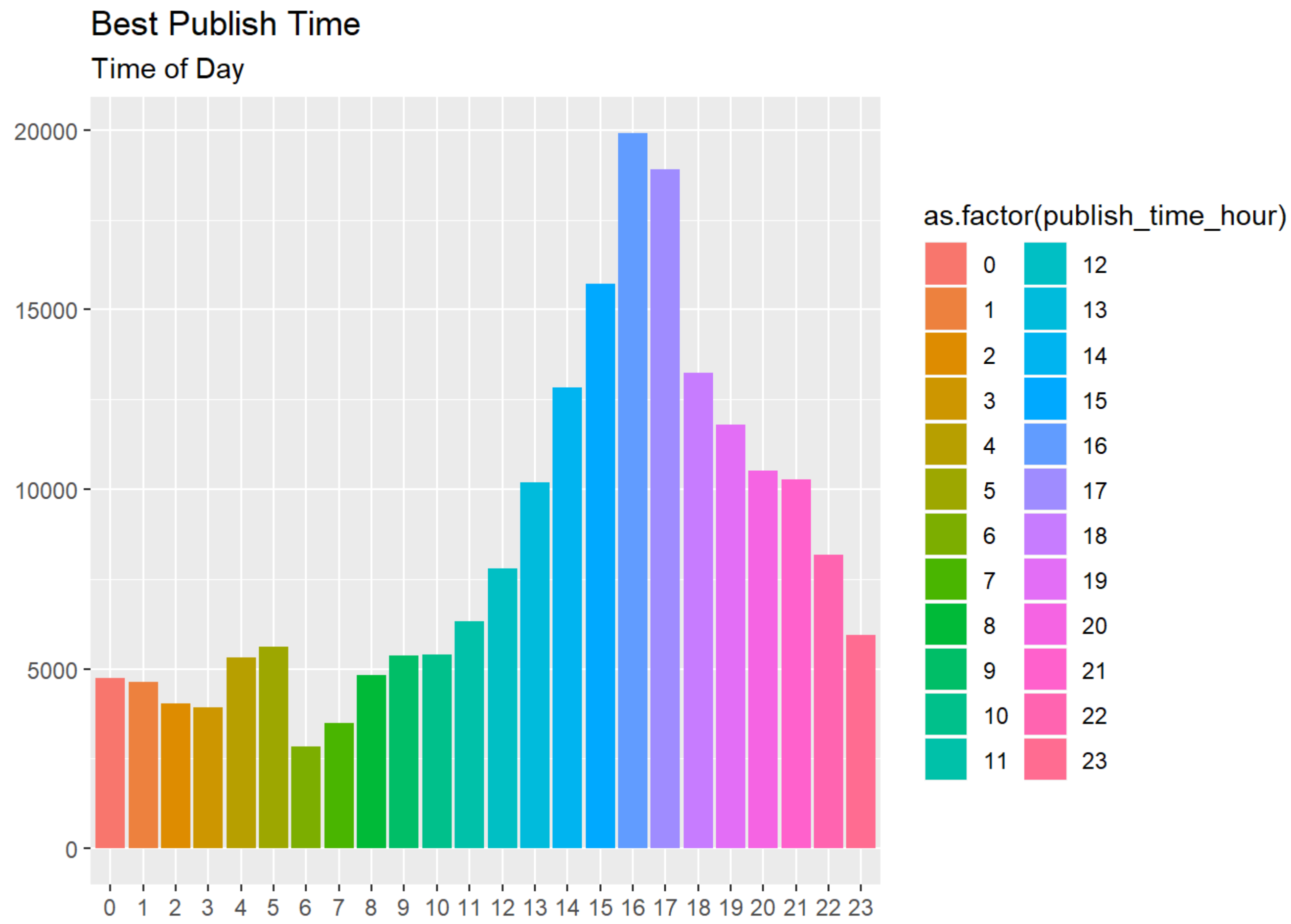
Part-7

What is the best publish time?

```
# Day of Week
ggplot(videos,aes(as.factor(publish_time_dayofweek),fill=as.factor(publish_time_dayofweek)))+geo
m_bar()+
labs(title="Best Publish Time",subtitle="Day of Week")
```



```
# Time of day
ggplot(videos,aes(as.factor(publish_time_hour),fill=as.factor(publish_time_hour)))+geom_bar()+
labs(title="Best Publish Time",subtitle="Time of Day")+labs(x = NULL, y= NULL)
```

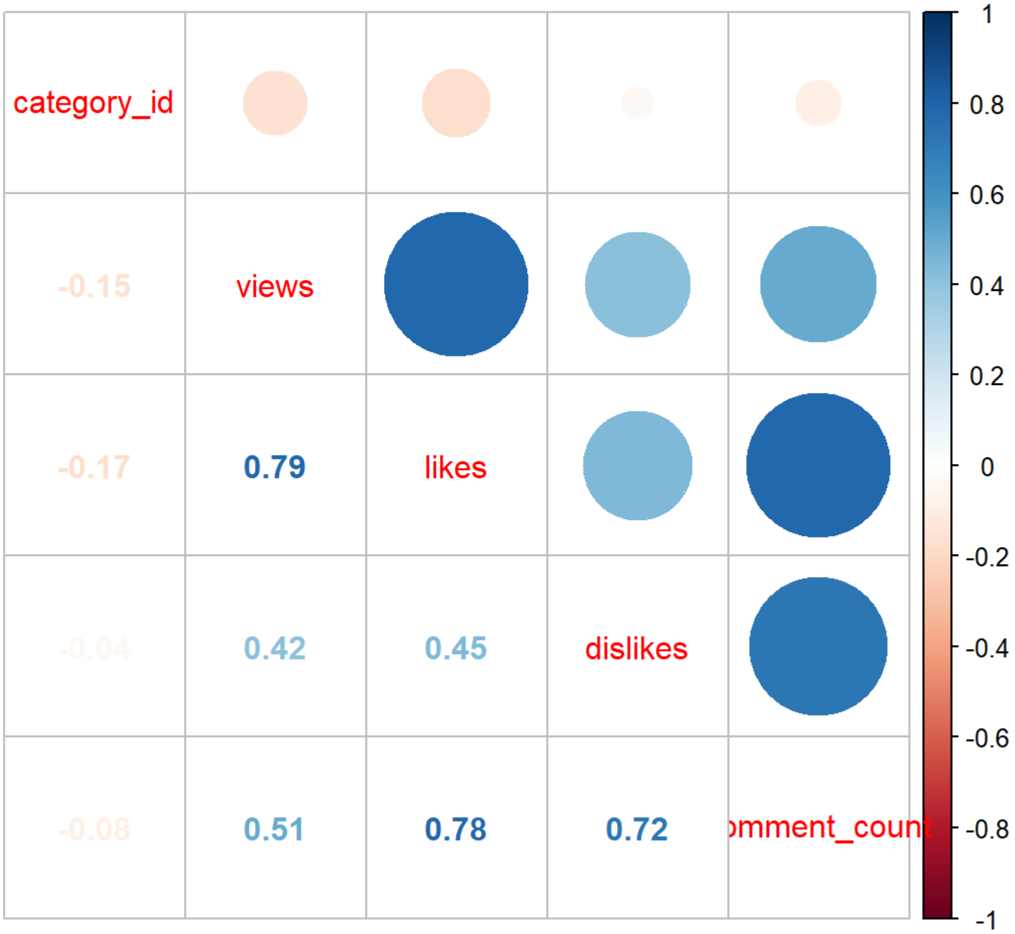


Most trending videos are published at afternoon. 3pm to 5pm is the peak publishing time. Trending videos are most published at Friday. With 1-2 days' time difference between published and trending, I conclude that the videos published at Friday/Thursday are more likely to become trending during weekends, since users spend more time watching Youtube videos during weekends.

Part-8

Finally, let't see the correlation between variables.

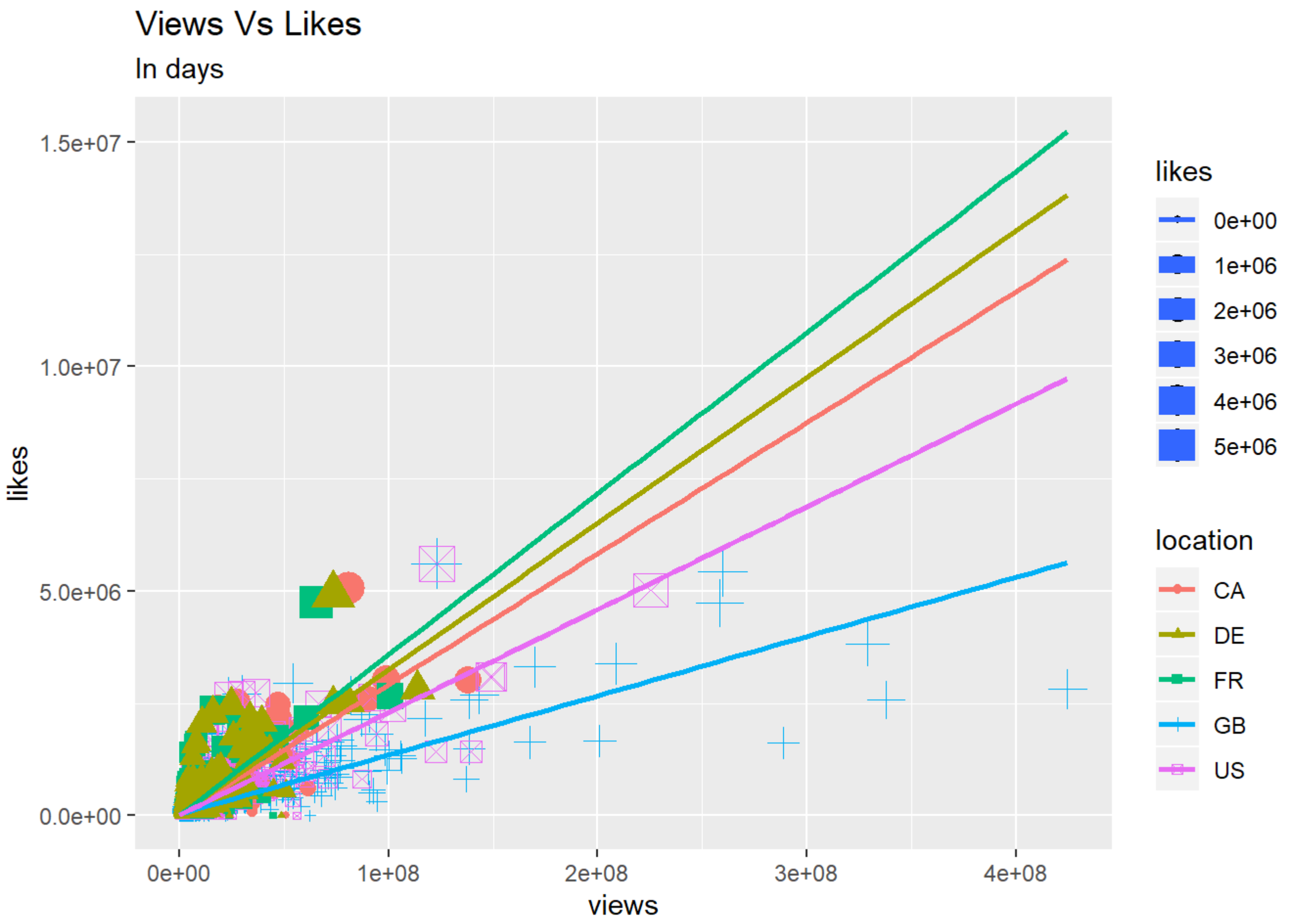
```
corrplot.mixed(corr = cor(videos[,c("category_id","views","likes","dislikes","comment_count"),wi
th=F]))
```



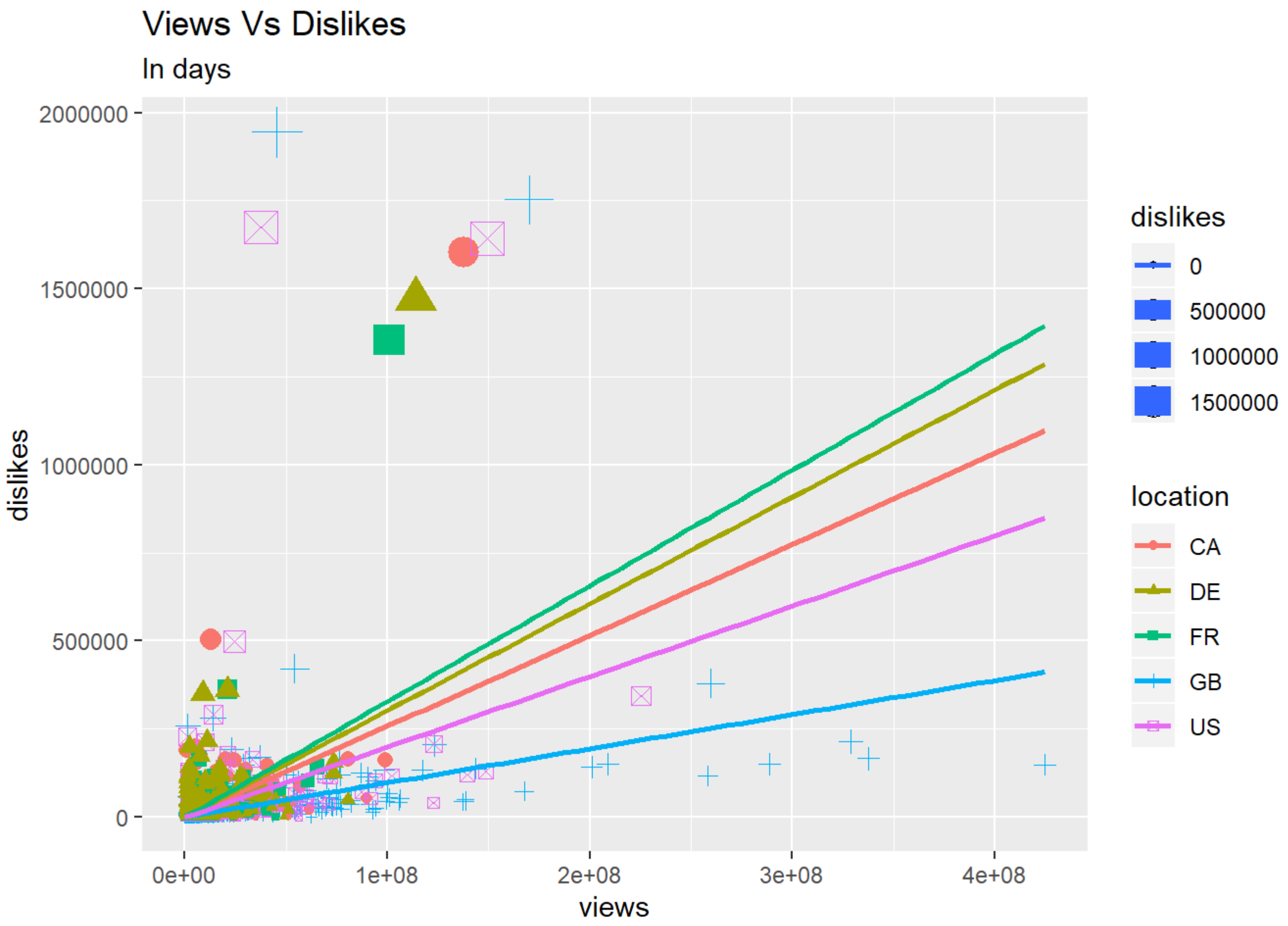
We can see that between views and likes we have a high correlation, I supposed that we will have a similar correlation between views and dislikes, but it turned out to be only half of the like correlation.

Corelation between views and Likes/ Dislikes/ Comments

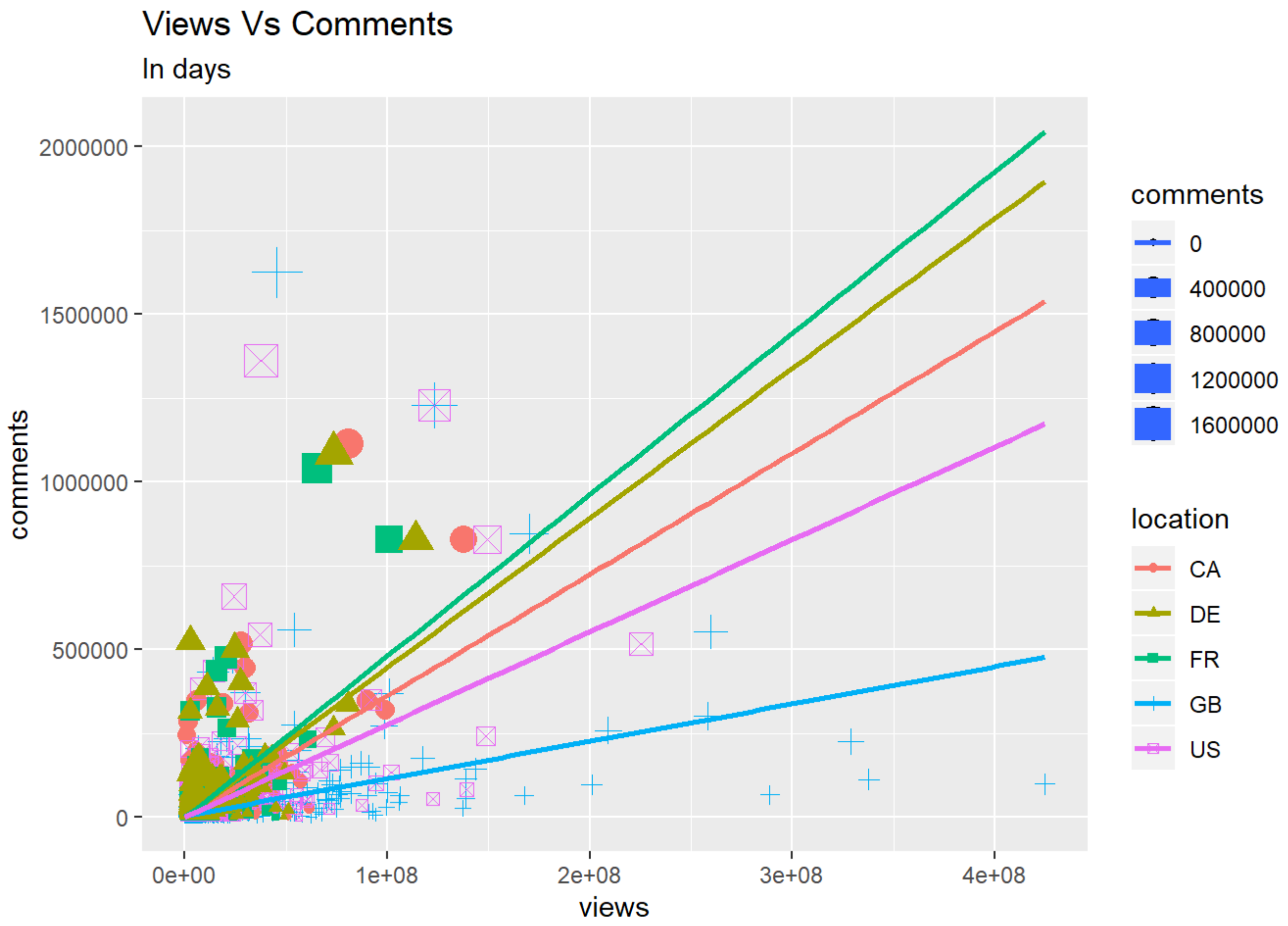
```
# Views vs Likes
view_likes = videos[,.("views"=max(views),"likes"=max(likes)),by=.(title, location)]

ggplot(view_likes,aes(views,likes,colour=location,shape = location,size = likes))+
geom_jitter()+geom_smooth(method=lm,se=FALSE, fullrange=TRUE)+labs(title="Views Vs Likes",subtit
le="In days")
```



```
# Views vs Dislikes
view_dislikes = videos[,.("views"=max(views),"dislikes"=max(dislikes)),by=.(title, location)]

ggplot(view_dislikes,aes(views,dislikes,colour=location,shape = location,size = dislikes))+
geom_jitter()+geom_smooth(method=lm,se=FALSE, fullrange=TRUE)+labs(title="Views Vs Dislikes",sub
title="In days")
```



```
# Views vs Comments
view_comments = videos[,.("views"=max(views),"comments"=max(comment_count)),by=.(title, locatio
n)]

ggplot(view_comments,aes(views,comments,colour=location,shape = location,size = comments))+
geom_jitter()+geom_smooth(method=lm,se=FALSE, fullrange=TRUE)+labs(title="Views Vs Comments",sub
title="In days")
```

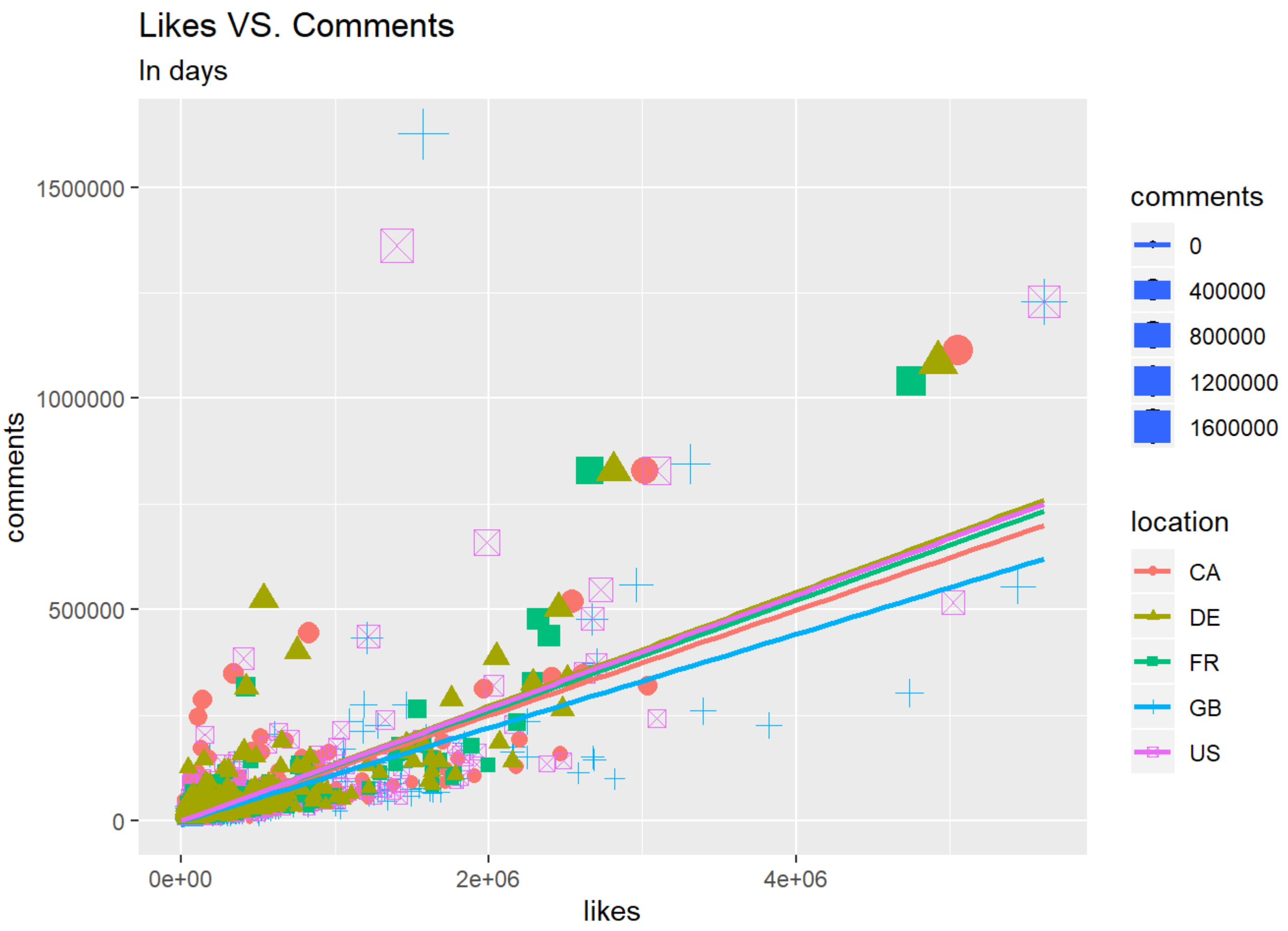


The slope of Views vs. Dislikes is much flatter than Views vs. Likes, which once again proves that dislikes is not that relevant with views. Comparing different country, French and German users have a higher ratio of like/views and comments/views. They tend to be more active users. British users are the least active in comments and likes.

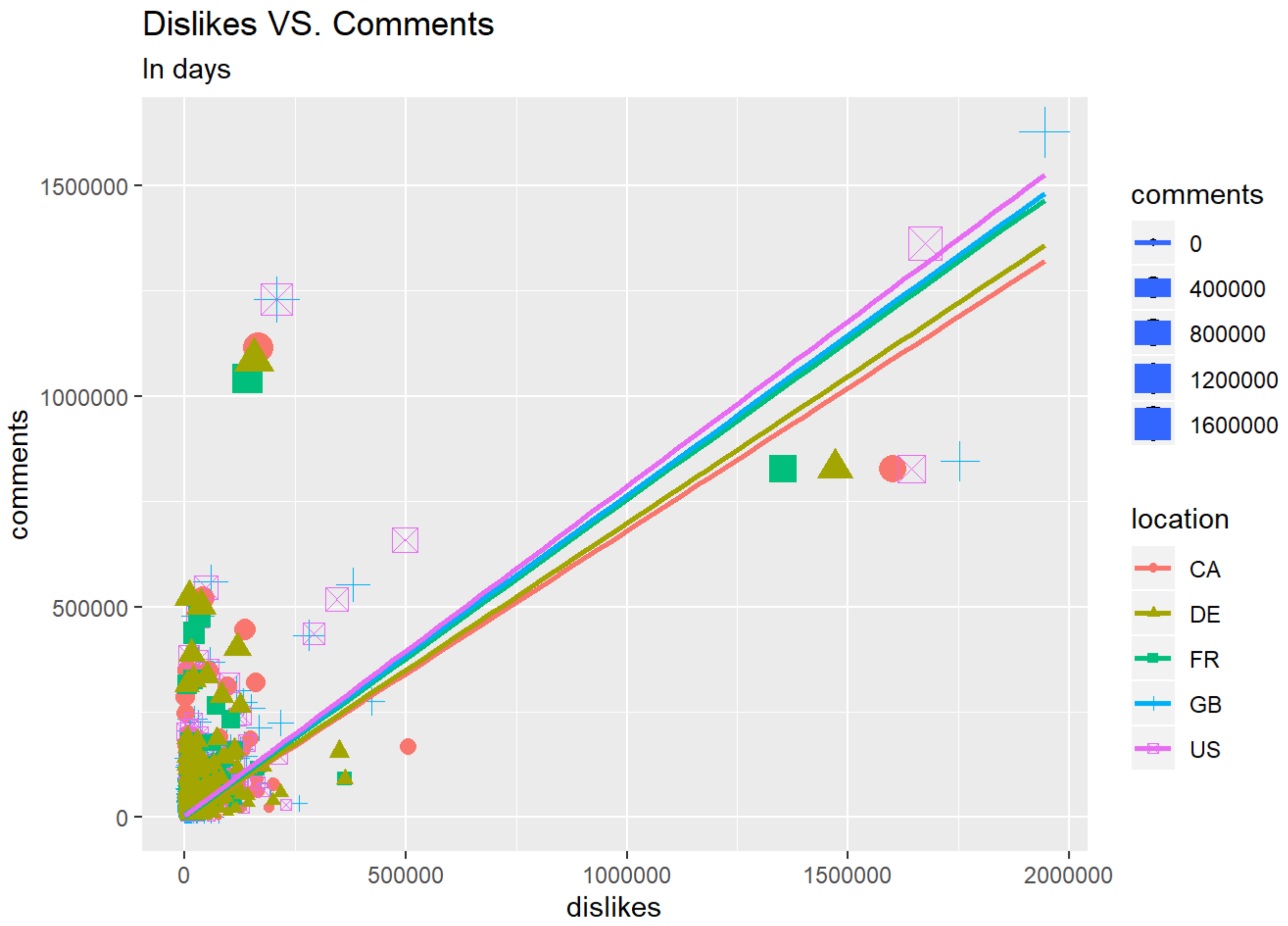
Correlation between comments and likes/ dislikes

I supposed that comments goes up with likes. The same as dislikes. Only when users explictly like/dislike a video, they will leave comments. Let’s see if my intuition is correct.

```
comments_likes = videos[,.("likes"=max(likes),"comments"=max(comment_count)),by=.(title, locatio
n)]

comments_dislikes = videos[,.("dislikes"=max(dislikes),"comments"=max(comment_count)),by=.(titl
e, location)]

p_likes = ggplot(comments_likes,aes(likes,comments,colour=location,shape = location,size = comme
nts))+
geom_jitter()+geom_smooth(method=lm,se=FALSE, fullrange=TRUE)+labs(title="Likes VS. Comments",su
btitle="In days")

p_dislikes = ggplot(comments_dislikes,aes(dislikes,comments,colour=location,shape = location,siz
e = comments))+
geom_jitter()+geom_smooth(method=lm,se=FALSE, fullrange=TRUE)+labs(title="Dislikes VS. Comments"
,subtitle="In days")

p_likes
```



```
p_dislikes
```



The trend line of dislikes vs. comments is much steeper than likes vs. comments. Youtube Users tend to leave comments when they have negative sentiments over a video.