# Zeru-Zhou-project05

September 30, 2021

## 1 Project 5 – Zeru Zhou

**TA Help:** NA

**Collaboration:** NA

- Get help from piazza
- Get help from Dr. Ward's video

### 1.1 Question 1

[74]:
```
us_youtube <- read.csv("/depot/datamine/data/youtube/USvideos.csv")
```

[3]:
```
dim(us_youtube)
```

```
1. 40949 2. 16
```

[4]:
```
head(us_youtube$trending_date)
```

```
1. '17.14.11' 2. '17.14.11' 3. '17.14.11' 4. '17.14.11' 5. '17.14.11' 6. '17.14.11'
```

[75]:
```
library(lubridate)
```

[ ]:
```
# First, change the format of those columns to date.
```

[76]:
```
us_youtube$trending_date <- ydm(us_youtube$trending_date)
```

[77]:
```
us_youtube$publish_time <- ymd_hms(us_youtube$publish_time)
```

[ ]:
```
# Second, extract the year from the columns.
```

[78]:
```
us_youtube$trending_year <- year(us_youtube$trending_date)
```

[79]:
```
us_youtube$publish_year <- year(us_youtube$publish_time)
```

[80]:
```
unique(us_youtube$trending_year)
```

```
1. 2017 2. 2018
```

[81]:
```
unique(us_youtube$publish_year)
```

1. 2017 2. 2011 3. 2015 4. 2012 5. 2010 6. 2016 7. 2009 8. 2013 9. 2008 10. 2014 11. 2018 12. 2006

[82]:
```
table(us_youtube$trending_year)
```

```

 2017  2018
 9600 31349
```

[83]:
```
table(us_youtube$publish_year)
```

```

 2006  2008  2009  2010  2011  2012  2013  2014  2015  2016  2017  2018
    1    11    14    19    27    24    44    32    35    35 10428 30279
```

[84]:
```
class(us_youtube$trending_year)
```

'numeric'

[85]:
```
typeof(us_youtube$trending_year)
```

'double'

[86]:
```
class(us_youtube$publish_year)
```

'numeric'

[87]:
```
typeof(us_youtube$publish_year)
```

'double'

[ ]:
```
# Test vectorized (They are all vectorized since they are all run on full␣
 ↪vector of data.)
```

[20]:
```
head(us_youtube$trending_date)
```

1. 2017-11-14 2. 2017-11-14 3. 2017-11-14 4. 2017-11-14 5. 2017-11-14 6. 2017-11-14

[14]:
```
head(year(us_youtube$trending_date))
```

1. 2017 2. 2017 3. 2017 4. 2017 5. 2017 6. 2017

[ ]:
```
# Without using functions above
```

[67]:
```
us_youtube <- read.csv("/depot/datamine/data/youtube/USvideos.csv")
```

[68]:
```
us_youtube$trending_year <- as.numeric(paste0("20",␣
 ↪substr(us_youtube$trending_date, 1, 2)))
```

[69]:
```
table(us_youtube$trending_year)
```

```
 2017  2018 
 9600 31349 
```

```
[70]: head(us_youtube$trending_year)
```

1. 2017 2. 2017 3. 2017 4. 2017 5. 2017 6. 2017

```
[64]: us_youtube$publish_year <- as.numeric(substr(us_youtube$publish_time, 1, 4))
```

```
[65]: table(us_youtube$publish_year)
```

```
 2006  2008  2009  2010  2011  2012  2013  2014  2015  2016  2017  2018 
    1    11    14    19    27    24    44    32    35    35 10428 30279 
```

```
[66]: head(us_youtube$publish_year)
```

1. 2017 2. 2017 3. 2017 4. 2017 5. 2017 6. 2017

All the results are expanded above. The new columns are double type and numeric class. In the provided code, all the functions are vectorized since they all run on full vector data. I got exactly the same results without using functions like “ydm”, “year”, “ymd_hms”, and “unique”, and I found that those functions such as “ydm” and “year” builds a much easier way.

## 1.2 Question 2

```
[7]: dataframe <- function(mycountry) {
         DF <- read.csv(paste0("/depot/datamine/data/youtube/", mycountry, "videos.csv"))
         DF$country_code <- mycountry
         return(DF)
         }
```

```
[9]: Countries <- c('CA', 'DE', 'FR', 'GB', 'IN', 'JP', 'KR', 'MX', 'RU', 'US')
```

```
[10]: Applied_results <- lapply(Countries, dataframe)
```

```
[11]: yt <- do.call(rbind, Applied_results)
```

```
[12]: dim(yt)
```

1. 375942 2. 17

```
[13]: colnames(yt)
```

1. ’video_id’ 2. ’trending_date’ 3. ’title’ 4. ’channel_title’ 5. ’category_id’ 6. ’publish_time’ 7. ’tags’ 8. ’views’ 9. ’likes’ 10. ’dislikes’ 11. ’comment_count’ 12. ’thumbnail_link’ 13. ’com-

ments_disabled' 14. 'ratings_disabled' 15. 'video_error_or_removed' 16. 'description' 17. 'country_code'

[14]:
```r
library(lubridate)
```

```

Attaching package: ‘lubridate’


The following objects are masked from ‘package:base’:

    date, intersect, setdiff, union

```

[15]:
```r
yt$trending_year <- year(ydm(yt$trending_date))
```

[16]:
```r
yt$publish_year <- year(ymd_hms(yt$publish_time))
```

Column “country code” is added. Columns of yt is printed, and yt has 375942 rows and 17 columns. Columns trending_year and publish_year is created.

## 1.3 Question 3

[17]:
```r
table(yt$publish_year)
```

```

  2006   2007   2008   2009   2010   2011   2012   2013   2014   2015   2016 
     1     37     59     32     62    117     43    127    121    159    153 
  2017   2018 
 88865 286166 
```

[21]:
```r
yt$trending_date <- ydm(yt$trending_date)
```

[22]:
```r
yt$publish_time <- ymd_hms(yt$publish_time)
```

[18]:
```r
table(yt$trending_year)
```

```

  2017   2018 
 84424 291518 
```

[23]:
```r
yt[yt$publish_year == 2006, ]
```

A data.frame: 1 x 19

| | video_id<br>&lt;chr&gt; | trending_date<br>&lt;date&gt; | title<br>&lt;chr&gt; | channel_<br>&lt;chr&gt; |
|---|---|---|---|---|
| 351288 | MJO3FmmFuh4 | 2018-02-05 | Budweiser - Original Whazzup? ad | dannotv |

The name (title) of the video is "Budweiser - Original Whazzup? ad", and it took 2018-2006=12 years to trend.

### 1.4 Question 4

```
[24]: table(yt$ratings_disabled)
```

```
 False  FALSE   True   TRUE
200214 168420   2096   5212
```

```
[25]: yt$ratings_disabled_combined <- NA
```

```
[26]: yt$ratings_disabled_combined[yt$ratings_disabled == "FALSE"] <- FALSE
```

```
[27]: yt$ratings_disabled_combined[yt$ratings_disabled == "False"] <- FALSE
```

```
[28]: yt$ratings_disabled_combined[yt$ratings_disabled == "TRUE"] <- TRUE
```

```
[29]: yt$ratings_disabled_combined[yt$ratings_disabled == "True"] <- TRUE
```

```
[30]: table(yt$ratings_disabled_combined, useNA="always")
```

```
 FALSE   TRUE   <NA>
368634   7308      0
```

```
[31]: class(yt$views)
```

'integer'

```
[32]: tapply(yt$views, yt$ratings_disabled_combined, mean)
```

**FALSE** 1338147.56100631 **TRUE** 742478.86302682

The average number of views for videos with ratings enabled and those with ratings disabled are 1338148 and 742479, respectively. As a result, it looks like disabling the rating hurts the views.

### 1.5 Question 5

```
[33]: yt$balance <- (yt$likes) - (yt$dislikes)
```

```
[34]: yt$positive_balance <- NA
```

```
[35]: yt$positive_balance[yt$balance > 0] <- TRUE
```

```
[36]: yt$positive_balance[yt$balance <= 0] <- FALSE
```

[37]:

```r
table(yt$positive_balance, useNA="always")
```

```
FALSE   TRUE   <NA> 
14359 361583      0 
```

There are 361583 videos that have a positive balance.

## 1.6 Question 6

[38]:

```r
tapply(yt$comment_count, yt$positive_balance, mean )
```

**FALSE** 3066.6932237621 **TRUE** 4300.91580909501

[39]:

```r
tapply(yt$views, yt$positive_balance, mean)
```

**FALSE** 657892.011073195 **TRUE** 1353122.38724719

I choose mean as the statistic for both comment_count and views. This is because the mean shows the average level of the number of views/comments that could be compared between different groups of data (In this case, there are 2 groups: with and without positive balance). The videos with positive balance tends to have much more views and slightly more comments than videos without positive balance.

## 1.7 Pledge

By submitting this work I hereby pledge that this is my own, personal work. I've acknowledged in the designated place at the top of this file all sources that I used to complete said work, including but not limited to: online resources, books, and electronic communications. I've noted all collaboration with fellow students and/or TA's. I did not copy or plagiarize another's work.

> As a Boilermaker pursuing academic excellence, I pledge to be honest and true in all that I do. Accountable together – We are Purdue.